Available online at www.sciencedirect.com

SCIENCE DIRECT®

J. Vis. Commun. Image R. 16 (2005) 512–526

JOURNAL OF VISUAL Communication & IMAGE Representation

www.elsevier.com/locate/jvci

Real-time smoothing for network adaptive video streaming

Kui Gao [a,b,*], Wen Gao [a,b], Simin He [a], Yuan Zhang [b,c]

[a] *Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China*
[b] *Graduate School, Chinese Academy of Sciences, Beijing 100039, China*
[c] *Beijing Broadcasting Institute, Beijing 100024, China*

Received 15 March 2004; accepted 3 December 2004
Available online 9 April 2005

Abstract

Real-time streaming delivery over the Internet with bandwidth variation is a very challenging task. It is important to smooth the quality variability and improve the utilization of the available network bandwidth. In this paper, we propose a real-time optimal smoothing scheduling algorithm for network adaptive video streaming with the variable network bandwidth and packet loss. The algorithm adopts a rate-distortion optimized framework and real-time scheduling scheme to select and schedule the packets according to the network status. It attempts to minimize the quality variability at the client end while at the same time maximizing the utilization of the variable network bandwidth. Experiments show that, compared with frame-based scheduling algorithm, our proposed real-time smoothing algorithm improves and smoothes the quality in decoded video frames.
© 2005 Elsevier Inc. All rights reserved.

Keywords: Real-time scheduling; Scalable streaming; Fine granularity scalable; Smoothing; Network adaptive

* Corresponding author. Fax: +86 10 58858301.
E-mail addresses: kgao@jdl.ac.cn (K. Gao), wgao@jdl.ac.cn (W. Gao), smhe@jdl.ac.cn (S. He), yzhang@jdl.ac.cn (Y. Zhang).

1047-3203/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2004.12.001

1. Introduction

With the advances of network technology and communication technology, real-time audio and video streaming services are becoming increasingly popular over the Internet. Unlike digital broadcasting systems, where bandwidth and channel characteristics are known, the current Internet only provides best-effort services and does not provide Quality of Service (QoS) guarantee for real-time streaming media services. Network conditions such as available network bandwidth, packet loss ratio, delay and delay jitter vary from time to time. Therefore, it is very desirable to develop scalable media coding and network-adaptive streaming technologies that can adapt to various conditions of heterogeneous network. The constraints on resources (such as available network bandwidth and buffer size) are harmful for real-time streaming applications, which require a tight end-to-end delay constraint and a low data loss rate for reasonably good playback quality. These resources could be limited in such a way that it may not be possible to deliver full-quality video. In such a situation, it is desirable to minimize the degradation in the video quality while operating within the resource constraints. The growing need for an efficient network-adaptive streaming system over the Internet motivated the development of MPEG-4 FGS (Fine Granularity Scalability) framework [12,13,22], which was established in the Amendment on Streaming Video Profile of the MPEG-4 standard [17]. The FGS framework accommodates a wide range of data rate variability by distributing enhancement layers over a wide range of bit rates and provides an efficient coding based on bit-plane coding which is more efficient than run-level coding. It differs from all of previous layered video coding schemes where only limited layers are available and any desired bits at the enhancement sub-streams can be adjusted at transmission time with a very fine granularity and very little complexity. However, only the encoder/decoder structure is defined in the standard without specifying how to perform optimal truncation. It is not a trivial task to select data in a streaming session such that better but smoothing quality playback is ensured when the network conditions are constantly varying. Without smoothing, the playback quality would be variable along with network bandwidth fluctuation at the client end, which is annoying to human ears and eyes. It is important to smooth the quality variability and improve the utilization of the variable network bandwidth in a streaming session from a scalable streaming server to the client over the network with bandwidth variation.

Many research results exist on smoothing. Salehi et al. [23] proposed a work-ahead smoothing technique which achieves the greatest possible reduction in rate variability when transmitting stored video from a server to a client across a network. Grossglauser et al. [9] argued that a renegotiated service best addresses the presence of burstness over multiple time-scales and design the Renegotiated Constant Bit Rate (RCBR) service for carrying compressed video traffic. However, rate smoothing cannot be used for VBR video over the best-effort network with bandwidth variation. Layered video multicast has been discussed to accommodate the heterogeneity of receivers [10,14]. Due to the delivery deadline constraint and the limitation of the available network, not all layers can be transmitted. The number of layers transmitted is dynamically varying according to the available network bandwidth. The streaming server will send the data to client through network as many layers as possible when the available bandwidth is large, and will truncate some layers when the available bandwidth is small. However, frequent adding and dropping of layers can incur significant quality variability which is annoying to human eyes. Nelakuditi et al. [19] describe an adaptive algorithm for providing smoothed layered video delivery by utilizing the client buffer for prefetching. However, the algorithms assume the video is CBR with linearly spaced layers; hence they do not maximize perceptual quality of fine grained scalable video. The optimized scheduling of layered streaming media delivery over the best-effort network was first proposed by Podolsky et al. [21], who adopt the Markov chain to analyze and find the optimal packets transmission and retransmission policies. In their model, the bandwidth between server and client is constant, but packets are independently lost with a constant probability. The same problem was addressed with a rate-distortion analysis [3,16]. Saparilla and Ross [24] model the available bandwidth as a stochastic process and propose an optimal bandwidth allocation scheme among base and enhancement layers. Kim and Ammar [11] developed an optimal adaptation scheme and an online heuristic based on whether the network conditions are known a priori. Gao et al. [6,7] proposed real-time scheduling algorithm but did not take into according smoothing scheme.

In this paper, we proposed a dynamic real-time smoothing algorithm to minimize the quality variability at the client end while at the same time maximizing the utilization of the variable network bandwidth. We adopt a rate-distortion optimized framework and real-time scheduling scheme to select and schedule the packet to the client according to the network bandwidth. Given a set of packets that are the candidates to be transmitted by the server, we define a schedule as the transmission order of all those candidate packets, which specifies whether and when a packet should be transmitted. Clearly, the order of delivering the packets has an impact on the actual playback quality, because of the delay constraint and data dependencies. The importance of a packet needs to take into account not only the playback distortion and its deadline, but also the network bandwidth. We select the most important (i.e., the largest distortion) packet that must satisfy the real-time constraint. The smoothing algorithm combines distortion of packet and real-time scheduling scheme. Thus, important layer data can be transmitted earlier and the playback quality will be more smoothing at the client end with the bandwidth variation. The scheme not only smoothes the playback quality but also improves the utility of the bandwidth.

The rest of this paper is organized as follows. Section 2 briefly introduces the framework of scalable streaming system. Section 3 presents the real-time smoothing scheduling algorithm for the scalable streaming media. Section 4 gives some experimental results and comparisons among different algorithms. Section 5 concludes this paper.

2. The architecture of network-adaptive streaming system

2.1. The overview of MPEG-4 FGS

Fine granularity scalability is provided by the MPEG-4 FGS coding scheme [12,13], which has become a part of the MPEG-4 as an amendment to the traditional non-scalable MC-DCT approach for streaming video profile [17]. It consists of one base layer that is coding with an MPEG-4 compliant non-scalable coder, as well as one or more enhancement layers coded progressively with the embedded DCT coding scheme. Fine granularity is achieved from the base layer bitstream upwards as each enhancement frame can be encoded independently with an arbitrary number of bits. The base layer is usually of low quality and is very thin to fit in typical small bandwidths. The enhancement layers are created by DCT and entropy coding of the residual image of the base layer with the efficient coding based on bit-plane coding technology. Bit-plane coding yields an embedded bitstream and achieves the desired fine granularity scalability. Compared with the non-scalable video codec where the rate adaptation are performed in the form of transcoding, the MPEG-4 FGS codec simplifies the rate adaptation task a lot since the bitstream in enhancement layers can be arbitrarily truncated. The FGS framework strikes a good balance between coding efficiency and scalability while maintaining a very flexible and simple video-coding structure [22]. Therefore, the FGS streaming can adapt a wide range of data rate variability by distributing enhancement layers over a wide range of bit rates and the decoder can use all of the truncated bitstream to increase video quality at the client end. In MPEG-4 FGS coding scheme, the base layer and all enhancement layers in predicted frame are always predicted from the reconstructed version of the base layer in the reference frame. Fig. 1 shows conceptually such a framework.

2.2. The architecture of network-adaptive streaming system

Typically, the architecture of a streaming system is shown Fig. 2. The system consists of three main components: a real-time streaming server, a corresponding real-

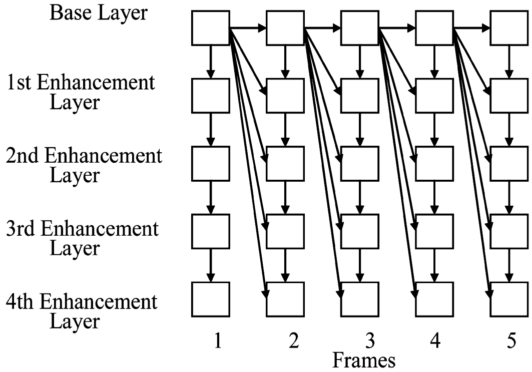

Fig. 1. The FGS framework.

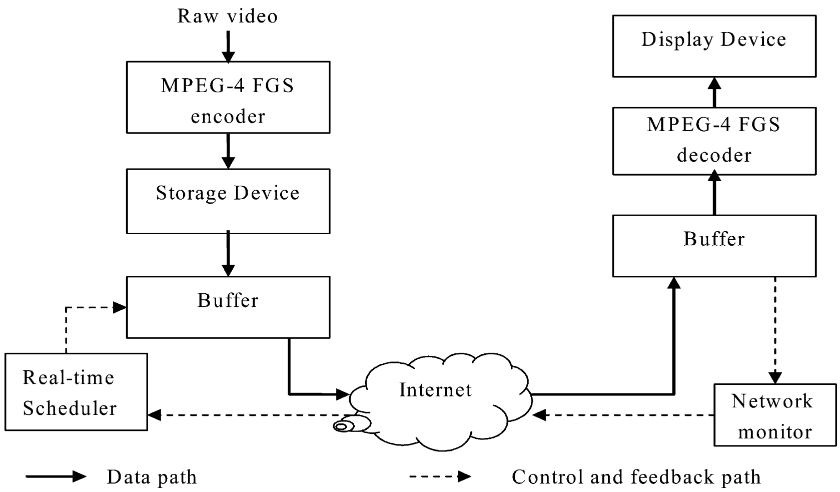

Fig. 2. The scalable video streaming system.

time streaming client and a network which also serve for transmission of media data. The client requests are sent to the server via network connections. The buffers at the client end are used to provide some tolerance on variations in network delay as well as data consumption rates. The streaming media sequence consists of many frames, which are compressed into one base layer and one or more enhancement layers. The bitstreams of base layer and enhancement layers are packetized and fed into the server's transmission buffers. The packets in transmission buffers are ready to be scheduled for transmission. The scheduler in the server controls the packet size and sequence, manages the server transmitted buffer and packets via the network to the clients' buffers. When a playback requirement is accepted by the server, the scheduler determines the layers that should be sent to the client according to the available network bandwidth and scheduling scheme. The Internet only provides best-effort services and the streaming packets over best-effort the networks is complicated by a number of factors including unknown and time-varying bandwidth, delay, and losses. There are two schemes of recover lost packets over a lossy network: the transmission of redundant forward error correction (FEC) packets [25] and retransmission of lost packets that arrive at the receiver before their decoding deadlines [26]. Retransmission-based error recovery requires at least one additional round-trip time (RTT) delay to recover lost packets. Despite its latency drawback, retransmission is still an attractive solution because of its modest bandwidth and processing costs. In this paper, we adopt retransmission scheme to recover lost packet. There are also packets in the buffer that are waiting for retransmission at the server end because the client reports them loss. The scheduler selects one packet at a time from those buffers and sends it to the lossy channel. Some packets may be lost, damaged or delayed (delayed packets are also considered lost if they exceed their playback delay). At the client end, the lost or damaged packets are reported to the server via a feedback channel. The server will determine the sending rate according to the feedback information by TCP-Friendly Rate Control (TFRC) protocol [5]. TFRC is a congestion control mechanism for unicast flows operating in a best-effort Internet environment. It is reasonably fair when competing for bandwidth with TCP flows, but has a much lower variation of throughput over time compared with TCP, making it more suitable for streaming media applications where a relatively smooth sending rate is of importance.

3. Real-time smoothing for network-adaptive streaming

Scalable streaming media have timing constraints because of their sensitivity to delay and jitter. It is important to select and schedule packet delivery of scalable streaming media over a lossy network. Retransmission can be used to recover the lost packets over a best-effort network [26]. In general, the higher the amount of detail in the played video, the better is its quality. However, it is generally agreed that it is visually more pleasing to watch a video with consistent, albeit lower, quality than one with highly varying quality. Due to the delivery deadline constraint and the limitation of the available network, not all lost packets can be recovered by retransmission. However, if the server schedules a packet to be sent much earlier than its playback time, this packet will have more chances to be retransmitted before it is too late for display. If a packet is not available at its expected display time at the receiver, it will miss its deadline. In addition, even if there is still time to retransmit a packet at a given time, a decision needs to be made on whether it should be retransmitted or not. Initially, we exclude interactive actions such as pause/resume and repositioning. With MPEG-4 FGS coding scheme, bit-plane coding in enhancement layers yields an embedded bitstream and achieves the desired fine granularity scalability. This radically discriminates FGS from other ordinary layered coding techniques. The dependency among bit-plane is very strong. A bit-plane is decodable only if all its ancestors, i.e., lower bit-planes that is depends on, are received and successfully decoded. In this section, we want to find a packet transmission policy to select the packets to be transmitted or retransmitted at any give time during a streaming session, in a way that the playback quality can be improved and smoothed.

Consider a set of n frames in a scalable media stream $F = \{F_0, F_1, \ldots, F_{n-1}\}$. To reduce the dependence among packets, the different bit-planes in the same frame are packetizated into different packets. The base layer and all enhancement layers in predicted frame are always predicted from the reconstructed version of the base layer in the reference frame as shown in Fig. 1. Let $p_{i,j} (0 \leqslant i \leqslant n-1, 0 \leqslant j \leqslant m)$ denote the packet of the jth layer in frame F_i. m is the number of layers including base layer and all the enhancement layers. When $j = 0$, that means $p_{i,j}$ is the base layer packet of frame F_i. When $j > 0$, $p_{i,j}$ is the enhancement layer packet. We assume that the packets in the same frame have the same release-time a_i and deadline d_i. The release-time a_i is the earliest time at which the packet $p_{i,j}$ become ready for scheduling in the transmission buffer. Deadline d_i is the latest time at which the packets of frame F_i should be sent to the client, otherwise it is too late for playback. The schedule-time

$s_{i,j}$ is the time at which the scheduler sends packet $p_{i,j}$ to client. The $RTT(t)$ (round-trip-time) is defined as the interval from the time a packet is sent from the server to the time the server gets feedback of this packet from the client at time t. $RTO(t)$ is the retransmission timeout. Let $X(t)$ be the available network bandwidth at time t. The packet loss rate is $p_{\text{loss}}(t)$ at time t over the lossy network. Information related to $RTT(t)$ and packet loss rate $p_{\text{loss}}(t)$ can be received by server via client's feedback. The size of the packet $p_{i,j}$ is $b_{i,j}$. The processing time of the packet $p_{i,j}$ is $c_{i,j} = b_{i,j}/X(t)$. The fulfill-time of a packet $p_{i,j}$ is $f_{i,j} = s_{i,j} + c_{i,j}$. At time t the server knows the current available bandwidth $X(t)$ and its past values, but has no knowledge of its future values.

As mentioned above, the base layer carries the most important information with MPEG-4 FGS coding scheme. The base layer sub-stream is very sensitive to the channel errors. If the packets of the base layer are lost, the quality of reconstructed video can be degraded severely. However, the enhancement layers can tolerate the channel errors. When there are errors in the enhancement layer bitstream, a decoder can simply drop the rest of the enhancement bitstream of this frame. Generally, since the bit rate of the base layer is very low as compared with the enhancement layers, the priority-based scheduling scheme can be adopted for the base layer. In the rest of this section, we describe the constraint imposed by playback continuous for base layer first and then propose the real-time smoothing scheduling algorithm to the overall bitstream.

3.1. Constraints of playback continuous for base layer

Considering the characteristics of MPEG-4 FGS coding, it is well known that different portion of video bitstream has different importance to the quality of the reconstructed video. The delay constraint of real time streaming introduces limitation on the server retransmission capability [16]. Assuming the server gets feedback from the client, i.e., acknowledgments (ACKs) or negative acknowledgments (NAKs). The lost packet (indicated by a NAK), might not arrive to the receiver when the error rate is large. We set the release-time $a_{i,0}$ of packet $p_{i,0}$ according to the current network conditions (such as loss rate, RTT, etc.) before its deadline d_i to allow limited retransmission. MPEG-4 video stream can be decoded with acceptable video quality at error rate of 10^{-5} or lower [8]. For continuous playback of the scalable video stream, the server must ensure that the packet loss rate of base layer is not larger than 10^{-5} by retransmission. If the release time is too early before the deadline, the transmission buffer must be very large, it is not efficacious to the server for a large number of subsequent users. However, if the release time is too late, there is no chance to retransmit lost packets. Therefore, it is important to select the expected retransmission times K for lossy packets and set the release time a_i of the base layer to continuous playback. Let $\tau_{i,0}(K)$ denote the total loss probability of the packet $p_{i,0}$ through K times retransmission. We can get the smallest integer K when $\tau_{i,0}(K) \leqslant 10^{-5}$. We have

$$a_i = d_i - K * RTO, \tag{1}$$

where a_i is the earliest release time of packet $p_{i,0}$. Therefore, the server can read $p_{i,0}$ from the storage device and put it into the transmission buffer at the time a_i according to the $RTT(t)$ and lost rate of the network dynamically. We will adopt priority-based scheduling scheme as described in the follow and there is more retransmission chance for the lossy packet of the base layer.

3.2. Real-time smoothing scheduling algorithm

As for the enhancement layers, the residue between the original image and the reconstructed image of base layer is compressed with bit-plane coding technique to form the enhancement bitstream. Since bit-plane coding produces an embedded bitstream with fine granularity scalability, the enhancement bitstream can be arbitrarily truncated to fit the available channel bandwidth and tolerate the channel errors. Real-time streaming playback requires the server to transmit the data packets prior to their playback instants (i.e., deadlines). Packets scheduling algorithms with real-time deadlines can optimize the quality of the video reconstructed at the client end. Traditional real-time scheduling scheme [1] cannot be used directly to adapt to the features of the fine scalable video stream. Chou and Miao [3] proposed a transmission scheme to minimize the end-to-end distortion in a rate-distortion optimized way. An interactive descent algorithm is used to minimize the average end-to-end distortion. However, the high computational complexity of this approach makes it less appealing during real-time streaming where the server must adapt to bandwidth variations very quickly. Moreover, it doesn't take into account smoothing quality.

We now propose a real-time smoothing scheduling algorithm to select packets for transmission at any given time. It is assumed that packets have the same release time in a frame. The release time a_i of packet $p_{i,j}$ can be set according to Eq. (1), i.e., packet $p_{i,j}$ is put into transmission buffer and becomes ready for scheduling at a_i. A packet $p_{i,j}$ can be selected and scheduled if the following conditions are satisfied: the current time $t_{\mathrm{cur}}(t_{\mathrm{cur}} \in [t, t+T))$ is later than its release-time a_i, and its fulfill-time $f_{i,j}$ is earlier than its deadline, i.e., $a_i \leqslant t_{\mathrm{cur}}$ and $t_{\mathrm{cur}} + c_{i,j} \leqslant d_i$. We adopt a rate-distortion optimized framework and real-time scheduling scheme to select and schedule the packets of enhancement layer. Dai et al. [4] analyzed the rate distortion optimal modeling of MPEG-4 FGS. The enhancement layers of MPEG-4 FGS use bit-plane coding, which considers each input value as a binary number instead of a decimal integer. Assume that the quantization step applied to a given frame is Q, which depends on the bit-plane number where the server stopped transmission of the FGS layer. If the maximum number of bit-planes in a given frame is m and the last transmitted bit-plane is z (in the order form the most significant to the least significant), then $Q = 2^{m-z}$. A classical distortion model built for uniform quantizer (UQ) is often used for a variety of sources due to its simplicity

$$\Delta D(Q) = \frac{Q^2}{\beta}, \tag{2}$$

where $\beta = 12$. The dependency among bit-planes is very strong as illustrated in Fig. 1. A bit-plane is decodable only all its ancestors, i.e., lower bit-planes that is depends on, are received and successfully decoded. Cai et al. [2] define the performance metric of streaming FGS bit stream over lossy network as follows:

$$J = \sum_{p_{i,j} \in F} \Delta D(Q_{i,j}) * (1 - p_{\text{loss}}(p_{i,j})) * \prod_{p_{x,y} \prec p_{i,j}} (1 - p_{\text{loss}}(p_{x,y})) \tag{3}$$

and

$$\Delta D(p_{i,j}) = \Delta D(Q_{i,j}) * (1 - p_{\text{loss}}(p_{i,j})) * \prod_{p_{x,y} \prec p_{i,j}} (1 - p_{\text{loss}}(p_{x,y})). \tag{4}$$

$\Delta D(p_{i,j})$ is the expected contribution, i.e., the distortion reduction that is achieved if $p_{i,j}$ is successfully decoded. $p_{\text{loss}}(p_{i,j})$ and $p_{\text{loss}}(p_{x,y})$ is the packet loss probability of the packet that $p_{i,j}$ resides in and its ancestor packets $p_{x,y}$, respectively. And $p_{x,y}$ must be received to decode the received packet $p_{i,j}$. Note that in a bit-plane coded bit stream, the dependency is between bit-planes not between frames that are spatially tiled. Therefore, the server just only take account into the dependency between bit-planes in Eq. (3) and can roughly compute the corresponding distortion at each bit-plane from Eq (2). Obviously, the total quality reconstructed is the best at the client end when J is maximized.

In a real-time streaming system, it is impossible for a server to examine the entire media sequence when the server only has limited buffer. The available bandwidth and packet loss is unknown and time-varying. Therefore, it is difficult to compute J when a streaming starts to playback. The server can only make the decision based on the packets currently in the buffer. We can construct the possible model that a packet in transmission buffer is in one of two states: ready or blocked. When the packet is in ready state, it can be scheduled for transmission when the network is available. While the packet is in blocked state, it cannot be scheduled until get NAKs or its timeout is not larger than current time t_{cur}. The server selects a ready packet to transmit at t_{cur} from the transmission buffer. Let S_{buf} is the set of all the ready packets that are the candidates to be transmitted in the buffer, i.e., $S_{\text{buf}} = \{p_{i,j} | a_{i,j} \leqslant t_{\text{cur}}$ and $t_{\text{cur}} + c_{i,j} \leqslant d_i$ and in ready state$\}$. A real-time *background scheduling* [1] scheme to handle ready packets of enhancement layers in the presence of base layer packets is to schedule them in background. The server selects the ready packets of base layer with earliest deadline firstly (EDF) [15]. If there are no ready packets of base layer in the transmission buffer, the server will select and schedule the ready packets of enhancement layers. We select the ready packets of enhancement layers according to rate distortion optimal framework that we mentioned above. We know the lower (more important) layer packets have larger distortion from Eq (2). If the enhancement layer packets have same distortion, packets with earliest deadline are served first. Thus, base layer and important enhancement layers data can be transmitted/retransmitted earlier. If it is lost, it can have more chances to be retransmitted. The algorithm not only improves the utility of the bandwidth but also smoothes the playback quality. The precise description of real-time smoothing scheduling algorithm for delivery of scalable streaming media over a lossy network is given below.

Real-time smoothing scheduling algorithm:

Step 1: Put the packets according to release time a_i and set these packets in ready state. Get the current available network bandwidth $X(t_{\text{cur}})$ and $RTT(t_{\text{cur}})$. Compare the current time t_{cur} with the deadline d_i of the ready packets in the server transmit buffer. If $t_{\text{cur}} > d_i$, remove the packet from the server transmit buffer.

Step 2: If there are no ready packets of base layer in transmission buffer, go to Step 3. Select the ready packets of base layer by earliest deadline first and send it to the client via the network. Set the timeout of this packet $t_{i,j} = t_{\text{cur}} + \text{RTO}$ and set the packet in blocked state and go to Step 5.

Step 3: If the ancestor packet $p_{x,y}$ is in blocked state, $p_{\text{loss}}(p_{x,y})$ is the loss rate of network at the time that packet $p_{x,y}$ is sent; else if the ancestor packet $p_{x,y}$ has been received at the client end,

$p_{\text{loss}}(p_{x,y}) = 0;$

else $p_{\text{loss}}(p_{x,y}) = 1.$

Step 4: Computer $\Delta D(p_{i,j})$ and select the ready packet of enhancement layer according to $\max(\Delta D(p_{i,j}))$. If many ready packets have the same expected distortion, select them with earliest deadline first.

Step 5: If the server gets ACK of a blocked packet from the client, remove it from the buffer; else if a blocked packet reaches its timeout RTO, set it in ready state and set its release time as the current time t_{cur}. Go to Step 1.

Because of the variability and limitation of the available network bandwidth, some packets of enhancement layer and retransmission buffers are discarded if they miss their deadline. The algorithm transmits the most important packets to reconstruct the playback quality as soon as possible. Therefore, it not only improves the utility of the bandwidth but also smoothes the playback quality.

4. Simulation results

In the Internet environment, data are transmitted on a packet-by-packet basis. When delivered over the Internet, usually a packet is either received correctly or completely lost. These losses are mainly caused by network congestion and queuing delay. A two-state Markov model proposed by Gibert [27] is used to simulate packet losses over the lossy channel in our simulation. This model can characterize the error sequences generated by data transmission channels. In good state (G) errors occur with low probability while in bad state (B), they occur with high probability. The errors occur in cluster or bursts with relatively long error free intervals (gaps) between them. The state transitions are shown in Fig. 3 and summarized by the following transition probability matrix:

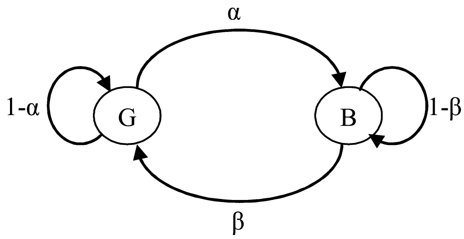

Fig. 3. Two-state Markov model for the network simulation.

$$P = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}.$$

α is the transition probability from good state to bad state and β is the transition probability from bad state to good state. The average packet loss rate is:

$$\varepsilon = \frac{\alpha}{\alpha+\beta}.$$

Details of the model can be found in [27]. It is assumed that the sending rate can be decided by TFRC protocol [5]. The receiver monitors the network condition and gathers related information, while the sender changes its sending rate according to the available network bandwidth estimated from the packet loss rate, round trip time, and retransmission timeout values. The protocol uses an equation-based way to estimate available bandwidth:

$$R_{\text{net}} = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{\text{RTO}}\left(3\sqrt{\frac{3p}{8}}\right)p(1+32p^2)}, \tag{5}$$

where s is the packet size, R is the round trip time, t_{RTO} is the retransmission timeout value, and p is the packet loss ratio. The sending rates in different RTTs and loss rates simulated by ns-2 [20].

The MPEG-4 FGS MoMuSys encoder/decoder [18] is used in the simulation. The base layer is encoded with MPEG-4, and the enhancement layer is encoded with FGS coding. Extensive simulations have been performed to test the performance of the proposed algorithms. The sequences Foreman, Coastguard and Akiyo in CIF format are used in the simulation. They are encoded with 30 frames per second and 300 frames are encoded and transmitted. For example, the maximum level of bit-plane is 7 in the sequence Foreman, so there are 7 enhancement layer. Different bit-plane has different size. The enhancement layer 0 (EL0) has the smallest size, yet it is most significant. The enhancement layer 6 (EL6) is the largest in size, yet it is the least significant. The average rate of base layer is 202.7912 Kbps and average rate of all enhancement layers is 18,868.4672 Kbps. The playback frame rate is 30 Hz. In our simulations, the average channel packet loss rate varies from 0.5 to 10%

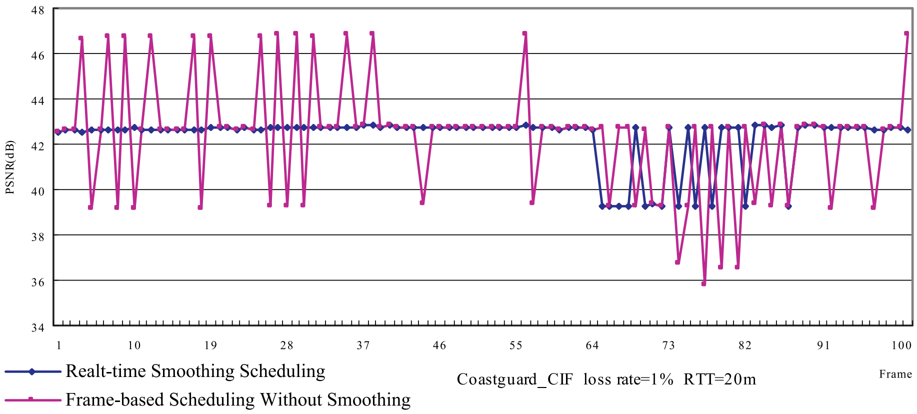

Fig. 4. PSNR comparisons of different scheduling algorithms with loss rate = 1% and RTT = 20 ms.

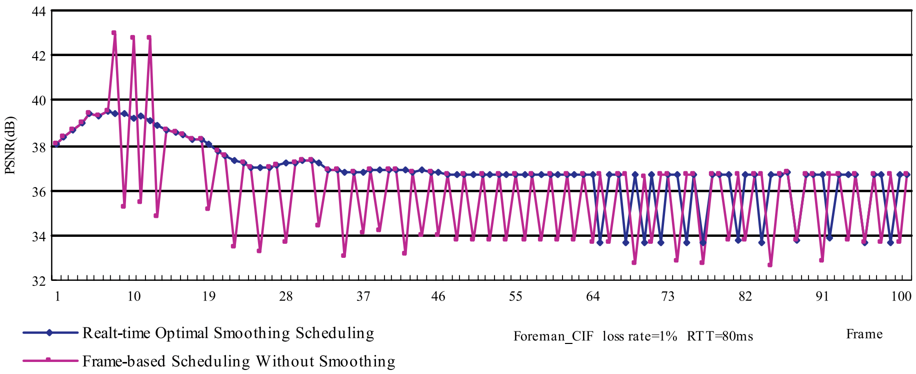

Fig. 5. PSNR comparisons of different scheduling algorithms with loss rate = 1% and RTT = 80 ms.

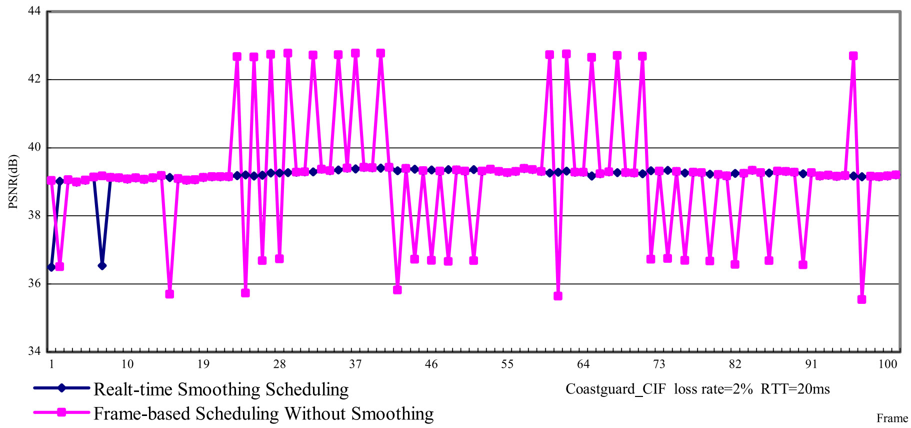

Fig. 6. PSNR comparisons of different scheduling algorithms with loss rate = 2% and RTT = 20 ms.

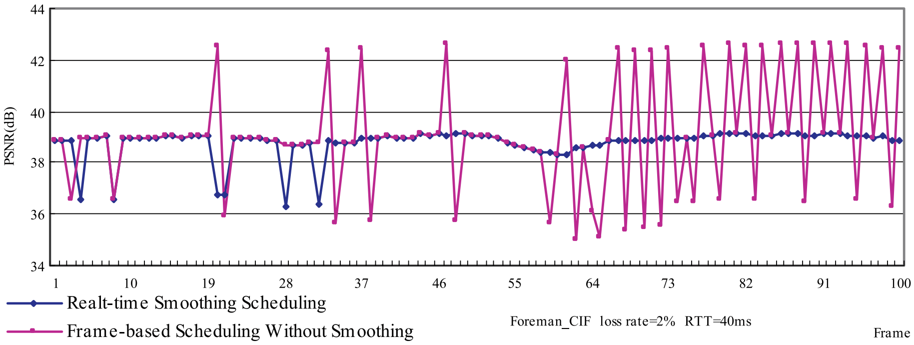

Fig. 7. PSNR comparisons of different scheduling algorithms with loss rate = 2% and RTT = 40 ms.

and the RTT varies from 20 to 160 ms. The playback quality is measured by Peak Signal-to-Noise Ratio (PSNR) of the video frames reconstructed at the client end based on all available packets.

Figs. 4–7 show comparisons of PSNR with real-time smoothing scheduling algorithm and frame-based scheduling algorithm without smoothing scheme by different sequences. It can be seen that, overall, real-time smoothing scheduling algorithm outperforms frame-based scheduling algorithm. Obviously, the real-time smoothing scheduling algorithm improves the utility of the bandwidth and smoothes the playback quality.

5. Conclusion

In this paper, we propose a real-time smoothing algorithm for network adaptive streaming media over the best-effort network. We adopt a real-time smoothing scheduling scheme under a rate-distortion optimized framework to select and schedule the packet to the client according to the network bandwidth. The simulation results show that the algorithm improves the utility of the bandwidth and smoothes the playback quality in a wide range of bandwidth scenarios. The studies of admission control, buffer management for scalable streaming server over the Internet are interesting topics for future work.

Acknowledgments

This work was supported by the National High Technology Development 863 Program of China under Grants No. 2001AA112100; the National Natural Science Foundation of China under Grant No. 69983008; the Knowledge Innovation Program of Chinese Academy of Sciences under Grant No. KGCXZ-103.

References

[1] G.C. Buttazzo, Hard Real-time Computing Systems: Predictable Scheduling Algorithms and Applications, Kluwer Academic Publishers, Dordrecht, 1997.

[2] H. Cai, G. Shen, Z. Xiong, S. Li, B. Zeng, An optimal packetization scheme for fine granularity scalable bitstream. Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS, 2002.

[3] P.A. Chou, Z. Miao, Rate-distortion optimized streaming of packetized media. IEEE Transactions on Multimedia, online available at: <http://research.microsoft.com/users/pachou/publications> February 2001 (submitted).

[4] M. Dai, D. Loguinov, H. Radha, Analysis and distortion modeling of MPEG-4 FGS. Proceedings of IEEE International Conference on Image Processing, ICIP, 2003.

[5] S. Floyd, M. Handley, J. Padhye, J. Widmer, Equation-based congestion control for unicast applications. Proceedings of ACM SIGCOMM, 2000.

[6] K. Gao, W. Gao, S. He, P. Gao, Y. Zhang, Real-time scheduling on scalable media stream delivery. Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS, 2003, pp. 824–827.

[7] K. Gao, W. Gao, S. He, Y. Zhang, Real-time scheduling and on-line resource allocation on scalable streaming media server. Proceedings of the SPIE Visual Communication and Image Processing, VCIP, 2003.

[8] S. Gringeri, R. Egorov, K. Shuaib, A. Lewis, B. Basch, Robust compression and transmission of MPEG-4 video. ACM Multimedia 1999 Electronic Proceedings, 1999.

[9] M. Grossglauser, S. Keshav, D.N.C. Tse, RCBR: a simple and efficient service for multiple time-scale traffic, IEEE/ACM Trans. Network. 5 (6) (1997) 741–755.

[10] T. Kim, M.H. Ammar, A comparison of layering and stream replication video multicast schemes. Proceedings of the Network and Operating System Support Digital Audio and Video, NOSSDAV, 2001.

[11] T. Kim, M.H. Ammar, Optimal quality adaptation for MPEG-4 Fine-Grained Scalable Video. Proceedings of the IEEE INFOCOM, 2003.

[12] W. Li, Fine granularity scalability in MPEG-4 for streaming Video. IEEE International Symposium on Circuit and System, Geneva, May 2000, pp. 299–302.

[13] W. Li, Overview of fine granularity scalability in MPEG-4 video standard, IEEE Trans. Circ. Syst. Video Technol. 11 (3) (2001) 301–317.

[14] X. Li, S. Paul, M.H. Ammar, Layered video multicast with retransmission (LVMR): evaluation of hierarchical rate control. Proceedings of INFOCOM, 1998.

[15] C.L. Liu, J.W. Layland, Scheduling algorithms for multiprogramming in a hard-real-time environment, J. Assoc. Comput. Mach. 20 (1) (1973) 46–61.

[16] Z. Miao, A. Ortega, Expected run-time distortion based scheduling for elivery of scalable media. Proceedings of the International Packet Video Workshop, 2002.

[17] MPEG-4, Coding of Audio-Visual Objects, Part-2 Visual, Amendment 4: Streaming Video Profile, ISO/IEC 14496-2/FPDAM4, July 2000.

[18] MPEG-4, ISO/IEC 14496-5:2001/FDAM1, ISO/IEC JTC1/SC29/WG11/N4711, Jeju Island, March 2002.

[19] S. Nelakuditi, R.R. Harinath, E. Kusmierek, Z.L. Zhang, Providing smoother quality layered video stream. Proceedings of Network and Operating System Support Digital Audio and Video, NOSSDAV, 2000.

[20] ns-2, ns-2 network simulator. Available from: <http://www.isi.edu/nsnam/ns>, 2002.

[21] M. Podolsky, S. McCanne, M. Vetterli, Soft ARQ for layered streaming media, J. VLSI Sig. Proc. Syst., Special issue on multimedia signal processing 27 (1–2) (2001) 81–97.

[22] H. Radha, M. van der Schaar, Y. Chen, The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP, IEEE Trans. Multimedia 11 (3) (2001) 51–68.

[23] J.D. Salehi, Z.-L. Zhang, J.F. Kurose, D. Towsley, Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing, IEEE/ACM Trans. Network. 6 (4) (1998) 397–410.

[24] D. Saparilla, K.W. Ross, Optimal streaming of video. Proceedings of the IEEE INFOCOM, 2000.
[25] Y. Wang, Q. Zhu, Error control and concealment for video communications: a review, Proceeding of the IEEE 86 (5) (1998) 974–997.
[26] D. Wu, Y.T. Hou, W. Zhu, Y.-Q. Zhang, Streaming video over the Internet: approaches and directions, IEEE Trans. Circ. Syst. Video Technol. 11 (3) (2001) 282–300.
[27] J.R. Yee, E.J. Weldon, Evaluation of the performance of the error-correcting codes on a Gilbert channel, IEEE Trans. Commun. 43 (8) (1995) 2316–2323.