IEICE TRANS. INF. & SYST., VOL.E92–D, NO.2 FEBRUARY 2009

PAPER

Transcoding-after-Smoothing System for VBR MPEG Video Streaming*

I Gusti Bagus Baskara NUGRAHA[†a)] ***and*** **Hiroyoshi MORITA**[†], ***Members***

SUMMARY Delivering video streaming service over the Internet encounters some challenges. Two of them are heterogeneity of networks capacity and variability of video data rate. The capacity of network segments are constrained. Meanwhile, the rate of video data to be transmitted is highly variable in order to get near-constant images quality. Therefore, to send variable bit rate (VBR) video data over capacity-constrained network, its bit rate should be adjusted. In this paper a system to adjust the transmission bit rate of VBR MPEG video data called Transcoding-after-Smoothing (TaS), which is a combination of bit rate transcoding and bit rate smoothing algorithm, is proposed. The system smoothes out transmission rate of video data while at the same time also performs transcoding on some video frames when necessary in order to keep the transmission bit rate below a certain threshold value. Two kinds of TaS methods are proposed. One method does not have transcoding preference, while the other method uses frame type preference where an intra-coded frame is the last one that will be transcoded. These methods are implemented in our video server where a VBR video data is accessed by a client. Our experiment results show that the first TaS method yields significant reduction in the number of transcoded frames compared with the second TaS method and conventional frame-level transcoding. However, the second method performs better in minimizing the quality distortion.

key words: *variable bit rate video streaming, bit rate smoothing, bit rate transcoding*

1. Introduction

MPEG video is one video compression format which is currently supported by some popular video streaming systems, such as VideoLAN [1], QuickTime [2], and RealNetworks [3]. One type of MPEG encoding scheme is variable bit rate (VBR) video, such as in MPEG-2. VBR MPEG video has better or at least equal coding performance than constant bit rate (CBR) one to gain near-constant images quality. Depending on the content and encoding method, the size of encoded frames highly vary. For example, in MPEG-2 coding one type of frames called I-frame usually has larger size than the other type of frames (P-frame and B-frame) [4].

Given the variability of encoded video bit rate, one problem arises when video data has to be delivered over a bit rate constrained network where the maximum amount of data allowed to be sent at time is restricted. This restriction may occur due to the technology used or traffic congestion. Sometimes the condition of network traffic is difficult to predict so that the streaming server must be able to adjust transmission bit rate on-the-fly. There exist three well-known mechanisms proposed to adjust transmitted video bit rate. They are scalable video coding (SVC) [5], multiple description coding (MDC) [6] and bit rate transcoding [7].

In SVC, an uncompressed video data is encoded into at least two layers. They are base layer and one or more enhancement layers. The base layer data is the lower bit rate version of the video data. Low quality of video presentation will be perceived when only decoding the base layer data. The enhancement layer data is obtained by encoding the difference to get the desired bit rate. It cannot be decoded independently. Decoding both base layer and enhancement layer data yields better quality of reconstructed video. In video transmission, base layer data is transmitted when the available capacity of network is low. Enhancement layers are sent altogether with the base layer data when there is enough capacity available on the network. SVC is part of MPEG-2 [8], MPEG-4 [9] and H.264 [10] standard.

SVC has drawbacks. First, video bit rate is adapted during the encoding process. The bit rate of base layer data and enhancement layer stream must be decided in advanced before encoding. The bit rate of any individual layer stream cannot be changed on an intermediate node along communication path without doing re-encoding which is costly. So, it is not adequate for our purpose. Second, the coding efficiency is less than the single layer version. To send the same resolution of video data, the output bit rate of scalable video encoder is higher than the video encoder without layering.

In MDC, one unit of video data, e.g., one frame, might be transformed into multiple chunks of new representations or descriptions. While SVC needs hierarchical decoding, MDC decoder can reconstruct any combination of descriptions. Decoding parts of these descriptions may result in lower quality of reconstructed image. Transmission bit rate adjustment is performed by regulating the number of descriptions to be sent. According to [11], some practical implementations of MDC use Reed-Solomon coding [12], priority encoded transmission [13], and optimized bit allocation [14]. However, like SVC, MDC produces more data than the single layer version of encoded video data.

Bit rate transcoding takes another approach by altering the encoded video data to get desired bit rate. Numer-

Manuscript received June 2, 2008.
Manuscript revised September 23, 2008.

[†]The authors are with the Graduate School of Information Systems, The University of Electro-Communications, Chofu-shi, 182–8585 Japan.

*This paper is an extension of works that were presented at The 16th International Conference on Computer Communications and Networks 2007, Honolulu, USA [36], and at The 30th Symposium on Information Theory and Its Applications 2007, Kashikojima, Japan [37].

a) E-mail: baskara@is.uec.ac.jp

DOI: 10.1587/transinf.E92.D.298

ous methods of video transcoding had been proposed [4], [15]. Most proposals had presented transcoding at frame-level where an encoded frame will be modified when its size exceeds a certain value. The advantage of transcoding over SVC and MDC is that on its coding efficiency. However, modifying the content of a coded frame is not simple because dependency exists among frames. Altering some data in a frame may introduce error at other frames. In MPEG VBR video, the quality distortion affected by transcoding may be noticeable since the size of key frames is usually very large compared with the other frames. Because the size of these key frames is large, there is higher probability that the frames will be transcoded. If most of key frames must be transcoded, then most of frames in a video sequence will suffer from quality degradation. Hence, in order to maintain images quality, fine adjustment in transcoding process is required that makes the transcoding algorithm more complex.

Instead of doing fine adjustments, we take a different approach by combining transcoding and smoothing. The motivation behind this approach is that the quality of the images can be maintained if the number of frames required to be transcoded as well as the amount of bits taken a frame are reduced. To achieve those requirements, we make use of buffering process that are used during video data reception in order to compensate jitter. In this case, video server needs to run bit rate smoothing algorithm to prevent user's buffer from either underflow or overflow.

Bit rate smoothing is a lossless process where bits from some frames are arranged prior to transmission so that the server transmits almost the same amount of bits at any time. Smoothing algorithm solely cannot ensure the transmitted video bit rate will fit the transmission requirement. Transcoding is still required. A combination system between smoothing and transcoding is expected to maximize reconstructed video quality and at the same time also to comply with the transmission bit rate restriction.

Integrating bit rate smoothing and transcoding can be performed with two kinds of methods. They are by doing *smoothing after transcoding (SaT)*, or reversely by doing *transcoding after smoothing (TaS)*. Transcoding is a lossy process, meanwhile smoothing is a lossless process. Therefore, SaT method performs lossy process first, while TaS method performs the lossy process as the last resort. From this point of view, it is clear that SaT method does not gain any advantage because any frame which size exceeds certain threshold will be transcoded first ahead of smoothing. Hence, the number of transcoded frames is expected to be same as in the conventional transcoding system. Meanwhile, if the video bit rate are smoothed out first prior to transcoding process, then there is less possibility that the important frames, e.g., I-frames, will be transcoded. In addition, when a video frame must be transcoded the number of bits truncated from the frame will be less than in transcoding system solely. The less the number of transcoded frames, particularly the most important ones, the better the quality on the reconstructed images.

This paper discusses practical video streaming system for *VBR MPEG video* that combines transcoding and smoothing process, which is based on smoothing technique in *online bit rate smoothing algorithm* [16]. In this system, a video server first tries to make the transmission rate as smooth as possible. During the smoothing process when the bit rate at a specific time is still larger than a threshold value, transcoding process is executed. So, transcoding is used as the last choice to obtain the desired bit rate. Moreover, transcoding is conducted on GOP-level instead on frame-level so that the transcoder can choose to transcode the less important frames first, e.g., B-frames, in a GOP. The I-frame will be transcoded when the remaining number of bits from truncating all of the B and P-frames in the GOP is still larger than the threshold value.

Smoothing is closely related to buffer management. There exist some proposals on video transmission that present some techniques on transcoding that consider buffer management, such as [17], [18]. Although it has similarity, the main purpose of the buffer management in those references is mainly to prevent the decoder buffer from underflow and overflow. Transcoding is still their main tool for rate control. Meanwhile, in our system the buffer is used as a part of smoothing algorithm to make the transmission rate as smooth as possible. Moreover, smoothing process is given higher priority over transcoding in order to reduce data loss.

VBR video transcoding has been proposed in [19], [20]. Reference [19] performs transcoding on frame-level, meanwhile [20] performs it on GOP-level. Proposed methods in both references did not perform smoothing prior to transcoding.

Our method is implemented in a video streaming server and evaluated in our laboratory's local area network environment. Experiment results using various scenarios show that most of the frames do not need to be transcoded to conform the channel capacity.

The organization of this paper is as follows. Section 2 explains briefly about video bit-rate transcoding and bit rate smoothing algorithm. Section 3 explains our proposed system. Section 5 shows the experiment results. And, Sect. 6 gives our conclusion.

2. Video Bit Rate Smoothing and Transcoding

2.1 Video Bit Rate Smoothing

Bit rate smoothing is closely related with buffering. MPEG-2 has buffer specification called Video Buffering Verifier (VBV) as described in Annex C of the MPEG-2/ITU-T H.262 recommendation [21]. VBV is a hypothetical decoder which is conceptually connected to the output of an encoder. In VBR case, data enters the buffer at R_{max} if the buffer is not full, where R_{max} is the maximum bit rate specified in the bit_rate field of the sequence header. If the VBV buffer becomes full after filling at R_{max} for some time, no more data enters the buffer until some data is removed from the buffer. After some data is removed from the buffer, the data

input continues at R_{max}. Where there are skipped frames, the frame is not decoded and the previously decoded frames will be displayed until normal operation of VBV can resume. The VBV constraints ensure both encoder buffer and decoder buffer never overflow and underflow.

However, the transmission bit rate may highly fluctuate from 0 to R_{max}. This burstiness makes the use of network utilization less efficient. To reduce the bit rate variability, bit rate smoothing algorithm is used in this paper.

In case of live video streaming, sophisticated smoothing algorithms had been proposed in [16], [22]. In this paper the algorithm in [16], an extension of optimal offline smoothing algorithm proposed in [23], is used. The choice is made based on our previous works that had been implementing this algorithm successfully in real live video streaming scenarios [24]–[26]. The algorithm has been proven to be optimal in smoothness with the lowest possible rate variability.

Essentially the algorithm works as follows. A video server prepares a buffer of size b_s that can handle w video frames. The client has a receiving buffer of size b_c. The server reads video data and load one frame into its buffer. During loading the t^{th} frame, the server measures the size of that frame $d(t)$ and calculates the accumulated frame size $D(t)$ which represents transmission rate's lower bound, where $D(t) = D(t-1) + d(t)$ and $d(0) = 0$. In addition, the server also calculates the rate's upper bound $B(t)$, where $B(t) = D(t-1) + b_c$. $D(t)$ and $B(t)$ are calculated until w frames are loaded into server's buffer.

Based on $D(t)$ and $B(t)$ curves, smoothing algorithm in [23] is performed. Two curves, c_{max} and c_{min}, are created between $B(t)$ and $D(t)$ curves. Those c_{max} and c_{min} curves represent feasible maximum transmission rate and feasible minimum transmission rate within a time interval, respectively. The values of c_{max} and c_{min} are chosen so that the client's buffer will not overflow when sending data at c_{max}, and the buffer will never underflow when sending data at c_{min} within the interval. Data will be sent at a rate of either c_{max} or c_{min}. When client's buffer starvation will happen at a point, transmission rate at a time segment prior to that point is c_{max}. Meanwhile, when the buffer will be full at a point, the transmission rate at the segment prior to that point is set to c_{min}. In VBV model, the data is always transferred at rate R_{max}.

End-to-end delay variation or jitter may occur when transporting data from server to client. Jitter can cause buffer underflow at the client since the expected video data may arrive beyond the playing time deadline. To accommodate jitter, both $B(t)$ and $D(t)$ are shifted ω frametime to the right where ω is the maximum value of jitter experienced by the client in frametime unit. Then, smoothing algorithm is performed between $B(t)$ and the unshifted $D(t)$.

The output of the algorithm is a new transmission schedule $S(t)$ that represents the amount data should be transmitted at time t where $1 \leq t \leq w$. All bits that have scheduled to be transmitted at time $1 \leq t \leq \alpha$, where $\alpha \leq$ w, are sent sequentially. Since $S(\alpha)$ bits of data have been transmitted, the next α frames of video data are loaded into server's buffer and again $D(t)$ and $B(t)$ are calculated. After sliding smoothing window α time units to the right, smoothing algorithm is performed once more. This process is repeated until no frame data left to be sent.

However, although the online smoothing process described above emulates the dynamics of the client's buffer, it still cannot ensure that the client's buffer will never underflow in case of network congestion. Network nodes may drop some packets due to congestion. This will cause buffer underflow at client. A feedback from the client that indicates the status of the client's buffer is required to handle such case. The choice of transport protocol also affects the buffering mechanism. For example, the use of Transmission Control Protocol (TCP) [27] in congested network may fluctuate the transmission rate because TCP must reduce the rate when congestion occurs. TCP also has to retransmit lost packets that may make the decoder to wait for the retransmitted data before removing data from client's buffer. In this paper, we do not implement any feedback mechanism. We assume that no packet is lost during transmission and the transport protocol used in the experiment is User Datagram Protocol (UDP) [28], which has no feedback control mechanism.

2.2 Video Bit Rate Transcoding

Employing smoothing algorithm solely cannot ensure that the transmission bit rate will not exceed required rate threshold. Depending on buffer size, fluctuations occur. At some frametimes, sometimes the transmission rate still exceeds the rate threshold. Therefore transcoding is performed to reduce the transmitted data rate at those frametimes.

To reduce video bit rate at a frametime, the number of codewords required to represent an image in the video sequence must be reduced and the length of the codewords must be shorten. Numerous methods of video transcoding have been proposed [7], [15], [29]–[34]. They mainly use either requantization or truncation of high frequency coefficients, including some enhanced techniques to get lower distortion.

Requantization is a method to adjust the quantization parameters to new values so that the length of codewords required to code quantized DCT coefficients is shorter. The transcoder performs variable length decoding, inverse quantization, quantization, and variable length encoding process. In high-frequency truncation, both variable length decoding and inverse quantization are not required. The transcoder requires a video bit stream parser to locate the coefficients to be dropped out. After truncation, the transcoder tailors the remaining bits into a new stream.

For live video streaming that requires faster processing, the latter transcoding method is chosen. Although its implementation is very simple, the combination of this high-frquency truncation method and smoothing algorithm can still produce high quality of images as it will be shown on our experiment results in Sect. 5. The following section will

explain the procedure on how to implement this combination.

3. Transcoding-after-Smoothing (TaS)

In this section, our proposal is explained. Transcoding-after-smoothing (TaS) is a combination of transcoding and smoothing algorithm to achieve desired transmission rate. TaS system involves a server that runs smoothing and transcoding algorithm, client, and a packet network connecting the server and the client. Packet transmission over the network is assumed to be lossless.

Figure 1 draws block diagram of TaS system that consists of bit rate smoothing algorithm, transcoder, and sender. Based on the figure, TaS procedure can be explained as follows, where variables used in the procedure including their definitions are written in Table 1.

Step 1: Parsing
The size of w consecutive video frames is calculated by video parser. We note frame size as $d(t)$ and frame size vector as $\bar{d} = \{d(t)\}$ and $1 \le t \le w$. The vector $\bar{d}$ is then sent to smoothing algorithm and all w frames data are copied into sender's buffer.

Step 2: Smoothing
The smoothing algorithm tries to smooth out the values in $\bar{d}$ with constraints on client's buffer size b bytes. The smoothing algorithm produces a vector $\mathbf{s} = \{s(t)\}$ where $0 \le t \le w-1$, which represents the amount of data in bytes that should be sent at time t.

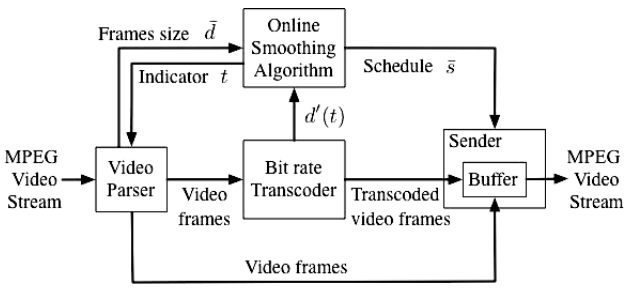


Fig. 1 Block diagram of smooth-transcoding system.

Step 3: Transcoding
During the smoothing process, at each step the smoothing algorithm checks the smoothing output $s(t)$. if $s(t)$ is greater than a threshold value C, then smoothing process is halted and an indicator t is sent to video parser. After that, video parser sends the t-th frame to bit rate transcoder. The transcoder will modify this frame to get smaller frame size. Once the frame had been transcoded, then the new frame size $d'(t)$ is transferred to the smoothing algorithm and **s** is recalculated. The data of the t-th frame in sender's buffer is overridden by the new transcoded frame data.

Step 4: Sending
The transmission schedule of the first α frametimes, which is $s(1) \ldots s(\alpha)$, is sent from smoothing algorithm to the sender. Using this information, the sender sends $s(1)$ bytes of data at frametime $t = 1$, $s(2)$ bytes of data at frametime $t = 2$, and so on until frametime $t = \alpha$. While sending, the next α consecutive frames are loaded by video parser and their frames size is calculated to form a new frame size vector $\bar{d}$. Then, the process from Step 1 is repeated.

In order to give a more clear illustration of TaS process, an example is given on Fig. 2. The example is described as follows.

1. Initially, based on input vector $\bar{d}$ and client's buffer size b, two variables $D(t) = D(t-1) + d(t)$ and $B(t) = D(t-1) + b$ for $1 \le t \le w$ are calculated, where $D(0) = 0$. The constants C, w, and α are also fixed in advance. Figure 2 (a) shows the curves C, $D(t)$, and $B(t)$. The values of w and α are set to 4 and 2, respectively.
2. The algorithm calculates the values of c_{max} and c_{min} according to their definitions on Table 1 step-by-step from $t = 1$ to $t = w$. At $t = 1$, the curves c_{max} and c_{min} are depicted on Fig. 2 (b).
3. The process continues for $t = 2$ so that we get a new value of c_{max} and c_{min} as shown on Fig. 2 (c).
4. When calculating c_{min} for $t = 3$, the curve of c_{min} is not feasible since $c_{min} > B(3)$ (see Fig. 2 (d)) or in other words the client's buffer will overflow when data is sent at rate c_{min}.
5. Since c_{min} is not feasible, the calculation stops at t_D

Table 1 Definition of variables in TaS algorithm.

Variable	Definition
w	The length of smoothing window.
α	Slide length.
N	Total number of frames in a video sequence.
$d(t)$	Size of t^{th} frame in bytes.
$D(t)$	Cumulative frame size over $[1, t]$. $D(t) = \sum_{i=1}^{t} d_i$.
b	Client buffer size.
$B(t)$	Maximum cumulative frame size that can be received by the client over $[1, t]$ without overflowing the buffer, or $B(t) = \min\{D(t-1) + b, D(N)\}$.
c_{max}	The maximum transmission rate over an interval $[a, b]$ without overflowing client buffer.
t_B	The latest time t at which the client buffer is full when the server transmits at c_{max} over $[a, b]$.
c_{min}	The minimum transmission rate over an interval $[a, b]$ such that the client buffer never starves.
t_D	The latest time t at which the client buffer is empty when the server transmits at c_{min} over $[a, b]$.
C	Target rate.

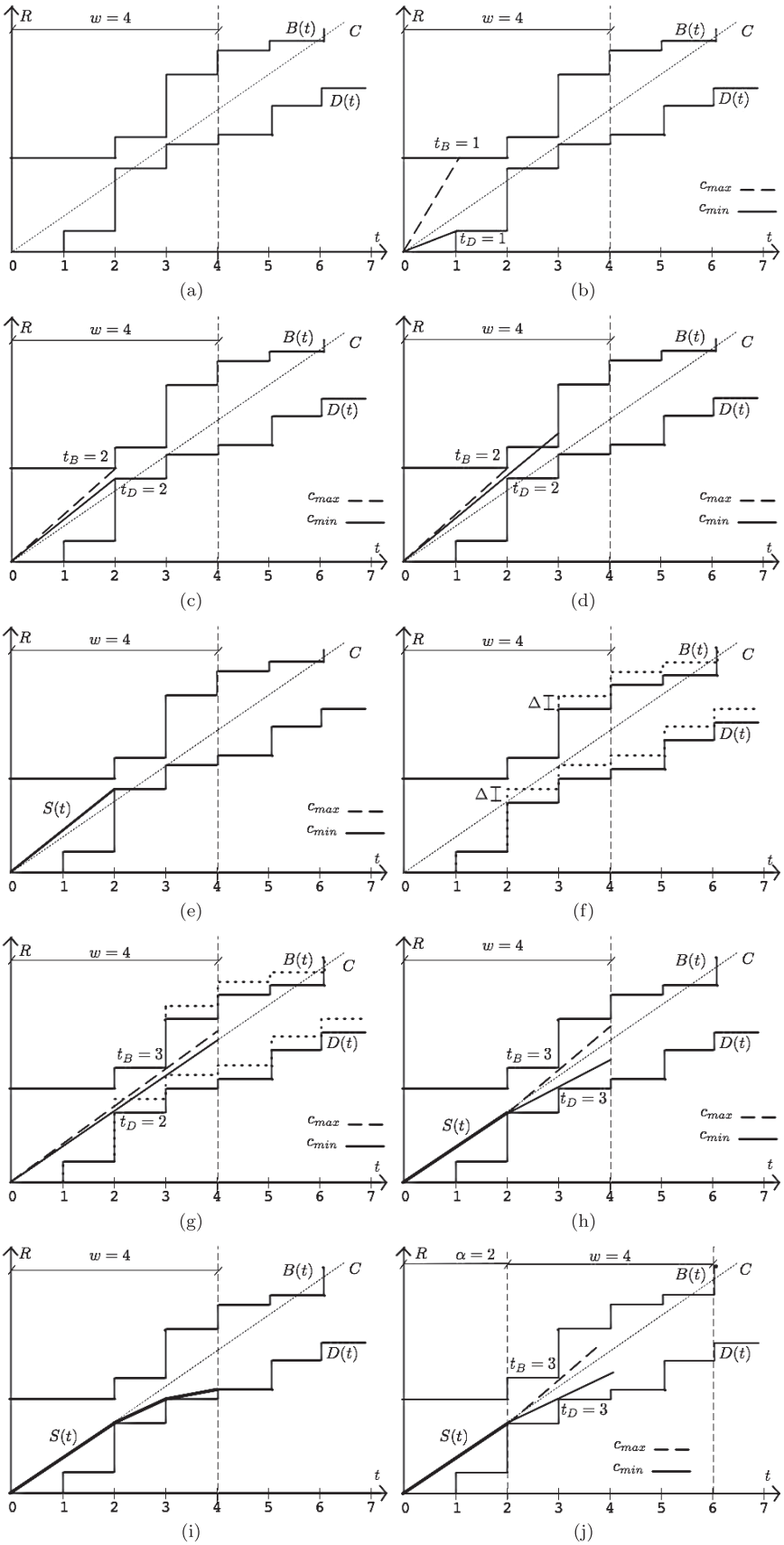


Fig. 2 An example of TaS process for $w = 4, \alpha = 2$.

Algorithm 1 TaS Algorithm

1: $t_s = 0, t_e = 1, q = 0$
2: $c_{max} = b, t_B = 1, c_{min} = d(1), t_D = 1$
3: REPEAT
4: Set $t'_e = t_e + 1$
5: IF $c_{max} < \frac{D(t'_e)-(D(t_s)+q)}{t'_e-t_s}$
6: IF $c_{max} > C$
7: $B(t_B) = \max\{C(t_B - 1), D(t_B)\}$
8: IF $B(t_B) > C$
9: $c_{tr} = Transcode(t_B^{th}$ frame)
10: ENDIF
11: $t_e = t_s + 1, t_B = t_s + 1, t_D = t_s + 1$
12: GOTO line 4:
13: ENDIF
14: Output segment $\langle t_B - t_s, c_{max}\rangle$
15: $t_s = t_B, t_e = t_B + 1, q = B(t_B) - D(t_B)$
16: ELSEIF $c_{min} > \frac{B(t'_e)-(D(t_s)+q)}{t'_e-t_s}$ OR $t'_e = N$
17: IF $c_{min} > C$
18: $c_{tr} = Transcode(t_D^{th}$ frame)
19: $t_e = t_s + 1, t_B = t_s + 1, t_D = t_s + 1$
20: GOTO line 4:
21: ENDIF
22: Output segment $\langle t_D - t_s, c_{min}\rangle$
23: $t_s = t_D, t_e = t_D + 1, q = 0$
24: ELSE
25: Set $t_e = t'_e$
26: ENDIF
27: Compute c_{max}, t_B, c_{min}, and t_D over $[t_s, t_e]$
28: UNTIL $t_s = N$

where $t_D = 2$. At this step, new curve $S(t) = c_{min}$ for $0 \le t \le t_D$ is obtained as shown on Fig. 2 (e).

6. However, since $S(t)$ is larger than C, the value of $D(2)$ must be adjusted to C as illustrated on Fig. 2 (f). This means that the frame at $t = 2$ must be transcoded.
7. After transcoding, a new value of $d(2)$ is obtained. Consequently, $D(t)$ and $B(t)$ must be recalculated. After updating the curves of $D(t)$ and $B(t)$, the calculation of c_{max} and c_{min} is repeated from $t = 1$ again so that we get the result as depicted on Fig. 2 (g) and 2 (h).
8. When $t = w$ is reached, the final curve of $S(t)$ is obtained.
9. At the next step, the window is shifted α frames to the right, where $\alpha = 2$ in this example. The calculation is then repeated from step a. for $t = 3, \ldots, 6$.

Formal description of TaS algorithm is written on Algorithm 1, which is derived from [23].

There is a possibility that $D(t)$ is still larger than C after transcoding the frame. In this case, the last value of $D(t)$ obtained will be used.

4. TaS with Frame Type Preference

TaS procedure in the previous section explains that during the smoothing process, when $D(t) > C$, the frame at frametime t will be transcoded. That is an ideal situation where a frame of size $D(2)$ (see Fig. 2) can be downsized exactly to C. In the real situation, the exact target size cannot always be obtained due to the variable length encoding system used in video encoder. Getting an exact value sometimes require a lot of computation. The minimum frame size achieved after transcoding even sometimes still larger than target value, especially on I-frame.

Therefore, instead of only transcoding one frame at frametime t, one or more frames from frametime t to frametime $t - \delta$ within smoothing window can be transcoded. Frametime range between $t - \delta$ and t is called *transcoding window*. To transcode multiple frames, the TaS procedure at Step 3 is modified as follows.

During the smoothing process, at each step the smoothing algorithm checks the smoothing output $s(t)$. if $s(t)$ is greater than a threshold value C, then smoothing process is halted and an indicator t is sent to video parser. After that, video parser sends the frames within transcoding window, from $t - \delta$-th frame to t-th frame, to bit rate transcoder. Frames are processed sequentially from t downward to $t - \delta$ to get their frame types. If frame type at frametime t is B-frame, then that frame will be transcoded. $D(t)$ is then recalculated. If $D(t)$ is still greater than C, then the process checks the frame type at frametime $t - 1$. If the frame type at frametime $t - 1$ is B-frame, then the frame at frametime $t - 1$ will be transcoded, and so on. If all of B-frames in the transcoding window had been transcoded but $D(t)$ is still greater than C, then the pointer is set to frametime t again, and now P-frames should be transcoded. I-frame will be the last frame to be transcoded. If $D(t)$ is still larger than C after transcoding all of the frames in the transcoding window. In this case, then the last value of $D(t)$ obtained will be used. Once the frames had been transcoded, then the new frames size $d'(t - \delta) \ldots d'(t)$ is transferred to the smoothing algorithm and $\mathbf{s}$ is recalculated. The frames data from the $t - \delta$-th frame to t-th frame in sender's buffer is overridden by the new transcoded frame data.

The idea on giving transcoding priority based on frame type has actually been proposed in [18], [35]. Both references proposed transcoding by applying different compression rate when transcoding each frame type. I-frames are given the lowest compression ratio and B-frames get the highest compression ratio. The difference between their method and our method is that in our method multiple frames are transcoded within one transcoding step in order to obtain desired bit rate at a specific frametime.

5. Experimental Performance Evaluation

5.1 Experiment Setup

An experiment is set up that consists of a video server and a client. The server is an Apple's PowerMac computer with PowerPC G5 Quad 2.5 GHz processors and 2 GB of memory. The client's machine is an Apple's macbook computer with 1.83 GHz Intel Core Duo processor and 1 GB of memory. Video server and the client are connected using gigabit ethernet in our lab's LAN environment.

The input of the video server is MPEG-2 video elementary stream recorded from a broadcasted television program. The resolution is 720×480 pixels at 29.97 frames per sec-

ond. So, the frametime interval is about 33.3 ms. The number of frames to be transmitted is set to 1000 frames for the ease of the presentation of our results in this paper. This video data is transmitted to client by using User Datagram Protocol (UDP) over Internet Protocol (IP) protocol.

The TaS system drawn on Fig. 1 is implemented in video server. The video parser used in this experiment is based on MPEG-2 decoder developed by MPEG Software Simulation Group [†]. The bit rate transcoder implemented is the simplest one, which works by truncating all AC coefficients in I-frame or all coefficients in P-frame or B-frame except the first one in each block. So only the first coefficient is retained when transcoding is performed on a frame.

At the client, a multimedia player called *mplayer* [††] is installed. The transmitted MPEG video data will be decoded and displayed on the client's screen by using this software.

The performance of TaS system will be compared with conventional transcoder. For TaS, two kinds of TaS experiments are set up. First, TaS *without preference* as explained in Sect. 3 where a frame which size exceeds bit rate threshold will be transcoded without considering its frame type. Second, TaS *with preference* as illustrated in Sect. 4 where frame type is taken into consideration. Here B frames in smoothing window are tried to be transcoded first followed by P frames and lastly I frames. To simplify writings, we called the first experiment as *TaS-1* and the latter one as *TaS-2*. Conventional transcoding is performed by using two methods. The first method is by using a software called *ffmpeg* [†††]. This software is a well-known tool for multimedia data conversion. It is actually not a "real" transcoder since ffmpeg decodes the encoded video data first and then re-encodes the decoded data back to an encoded video data with different compression settings. The second transcoding method is by discarding all DCT coefficients in every block in a frame, except the first coefficient in the block. Let note the first transcoding method as *TC-1* while the latter one as *TC-2*.

Four performance metrics will be evaluated. They are output bit rate, number of transcoded frames, quality distortion, and processing time. The result on output bit rate states the number of bit needed to be transmitted within a frametime interval. The bit rate should be lower than a given rate threshold C. In case of the number of transcoded frames, less number of transcoded frames is preferable to minimize distortion. Processing time is evaluated to figure out the feasibility of implementation.

5.2 Performance Evaluation

5.2.1 Output bit rate

Figure 3 shows the original transmission schedule of a 1000 frames of an MPEG-2 video data. Every δ seconds the server transmits $a(t)$ bytes of video data. The maximum, minimum, and average values are 103,990 bytes, 4,132 bytes, and 25,045.5 bytes per frametime, respectively.

Various combinations of values of buffer size b, smoothing window size w, and sliding window α are used in this experiment as shown on Table 2. However, because all of the results are similar and due to space constraint, only a specific case will be shown.

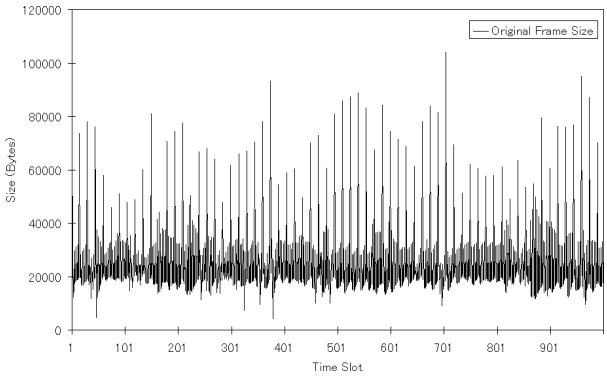


Fig. 3 Original transmission rate.

Table 2 Experiment variables.

Variable	Values
w	30 and 50 frames
α	10 and 30 frames
N	1000 frames
b	1 MB and 512 KB
C	10000, 15000, 20000, and 30000 bytes

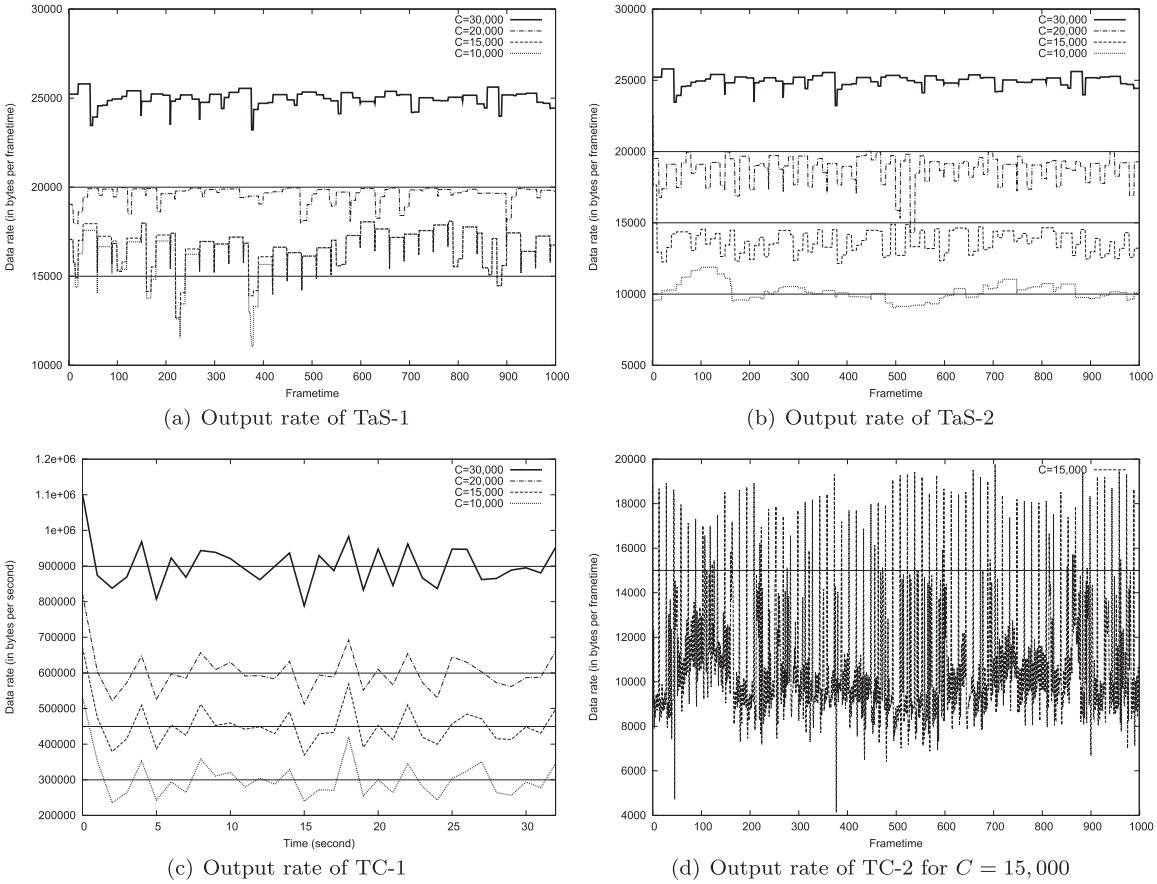
The output data rate of TaS-1, TaS-2, TC-1, and TC-2 are shown on Fig. 4. For TaS-1 and TaS-2, the figure shows the output rate for b = 1 MB, w = 30 frametime, and α = 10. The rate threshold C is set at 10,000, 15,000, 20,000 and 30,000 bytes per frametime.

The figure shows that TaS-2 produces less bit rate in average compared with TaS-1. Both TaS-1 and TaS-2 are able to deliver output rate that is under the threshold in case of C = 20,000 and C = 30,000 bytes per frametime. However, in case of C = 15,000 TaS-2 performs better where the output rate still can be limited below the threshold. For C = 10,000, the output rate at some points is above the threshold.

In case of TC-1, the output rate is shown in bytes per second instead of bytes per frametime because the transcoder, which is ffmpeg, takes input value in bits per second (bps) unit as threshold value. Figure 4 (c) shows that for all of the values of C, at some points the output rate is still higher than the desired threshold value. Meanwhile, in case of TC-2, because the graph produces bursty pattern of output rate, only one result on a certain value of C is depicted. The output rate for C = 15,000 is given on Fig. 4 (d) where sometimes the data rate is still above the threshold value. This result is similar for C = 10,000. For C = 30,000 and C = 20,000, the output rate is always below the threshold in spite of its burstiness.

From all of the above results on output rate, we can

[†]http://www.mpeg.org/MSSG/
[††]http://www.mplayerhq.hu/
[†††]http://ffmpeg.mplayerhq.hu



(a) Output rate of TaS-1

(b) Output rate of TaS-2

(c) Output rate of TC-1

(d) Output rate of TC-2 for $C = 15,000$

Fig. 4 Output rate of TaS-1 and TaS-2 for $b = 1$ MB, $w = 30$ frametime, and $\alpha = 10$ frametime, and also TC-1 and TC-2.

Table 3 Number of transcoded frames.

Threshold	TaS-1				TaS-2				TC-1				TC-2			
(bytes)	I	P	B	Total	I	P	B	Total	I	P	B	Total	I	P	B	Total
10,000	66	155	174	395	67	267	666	1000	67	267	666	1000	67	267	660	994
15,000	66	153	164	383	35	175	666	876	67	267	666	1000	67	267	581	915
20,000	65	86	26	177	4	39	577	620	67	267	666	1000	67	265	100	432
30,000	1	0	0	1	1	0	0	1	67	267	666	1000	67	192	3	262

conclude that TaS-2 has better performance than the others. In fact TaS-1, TaS-2, and TC-2 produce the desired output rate for $C = 30{,}000$ and $C = 20{,}000$. But, TaS-2 outperforms them for $C = 15{,}000$. For $C = 10{,}000$, none of them makes the desired rate. Please note that the output rate of the first video frame is an exception in this case because its size is too large to be reduced. However, it can be handled easily by work-ahead transmission.

5.2.2 Number of Transcoded Frames

Table 3 depicts the number of transcoded frames in case of conventional transcoding (TC-1 and TC-2) and TaS for $b = 1$ MB, $w = 30$ frametime, and $\alpha = 10$ frametime. The number of transcoded frames for each frame type is also presented. The table shows that TaS-1 transcodes less number of frames compared with TaS-2, TC-1, and TC-2. In TaS-2, the number of transcoded frames is higher than in TaS-1 and it even can be higher than in TC-2 because of the transcoding order that TaS-2 takes. In TC-1, all of frames were transcoded because the approach taken by ffmpeg. That transcoder takes the threshold rate to be in bytes per second, instead of in bytes per frame as taken by the other methods (TaS-1, TaS-2, and TC-2). From these results, TaS-1 performs better in the number of transcoded frames.

5.2.3 Quality Distortion

Distortion is measured by using mean square error (MSE) between decoded original MPEG video data and decoded transmitted MPEG video data. Figure 5 shows the results on MSE for TaS-1, TaS-2, and conventional transcoding in case of $C = 20{,}000$. The average values of MSE for $b = 1$ MB,

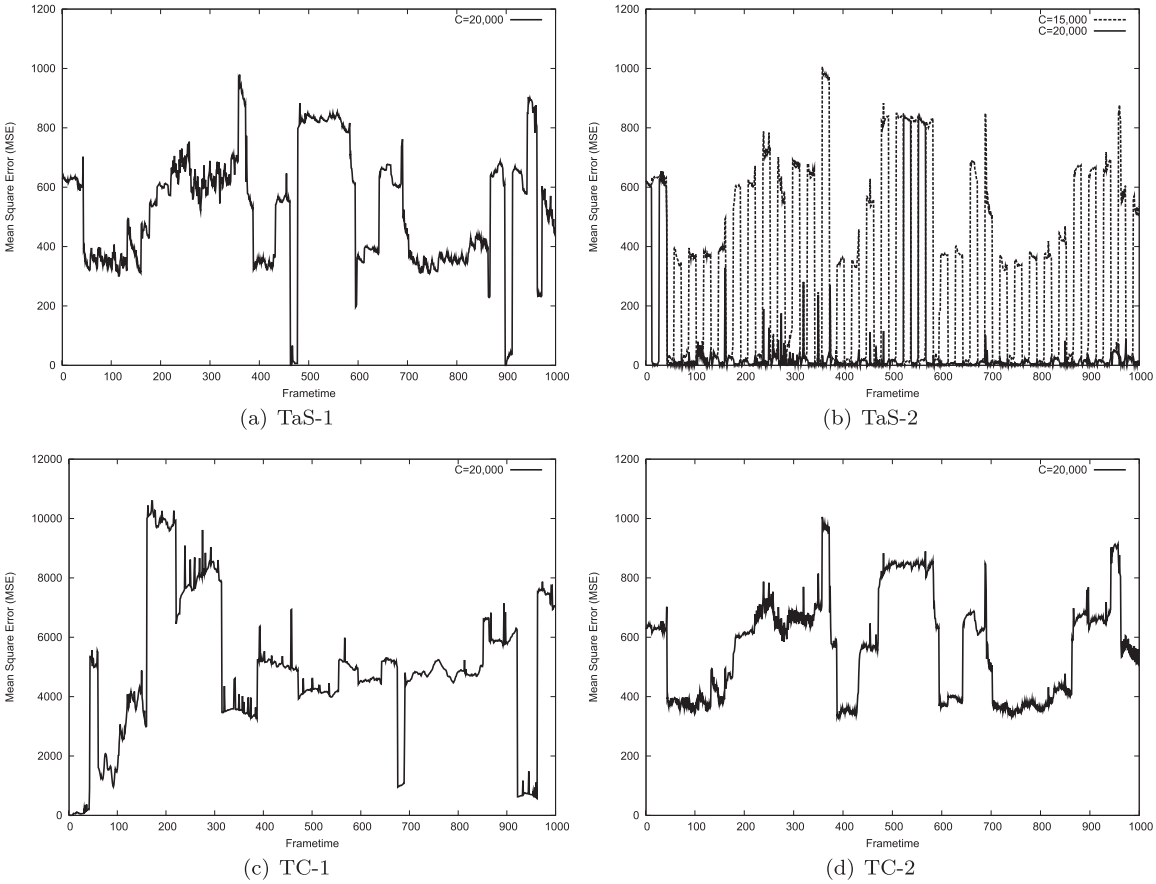


Fig. 5 Mean Square Error (MSE) for C = 20,000, b = 1 MB, w = 30 frametime, and α = 10 frametime.

Table 4 Average MSE for b = 1 M, w = 30, and α = 10.

C	TaS-1	TaS-2	TC-1	TC-2
10,000	554.82	575.38	4,857.13	575.34
15,000	554.25	308.37	4,861.86	574.70
20,000	525.59	55.15	4,864.33	569.89
30,000	7.88	7.88	4,866.52	555.38

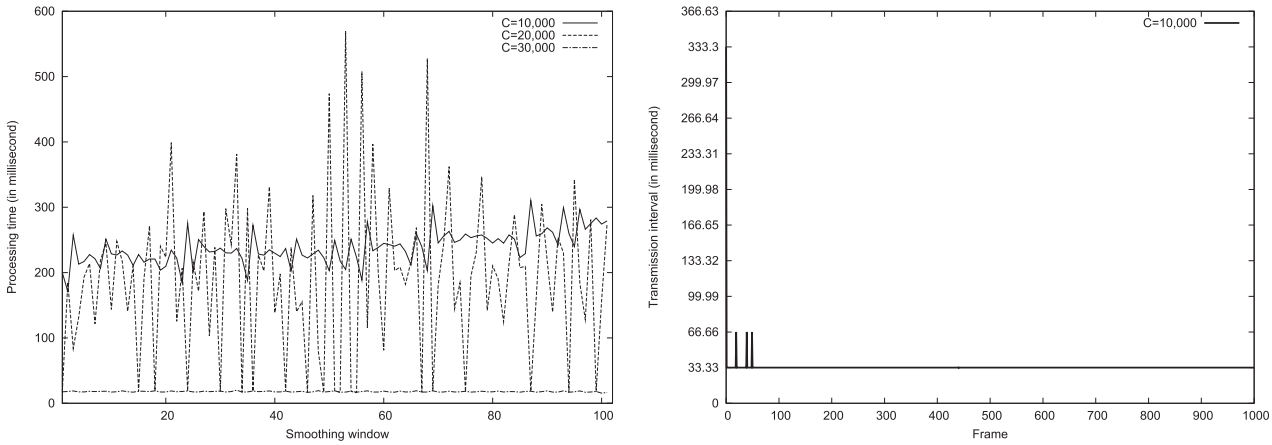
w = 30 frametimes, and α = 10 frametimes are also depicted on Table 4. Because there are no significant differences in distortion among C = 10,000, C = 15,000, and C = 20,000, only one specific case is shown on the figure, which is the case for C = 20,000. One exception exists on Fig. 5 (b) where there are a significant difference between C = 15,000 and C = 20,000. The results for C = 30,000 is not drawn in the figure because only the first frame is transcoded. Hence the MSE for C = 30,000 is always zero except for the first GOP. The size of the first frame of that first GOP is too large so that it has to be transcoded.

The results on TaS-1 (Fig. 5 (a)) and TaS-2 (Fig. 5 (b)) were acquired using the same experiment setting as in the previous subsection, where b = 1 MB, w = 30 frametimes, and α = 10 frametimes. The figure shows that TC-1 yields greater distortion than the others because it modifies the content of all of the frames. The largest value of MSE in transcoding is almost ten times the largest value of MSE in either TaS-1 or TaS-2. Meanwhile, TaS-2 in average has lower distortion than TaS-1 even though the number of transcoded frames in TaS-2 are greater than in TaS-1. The difference between TaS-1 and TaS-2 is significant for C = 20,000. Even though TaS-1 has less number of transcoded frames, the result on MSE shows that TaS-2 performs better than TaS-1. It also outperforms both TC-1 and TC-2.

However, please note that the distortion values shown above has a meaning of the difference of pixel values between the original video data and the reconstructed video data after processing. Larger difference does not always mean worse quality, but it is better to make the difference to be as small as possible. The quality of the reconstructed video still has to be evaluated by using subjective measurement in Sect. 5.2.5.

5.2.4 Processing Time

We investigate the processing time of the slowest system, which is TaS-2. Figure 6 (a) shows the time needed to perform TaS-2 algorithm within a smoothing window when the buffer size is 1 MB, window size is 30 frametime, and sliding window size is 10 frametime. The figure displays the processing time for C = 10,000 (solid line), C = 20,000



(a) Processing time (b) Transmission interval between two successive frames

Fig. 6 Processing time and transmission interval between two successive frames for b = 1 MB, w = 30 frametime, and α = 10 frametime.

(dashed line), and C = 30,000 bytes (dot-dashed line). The graph for C = 15,000 has similar pattern with the graph for C = 20,000 so that it is not shown in the Figure.

From Fig. 6 (a), the processing time for C = 10,000 bytes takes place between 171.2 to 530.36 ms (or 241.17 ms in average) per smoothing window. At C = 30,000 bytes, it only takes about 15.55 to 41.6 ms (or 18.1 ms in average) to perform TaS-2 algorithm within the smoothing window due to less recalculation performed. At C = 20,000, sometimes the system requires more recalculation than in case of C = 10,000. However, in average the processing time of the C = 20,000 case, which is 190.71 ms, is still lower than the processing time of the C = 20,000 case.

Figure 6 (b) shows the transmission time interval between two consecutive frames at video sender. This interval represents how long the sender should wait before sending out the next frame. Ideally, the time interval should be about 33.33 ms for 30 frame-per-second video data. The Figure shows that mostly the time interval can be kept at 33.33 ms although it goes beyond 33.33 ms at some occasions.

The Figure also show that the sender must wait 333.3 ms or about 10 times of the ideal value before sending out the first frame. This always occur because the sender must wait for the TaS algorithm within a smoothing window to finish.

Although the processing time (Fig. 6 (a)) may take hundreds of millisecond each time, video transmission can be performed as scheduled where the transmission time interval can be made 33.33 ms at most of the time as shown on Fig. 6 (b). This can happen because transmission process and TaS algorithm run in parallel. The video frames that come from TaS are saved in the sender buffer first prior to transmission. Meanwhile, at the same time, when data exist the sender sends out video data from the buffer every 33.33 ms.

Table 5 Mean Opinion Score (MOS) for C = 20,000, b = 1 M, w = 30, and α = 10.

TaS-1	TaS-2	TC-1	TC-2
1.44	3.44	4.78	1.22

5.2.5 Subjective Evaluation

Subjective evaluation is conducted by using Mean Opinion Score (MOS). Nine audiences were asked to watch one original video and four reconstructed videos, where each reconstructed video is a video received after using TaS-1, TaS-2, TC-1, and TC-2, respectively. They have to give a grade which ranges from 1 to 5. The grade of 5 means that the reconstructed video is difficult to distinguish from the original video. Meanwhile, the grade of less than 5 means that the quality of the reconstructed video is worse than the quality of the original video where the grade of 1 indicates the worst one. Table 5 shows the measurement result.

The table shows that the most of audiences gave the output TC-1 with the highest grade. The second preference is TaS-2. This is an expected result since the reconstructed video of TaS-2 sometimes still show some artifacts due to truncation of high frequency DCT coefficients when a frame must be transcoded. Meanwhile, the reconstructed video of TC-1 is visibly almost the same as the original video because TC-1 performs full decoding and re-encoding without truncation as described on Section 5.1. The table also shows that the output of TaS-2 is preferred than the output of TaS-1 since more data are dropped out from TaS-1.

6. Conclusion

A system to adjust the transmission bit rate of VBR MPEG video data called Transcoding-after-Smoothing (TaS) is proposed by using a combination of transcoding and bit rate smoothing algorithm in order to keep the transmission bit rate below a certain threshold value. Two kinds of TaS meth-

ods are proposed. They are TaS without preference (TaS-1) and TaS using frame type preference (TaS-2). Experiment results show that TaS-1 yields significant reduction in the number of transcoded frames compared with TaS-2 and conventional frame-level transcoding. However, TaS-2 performs better in minimizing the distortion.

On the other side, based on subjective measurement, the reconstructed video of TaS-2 seems to have lower preference than TC-1, which is a transcoding system using ffmpeg. This result is caused by the transcoding technique used in TaS experiment, which left only the first DCT coefficient in each transform block.

Some works remain to be done in the future, like transporting high definition MPEG video data where the amount of data to be transmitted is much larger than the standard definition video used in this paper. This may require faster algorithm to handle such amount of data.

Acknowledgments

Authors thank Prof. Hiroyuki Kasai of The University of Electro-Communications for his comments and suggestions to improve the quality of this paper.

References

[1] http://www.videolan.org
[2] http://www.apple.com/quicktime
[3] http://www.realnetworks.com
[4] H. Sun, X. Chen, and T. Chiang, Digital Video Transcoding for Transmission and Storage, CRC Press, 2005.
[5] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," IEEE Trans. Circuits Syst. Video Technol., vol.17, no.9, pp.1103–1120, Sept. 2007.
[6] V.K. Goyal, "Multiple description coding: Compression meets the network," IEEE Signal Process. Mag., pp.74–93, Sept. 2001.
[7] J. Xin, C-W. Lin, and M-T. Sun, "Digital video transcoding," Proc. IEEE, vol.93, no.1, pp.84–97, Jan. 2005.
[8] ISO/IEC JTC1 IS 13818 (MPEG-2), "Generic coding of moving pictures and associated audio," 1994.
[9] ISO/IEC JTC1 IS 14386 (MPEG-4), "Generic coding of moving pictures and associated audio," 2000.
[10] ISO/IEC 14496-10, "Information technology — Coding of audio-visual objects — Part 10: Advanced video coding," 2005.
[11] V. Padmanabhan, H. Wang, and P. Chou, "Resilient peer-to-peer streaming," Proc. IEEE Int. Conf. on Network Protocols, pp.16–27, Nov. 2003.
[12] S.B. Wicker, Error Control Systems for Digital Communication and Storage, Prentice Hall, 1995.
[13] A. Albanese, et. al., "Priority encoding transmission," IEEE Trans. Inf. Theory, vol.42, no.6, pp.1737–1744, Nov. 1996.
[14] G. Davis and J. Danskin, "Joint source and channel coding for image transmission over lossy packet networks," SPIE Conf. Wavelet Applications to Digital Image Processing, pp.376–387, Aug. 1996.
[15] I. Ahmad, X. Wei, Y. Sun, and Y-Q. Zhang, "Video transcoding: An overview of various techniques and research issues," IEEE Trans. Multimedia, vol.7, no.5, pp.793–804, Oct. 2005.
[16] S. Sen, J.L. Rexford, J.K. Dey, J.F. Kurose, and D.F. Towsley, "Online smoothing of variable-bit rate streaming video," IEEE Trans. Multimedia, vol.2, no.1, pp.37–48, March 2000.
[17] D. Ye, Q. Wu, and Z. Zhang, "A control-theoretical approach to adaptive internet video streaming," IEICE Trans. Commun., vol.E86-B, no.2, pp.585–594, Feb. 2003.
[18] H. Kasai, H. Tominaga, T. Hanamura, and W. Kameyama, "Rate control scheme for low delay MPEG-2 video transcoder," IEICE Trans. Commun. (Japanese Edition), vol.J83-B, no.2, pp.151–164, Feb. 2000.
[19] L. Yuan, H. Sun, and W. Gao, "MPEG transcoding for DVD recording," Proc. 2003 Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia, vol.1, pp.545–548, Dec. 2003.
[20] H. Xiong, C. Luo, S. Yu, and J. Sun, "Architecture and rate control for network TV broadcasting on application-oriented QoS," Proc. Ninth International Conference on Communication Systems 2004, pp.40–44, Sept. 2004.
[21] ITU-T Recommendation H.262 — ISO/IEC 13818-2, "Information technology – Generic coding of moving pictures and associated audio information: Video," 1995.
[22] J-W. Lin, R-I. Chang, J-M. Ho, and F. Lai, "FOS: A funnel-based approach for optimal online traffic smoothing of live video," IEEE Trans. Multimedia, vol.8, no.5, pp.996–1004, Oct. 2006.
[23] J.D. Salehi, Z-L. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing," IEEE/ACM Trans. Netw., vol.6, no.4, pp.397–410, Aug. 1998.
[24] D. Sun, M. Nishiara, and H. Morita, "On multiple smoothed transmission of MPEG video streams," IEICE Trans. Fundamentals, vol.E88-A, no.10, pp.2844–2851, Oct. 2005.
[25] I G.B.B. Nugraha, M. Nishiara, and H. Morita, "A prototype of Internet video broadcasting using unicast protocol," IEICE Technical Report, NS2005-55, June 2005.
[26] I G.B.B. Nugraha, S. Marugami, M. Nishiara, and H. Morita, "Multicast communication for broadcasting service over IPv4 network using IP option," IEICE Trans. Commun., vol.E89-B, no.5, pp.1570–1580, May 2006.
[27] IETF RFC 793, "Transmission control protocol," Sept. 1981.
[28] IETF RFC 768, "User datagram protocol," Aug. 1980.
[29] H. Sun, W. Kwok, and J.W. Zdepski, "Architectures for MPEG compressed bitstream scaling," IEEE Trans. Circuit Syst. Video Technol., vol.6, no.2, pp.191–199, April 1996.
[30] P.A.A. Assuncao, and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit rate reduction of MPEG-2 bit streams," IEEE Trans. Circuits Syst. Video Technol., vol.8, no.8, pp.953–967, Dec. 1998.
[31] Z. He and S.K. Mitra, "A unified rate-distortion analysis framework for transform coding," IEEE Trans. Circuits Syst. Video Technol., vol.11, no.12, pp.1221–1236, Dec. 2001.
[32] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architecture and techniques: An overview," IEEE Signal Process. Mag., vol.20, no.2, pp.18–29, March 2003.
[33] A. Eleftheriadis and P. Batra, "Optimal data partitioning of MPEG-2 coded video," IEEE Trans. Circuit Syst. Video Technol., vol.14, no.10, pp.1195–1209, Oct. 2004.
[34] A. Eleftheriadis and P. Batra, "Dynamic rate shaping of compressed digital video," IEEE Trans. Multimedia, vol.8, no.2, pp.297–314, April 2006.
[35] P.A.A. Assuncao and M. Ghanbari, "Buffer analysis and control in CBR video transcoding," IEEE Trans. Circuits Syst. Video Technol., vol.10, no.1, pp.83–92, Feb. 2000.
[36] I.G.B.B. Nugraha and H. Morita, "MPEG video bit-rate shaping technique using smooth-transcoding algorithm," Proc. 16th International Conference on Computer Communications and Networks 2007, pp.821–825, Aug. 2007.
[37] I.G.B.B. Nugraha, H. Kasai, and H. Morita, "Transcoding-after-smoothing (TaS) method for VBR video bit-rate adaptation," Proc. 30th Symposium on Information Theory and its Applications 2007, pp.853–858, Nov. 2007.

I Gusti Bagus Baskara Nugraha received the B.Eng. and M.Eng. degrees in electrical engineering from Institut Teknologi Bandung, Indonesia, in 1999 and 2001, respectively. He received the Doctor of Engineering degree from the University of Electro-Communications, Tokyo, Japan in 2006. He is currently with the University of Electro-Communications. His research interests are in digital video communication system.

Hiroyoshi Morita received the B.Eng. degree, the M.Eng. degree, and D.Eng. degree from Osaka University, in 1978, 1980 and 1983, respectively. In 1983, he joined Toyohashi University of Technology, Aichi, Japan as a Research Associate in the School of Production System Engineering. In 1990, he joined the University of Electro-Communications, Tokyo, Japan, first as an Assistant Professor at the Department of Computer Science and Information Mathematics, where from 1992, he was an Associate Professor. Since 1995 he has been with the Graduate School of Information Systems. Since 2005, he has been a Professor. He was Visiting Fellow at the Institute of Experimental Mathematics, University of Essen, Essen, Germany during 1993–1994. His research interests are in combinatorial theory, information theory, and coding theory, with applications to the digital communication systems.