Multimedia Tools and Applications 6, 47–60 (1998)

Dynamic Video Playout Smoothing Method for Multimedia Applications

MARIA C. YUANG, SHIH T. LIANG AND YU G. CHEN
Department of Computer Science and Information Engineering, National Chiao Tung University, Taiwan

Abstract. Multimedia applications including video data require the smoothing of video playout to prevent potential discontinuity. In this paper, we propose a dynamic video playout smoothing method, called the *Video Smoother*, which dynamically adopts various playout rates in an attempt to compensate for high delay variance of networks. Specifically, if the number of frames in the buffer exceeds a given threshold (TH), the Smoother employs a maximum playout rate. Otherwise, the Smoother uses proportionally reduced rates in an effort to eliminate playout pauses resulting from the emptiness of the playout buffer. To determine THs under various loads, we present an analytic model assuming the Interrupted Poisson Process (IPP) arrival. Based on the analytic results, we establish a paradigm of determining THs and playout rates for achieving different playout qualities under various loads of networks. Finally, to demonstrate the viability of the Video Smoother, we have implemented a prototyping system including a multimedia teleconferencing application and the Video Smoother performing as part of the transport layer. The prototyping results show that the Video Smoother achieves smooth playout incurring only unnoticeable delays.

Keywords: multimedia applications, video playout smoothing, interrupted Poisson process arrivals, transport layer

1. Introduction

Recent evolution in high-speed communication technology enables the development of multimedia applications combining a variety of media data, such as text, audio, graphics, images, and full-motion video. For supporting distributed multimedia applications, researchers have encountered various design problems. In particular, the smoothing of video playout has been considered essential to prevent potential playout discontinuity resulting from network delay variation while still achieving satisfactory playout throughput. As opposed to several existing approaches attempting to reduce delay variation from networks [5, 10], we tackle the problem from the end system perspective.

Several playout smoothing methods with various degrees of performance have been proposed. These methods fall into two main categories: *buffer-oriented* and *bufferless*. Buffer-oriented methods preserve playout continuity by buffering packets at the receiver [14, 15] or delaying the playout time of the first packet received [1–3, 6, 9, 10, 13, 14, 18, 19, 21]. These methods have been shown to be effective but lacking in taking dynamic changes of network delays into account. On the other hand, bufferless methods [12, 17] smooth playout through adjusting the source generation rate by means of feedback techniques. The weakness is that, since the generation rate of live sources is not adjustable, these methods can only be applied for stored-data applications.

In this paper, we propose a dynamic video playout smoothing method, called the *Video Smoother*. Generally, unlike existing methods described above, the Video Smoother dynamically adopts various playout rates according to the number of frames in the playout buffer in an attempt to compensate for high delay variance of networks. In particular, if the number of frames in the buffer exceeds a given threshold (TH), the Smoother employs a maximum playout rate. Otherwise, the Smoother uses proportionally reduced rates in an effort to eliminate playout pauses resulting from the emptiness of the playout buffer. To determine THs under various loads, we present an analytic model assuming Poisson and Interrupted Poisson Process (IPP) arrivals correspondent with networks with and without traffic shaper, respectively. Based on the analytic results, we establish a paradigm of determining THs and playout rates under various arrivals and loads of networks. Finally, to demonstrate the viability of the Video Smoother, we have implemented a prototyping system including the Video Smoother performing as part of the transport layer, and a multimedia teleconferencing application. The prototyping results show that the Video Smoother achieves smooth video playout at the expense of only unnoticeable delays.

The remainder of this paper is organized as follows. Section 2 presents our playout smoothing method, including the analytic model and results. The prototyping system and experimental results are then demonstrated in Section 3. Finally, conclusion remarks are given in Section 4.

2. Video Smoother

Generally, the Video Smoother dynamically adopts various playout rates according to the number of frames in the playout buffer in an attempt to compensate for high delay variance of networks. In this section, we first present a queueing model and analysis. The analytic data in turn establish the paradigm on which the playout rates under various network loads are based.

It is worth noting that the Video Smoother has been designed as a general synchronization solution for any generic video encoder/decoder system. It can be implemented in hardware physically co-located with the decoder, or in software functioning as the frontend of the decoder. In the case of supporting a primitive compression-less decoder card, the Smoother indispensably furnishes intra-media synchronization by directly treating captured fixed-size frames as packets. The Smoother can also support sophisticated synchronization-equipped video decoder systems, such as Moving Pictures Expert Group (MPEG) [6, 8, 16, 19]. In this case, video frames are encoded, packetized, and multiplexed [19] as fixed-size packets. These fixed-size packets are eventually received and saved in the decoder buffer from which frames are resumed, synchronized, and displayed. Essentially, owing to the buffer size constraint recommended by the standard organization, the decoder system has to deal with the decoder buffer overflow and underflow problems [8]. The overflow problem results in frame losses and inferior playout quality. On the other hand, the underflow problem, which arises when packets in the buffer are less sufficient for the playout of a picture, yields playout discontinuity. The Video Smoother thus suitably facilitates as a traffic smoother preventing these two problems.

2.1. Model and analysis

The model for the Video Smoother, as shown in figure 1, is composed of an arrival stream of video frames, a finite playout buffer with size N, an output stream of video frames to

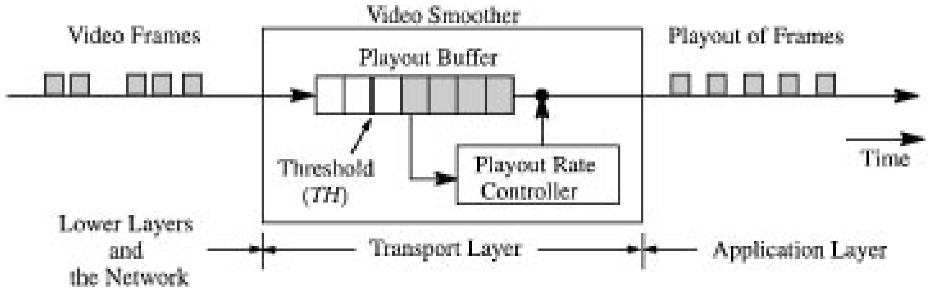


Figure 1. Model of Video Smoother.

be played out, and a playout rate controller responsible for adjusting the playout rate. The playout rate is generally dependent on the current number of video frames in the buffer and the threshold (TH). When the number of frames in the buffer exceeds TH, the Video Smoother employs a maximum playout rate denoted as μ; otherwise, the smoother uses proportionally reduced rates to eliminate playout pauses resulting from the emptiness of the playout buffer.

It is worth noting that the determination of TH can profoundly affect the system performance. If TH is overestimated, the playout rate tends to be reduced which results in serious degradation of playout performance. On the other hand, if TH is underestimated, the probability of having an empty buffer increases which results in playout discontinuity. To determine an optimal TH, we propose a novel queueing analysis where the service rate is state-dependent. In general, we first derive the steady-state queue occupancy distribution as a function of TH. This, in turn, allows us to compute the probability of having an empty buffer and the mean playout rate. The optimal TH can then be selected by trading off rising the probability of having an empty buffer against the increase of the playout rate. In what follows, the analytic model is given in detail.

Let $p_{i,j}$ denote the transition probability of the queue occupancy altered from i to j frames, as seen by departure frames. Thus, the queue occupancies at frame departing epoches form an embedded Markov chain, as shown in figure 2, with the state transition probability matrix (P) given as

$$P = [p_{i,j}] = \begin{bmatrix} p_{0,0} & p_{0,1} & p_{0,2} & \cdots & p_{0,N} \\ p_{1,0} & p_{1,1} & p_{1,2} & \cdots & p_{1,N} \\ p_{2,0} & p_{2,1} & p_{2,2} & \cdots & p_{2,N} \\ & \cdots & & & \\ p_{N,0} & p_{N,1} & p_{N,2} & \cdots & p_{N,N} \end{bmatrix}. \tag{1}$$

To derive the steady-state buffer size distribution, one has to first determine $p_{i,j}$, which is dependent on both the current state (i) and the frame arrival process. The arrival is modelled by the IPP [7], which has been widely accepted as a tractable model for the traffic which is bursty in nature. As shown in figure 3, an IPP changes from state ON to state OFF and state OFF to state ON with probability α and β, respectively. In addition, cells arrive in a rate of λ_{ON} in state ON and no cell arrives in state OFF (i.e., $\lambda_{\mathrm{OFF}} = 0$). Accordingly, the

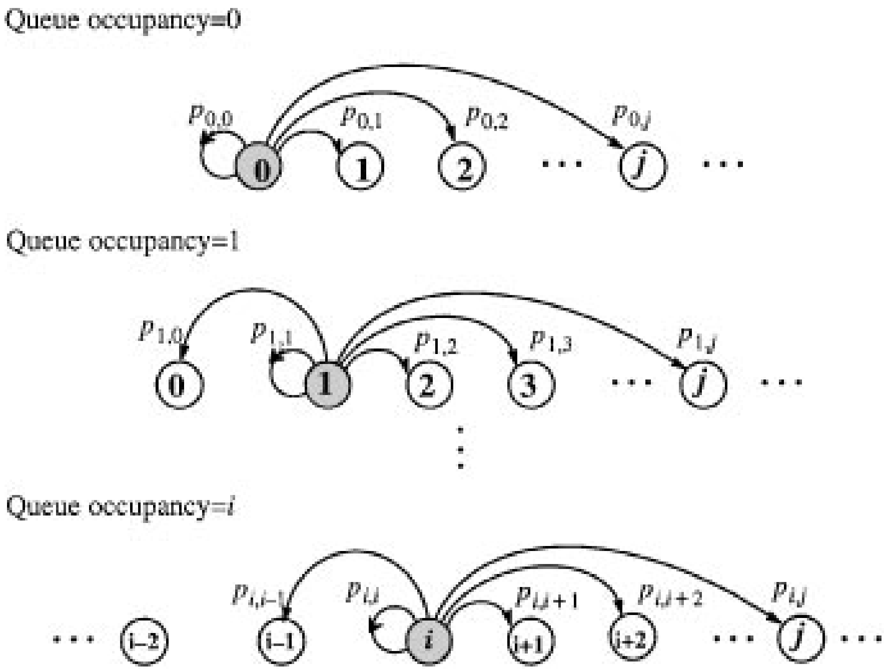


Figure 2. State-transition diagram for queue occupancies.

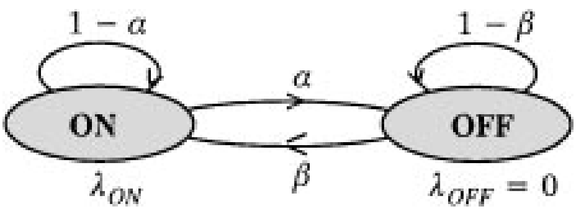


Figure 3. IPP source model.

transition probability matrix (Q) of an IPP can be expressed as

$$Q = \begin{bmatrix} q_{\mathrm{ON,ON}} & q_{\mathrm{ON,OFF}} \\ q_{\mathrm{OFF,ON}} & q_{\mathrm{OFF,OFF}} \end{bmatrix} = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}. \tag{2}$$

The steady-state probability distribution, denoted as $\Pi(\equiv[\Pi_{\mathrm{ON}}\Pi_{\mathrm{OFF}}])$, can be derived by solving the stationary equation $\Pi\boldsymbol{Q} = \Pi$, as

$$\Pi = \left[\frac{\beta}{\alpha+\beta} \quad \frac{\alpha}{\alpha+\beta}\right]. \tag{3}$$

Let $a^t_{r,s}(k)$ denote the probability that k frames arriving during $[t', t' + t - 1]$ within which the process progresses from state r at time slot t' to state s at time slot $t' + t$. It follows immediately that

$$a^1_{r,s}(k) = \begin{cases} 0, & \text{if } r \neq s \text{ or } k > 1, \\ 1 - \lambda_r, & \text{if } r = s \text{ and } k = 0, \\ \lambda_r, & \text{if } r = s \text{ and } k = 1. \end{cases} \tag{4}$$

In addition, by applying the Kolmogorov's Backward Equations [20], we have the following recurrent relation:

$$a^t_{r,s}(k) = \sum_{h \in \{\text{ON,OFF}\}} \left(a^{t-1}_{r,h}(k-1) \cdot \lambda_h + a^{t-1}_{r,h}(k) \cdot (1 - \lambda_h)\right) \cdot q_{h,s}, \quad t > 1. \tag{5}$$

We hereinafter analyze the system of the video smoother by means of a discrete-time (in *frame*) model. Let service time m_i be the number of frame times (e.g., a frame time = 1/30 second for non-interlaced scanning) spent serving a frame given that i customers have been in the buffer. Considering the same smoothing strategy as that given for Poisson arrivals, service time m_i can then be expressed as a function of μ, TH and i, as

$$m_i = \begin{cases} \left\lceil \dfrac{\text{TH}}{\max\{\text{i}, 1\} \cdot \mu} \right\rceil, & i < \text{TH}; \\ \left\lceil \dfrac{1}{\mu} \right\rceil, & i \geq \text{TH}. \end{cases} \tag{6}$$

As for $p_{i,j}$'s, our goal of deriving, we now consider two cases. The first case, as shown in figure 4(a), corresponds to the condition in which a non-empty queue with i frames is left behind by the nth frame departure. As shown in the figure, after the nth frame departs,

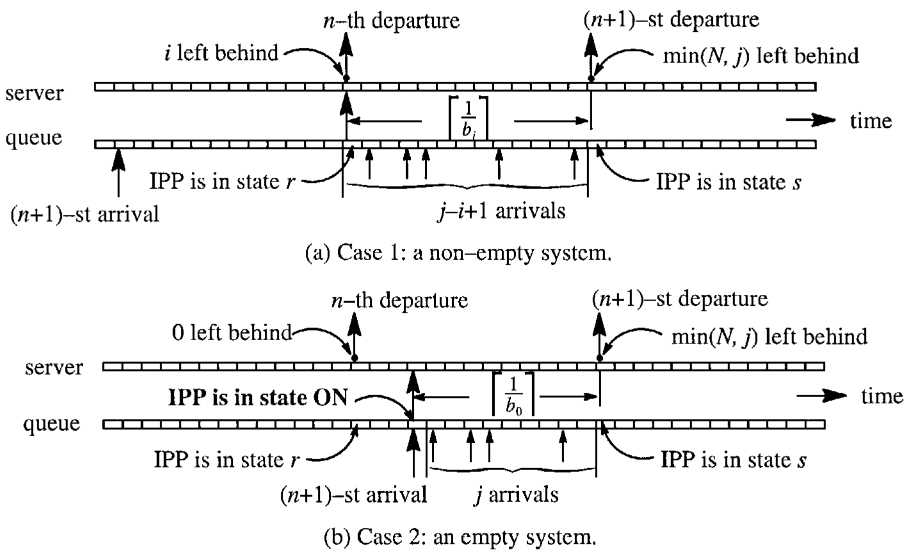


Figure 4. Embedded Markov chain.

the $(n+1)$th frame is immediately served and consumes m_i slots of service time before departing from the system (i.e., playout completes). During this period, frames continue to arrive and enter the playout buffer if the occupancy does not exceed the maximum capacity (N) of the buffer. Otherwise, frames are discarded. Thus, we have

$$p_{i,j} = \begin{cases} \sum\limits_{r\in\{\text{ON,OFF}\}} \Pi_r \cdot \sum\limits_{s\in\{\text{ON,OFF}\}} a_{r,s}^{m_i}(j-i+1), & \text{if } i>0 \text{ and } j<N; \\ \sum\limits_{k=1}^{\infty} \sum\limits_{r\in\{\text{ON.OFF}\}} \Pi_r \cdot \sum\limits_{s\in\{\text{ON,OFF}\}} a_{r,s}^{m_i}(j-i+k), & \text{if } i>0 \text{ and } j=N. \end{cases} \tag{7}$$

The second case, as shown in figure 4(b), corresponds to the condition in which an empty queue is left behind by the nth frame departure. In this case, the number of frames left behind by the departure of the $(n+1)$th frame is merely the same as the number of arrivals during its service time. Notice that the $(n+1)$th frame can arrive only when the IPP is in state ON. Moreover, at the time slot next to the $(n+1)$th arrival, the IPP progresses to state l with probability $q_{\text{ON},l}$ (see Eq. (2)). Consequently, from Eq. (5), we have

$$p_{i,j} = \begin{cases} \sum\limits_{l\in\{\text{ON,OFF}\}} q_{\text{ON},l} \cdot \sum\limits_{s\in\{\text{ON,OFF}\}} a_{l,s}^{m_i-1}(j), & \text{if } i=0 \text{ and } j<N; \\ \sum\limits_{k=N}^{m_i-1} \sum\limits_{l\in\{\text{ON,OFF}\}} q_{\text{ON},l} \cdot \sum\limits_{s\in\{\text{ON,OFF}\}} a_{l,s}^{m_i-1}(k), & \text{if } i=0 \text{ and } j=N. \end{cases} \tag{8}$$

With $p_{i,j}$'s given in Eqs. (7) and (8), we can now derive the limiting distribution of the queue occupancies. Let $\pi_{s,j}$ denotes the stationary probability that the IPP is in state s and the system possesses a queue occupancy of j at a frame departing epoch. The stochastic equilibrium distribution, $\Pi \equiv [\pi_0, \pi_1, \ldots, \pi_N]$, can thus be directly obtained by solving the stationary equation, $\Pi = \Pi P$. The frame loss probability can be respectively considered under the first case (a non-empty system with h frames in the buffer) and the second cases (an empty system) as follows:

$$\begin{cases} \text{case 1: } l_h = \sum\limits_{k=N-h+2}^{m_h} \sum\limits_{r\in\{\text{ON,OFF}\}} \Pi_r \cdot \sum\limits_{s\in\{\text{ON,OFF}\}} a_{r,s}^{m_h}(k); \\ \text{case 2: } l_0 = \sum\limits_{k=N+1}^{m_h-1} \sum\limits_{l\in\{\text{ON,OFF}\}} q_{\text{ON},l} \cdot \sum\limits_{s\in\{\text{ON,OFF}\}} a_{l,s}^{m_h-1}(k). \end{cases} \tag{9}$$

Releasing the condition on two cases, we get the frame loss probability (p_L) as

$$p_L = \sum_{h=0}^{N} (\pi_h \cdot l_h). \tag{10}$$

Furthermore, the mean playout rate $(\bar{B})$ can be formulated as

$$\bar{B} = \sum_{i=0}^{N} \pi_i \cdot \frac{\min\{\text{TH}, \max\{i, 1\}\}}{\text{TH}} \mu. \tag{11}$$

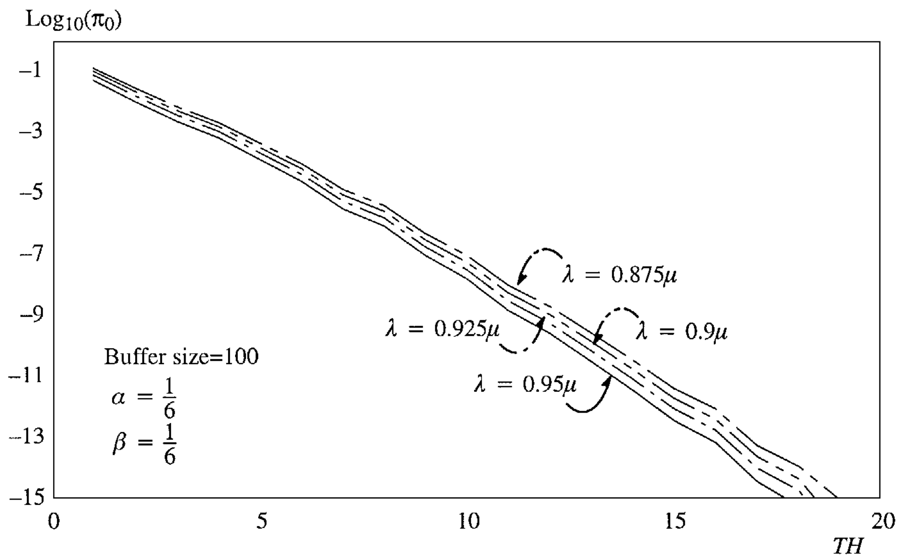


Figure 5. Effect of TH on π_0 under various mean IPP arrival rates.

2.2. *Analytic results*

We have so far obtained the steady-state buffer size distribution as a function of TH. To optimize TH, we consider three variables: the probability of having an empty buffer (π_0), the frame loss probability (p_L), and the mean playout rate ($\bar{B}$). To analyze how TH, π_0, p_L, and $\bar{B}$ are related, we experimented on a system of buffer size = 100 frames, frame size = 15 Kbytes, network access rate = 7.2 Mbps (implying a slot time of 16.67 ms), and the maximal playout rate (μ) = 0.33 frames/slot-time, i.e., 2.4 Mbps (20 frames/sec $\times$ 15 Kbytes/frame $\times$ 8 bits/byte).

In figure 5, π_0 decreases with TH, as was expected. This is because, the greater the TH is, the faster the playout rate reduces, thus the smaller the probability of having an empty buffer. Notice that the mean arrival rate (λ) is equal to $\frac{\lambda_{ON} \cdot \beta}{\alpha+\beta}$, where λ_{ON} is the mean arrival rate in state ON of the IPP. We also observe that, for any given TH, π_0 increases as the mean arrival rate declines. Figure 6 unsurprisingly illustrates that p_L increases with the TH and figure 7 demonstrates that the mean playout rate decreases with TH. This is again because the larger TH, the faster the playout rate reduces. In addition, the figures also show that both the frame loss probability and the mean playout rate increase with the mean arrival rate.

On the whole, a larger TH which results in a smaller π_0, implies pausedless playout; whereas a smaller TH implies better playout quality. In what follows, we present a paradigm of the determination of appropriate THs under various arrivals.

2.3. *Formal description of algorithm*

We now formally present the playout smoothing algorithm employed in the Video Smoother.

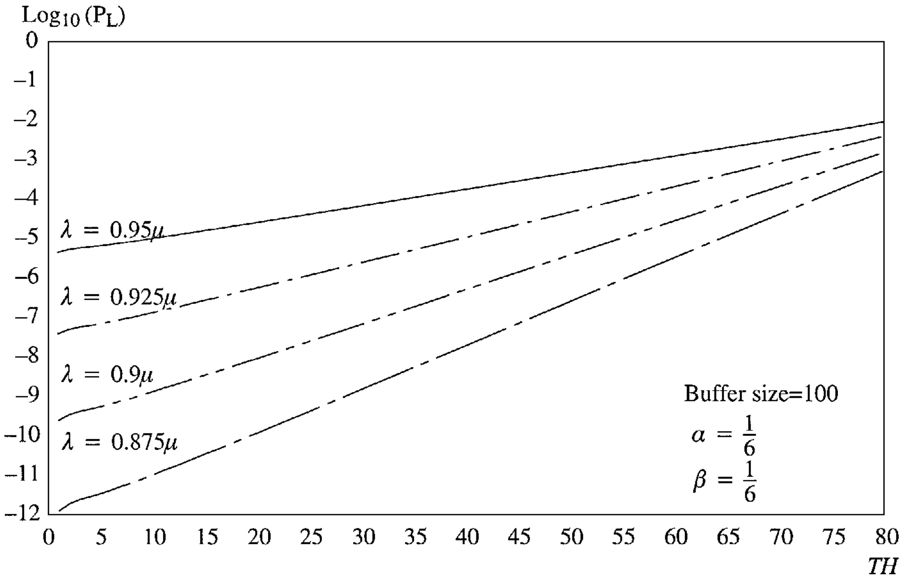


Figure 6. Effect of TH on loss probability under various mean IPP arrival rates.

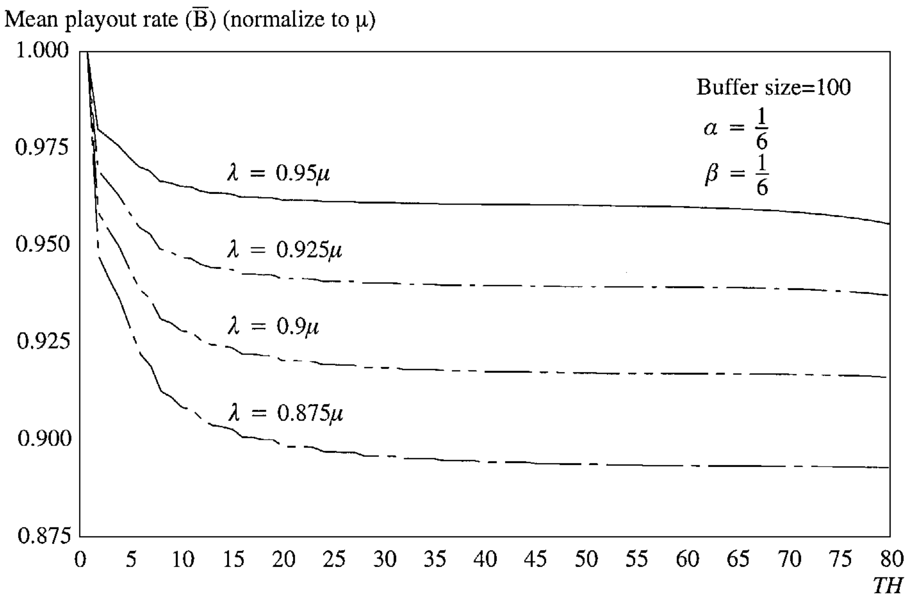


Figure 7. Effect of TH on mean playout rate under various mean IPP arrival rates.

Table 1. Recommended TH's.

Maximum π_0	10^{-2}	10^{-4}	10^{-6}	10^{-8}
Maximum P_L	10^{-8}	10^{-8}	10^{-8}	10^{-8}
Maximum $\bar{B}$	0.954μ	0.939μ	0.930μ	0.928μ
Recommended TH	3	6	9	11

[Video Playout Smoothing Algorithm]

(1) Determine a suitable TH. Based on the analytic results obtained from the previous subsection, a paradigm of TH determination can be constructed for each case of arrivals. Table 1 shows the paradigm for the Video Smoothers achieving four different playout qualities under a given traffic characteristic ($\lambda = 0.9$, $\alpha = \frac{1}{6}$, and $\beta = \frac{1}{6}$).
(2) Determine the playout rate.

while (a frame to be played out) **do**
 if (the number of frames in the buffer $\geq$TH)
  playout_rate = maximum_playout_rate;
 else
  playout_rate = Dynamic_reduced_rate(TH, i);

where the dynamic reduced rate can be selected according to Eq. (6).

[End of Algorithm]

3. Prototyping system and results

The prototyping system (see figure 8) was developed in Intel 80486 personal computers under the MS-Windows environment. The prototyping system is composed of a

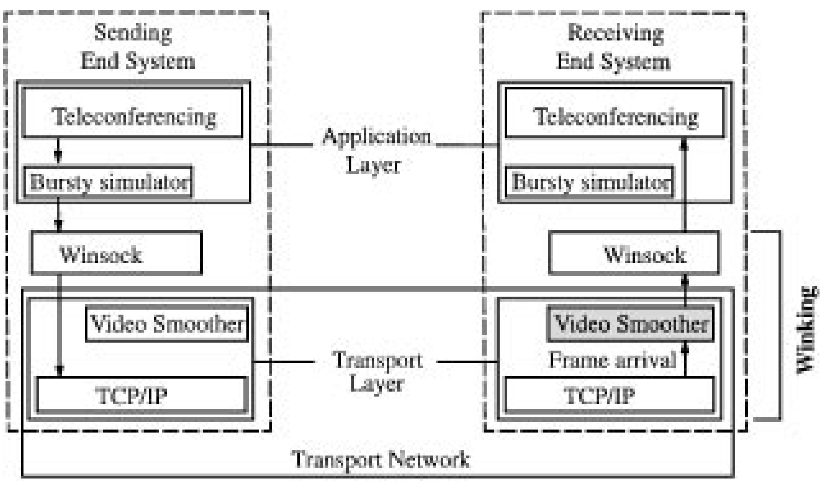


Figure 8. Prototyping system architecture.

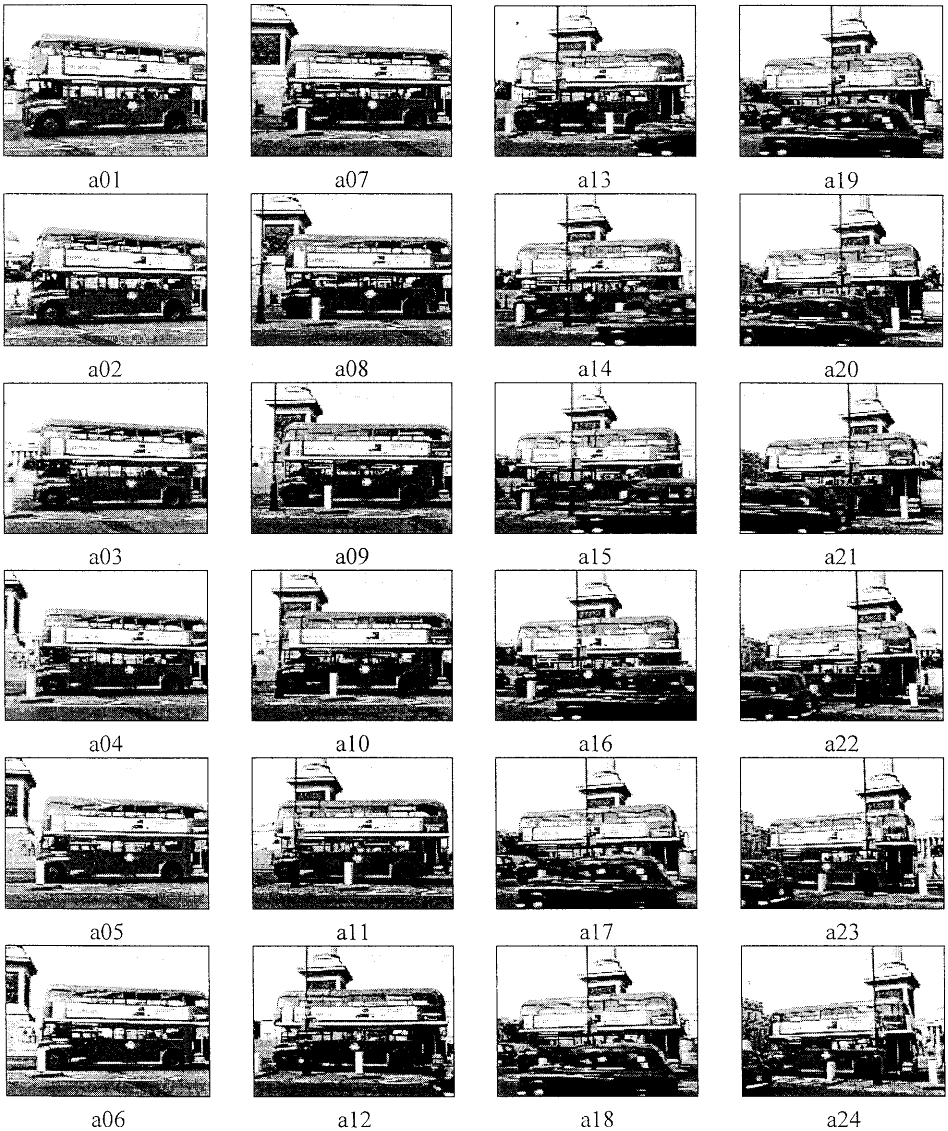


Figure 9a. Prototyping results—playout of a video segment without the Video Smoother.

teleconferencing multimedia application using Intel Smart Video Recorder, WinKing [11] and a bursty simulator. The WinKing package is an implementation of TCP/IP including the API, called Winsock, developed by Institute for Information Industry (III), Taiwan. In particular, the Video Smoother was implemented as a part of WinKing. The frame arrivals in the system are simulated by the bursty simulator as IPP with the following characteristics: $\lambda = 0.9$, $\alpha = \frac{1}{6}$, and $\beta = \frac{1}{6}$. According to the constructed paradigm shown above, we

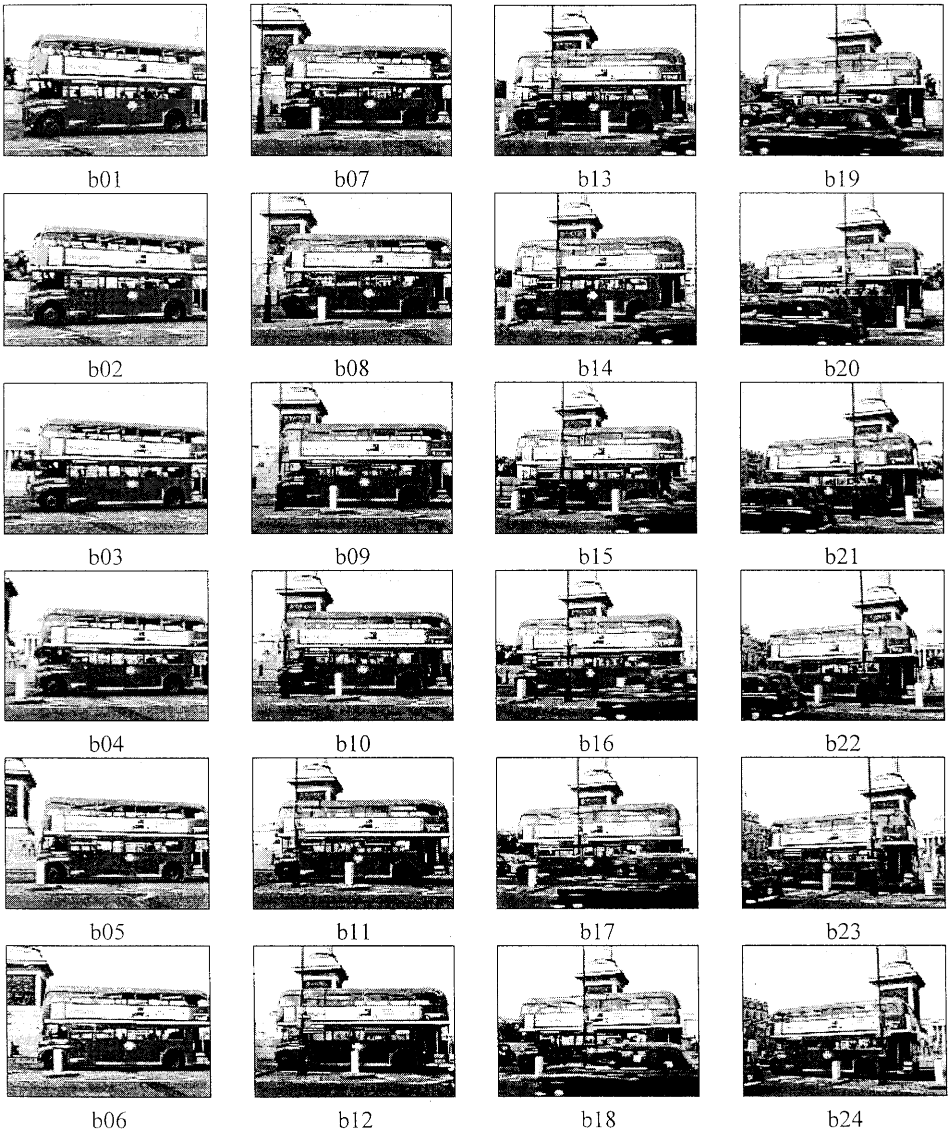


Figure 9b. Prototyping results—playout of a video segment through the Video Smoother.

employed an optimal TH of 9 for the Video Smoothers to achieve the playout quality of $\pi_0 < 10^{-6}$, $p_L < 10^{-8}$, and $\bar{B} \approx 0.930$.

A video segment (see figure 9) from Microsoft Video For Windows, was adopted as the demonstration film. In order to demonstrate the viability of the Video Smoother, the film was captured every 100 ms with and without the Video Smoother. Figures 9(b) and (a) show a series of scenes taken from the film with and without the Video Smoother applied, respectively. Without the Video Smoother, as shown in part (a) of the figure, we observe the playout discontinuity problem. In particular, we reveal playout pauses through a05 to a06 and a17 to a19. By contrast, with the Video Smoother, as shown in the part (b), the movements of the bus in figure 9(b) are much smoother than the ones in figure 9(a), at the expense of unnoticeable delays.

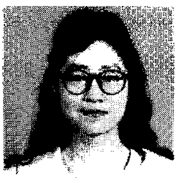
4. Conclusions

The paper proposed a dynamic video playout smoothing method, called the Video Smoother, to prevent potential discontinuity. In contrast with existing methods, the Video Smoother dynamically adopts various playout rates according to a threshold (TH) and the current number of frames in the playout buffer. To precisely determine THs under various traffic conditions, we established an analytic model assuming the Interrupted Poisson arrival. Based on the analytic results, we then proposed a paradigm of determining THs and playout rates achieving different playout qualities under various loads of networks. To demonstrate the viability of the Video Smoother, we developed a prototyping system, including a multimedia teleconferencing application and the Video Smoother as a part of TCP/IP. The prototyping results showed that the Smoother achieves smooth playout at the expense of only unnoticeable delays.

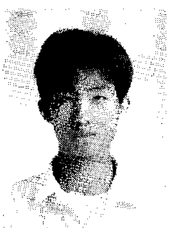
References

1. L. Aguilar, J.J. Garcia-Luna-Aceves, D. Moran, E.J. Craighill, and R. Brungardt, “Architecture for a multimedia teleconferencing system,” Proc. ACM SIGCOMM’86 Symposium, Aug. 1986, pp. 126–136.
2. G. Barberis, “Buffer sizing of a packet-voice receiver,” IEEE Transactions on Communications, Vol. COM-29, No. 2, pp. 152–156, Feb. 1981.
3. J.C. Bolot, “End-to-end packet delay and loss behavior in the internet,” Proc. ACM SIGCOMM, Sept. 1993, pp. 289–298.
4. D. Clark et al., “An analysis of TCP processing overhead,” IEEE Communication Magazine, Vol. 27, No. 6, pp. 23–29, 1989.
5. S. Dixit and P. Skelly, “MPEG-2 over ATM for video dial tone networks: Issues and strategies,” IEEE Network, Vol. 9, No. 5, pp. 30–40, Sept. 1995.
6. D. Gall, “MPEG: A video compression standard for multimedia applications,” Communications of the ACM, Vol. 34, pp. 305–313, April 1991.
7. O. Hashida and S. Shimogawa, “Switched batch bernoulli process (SBBP) and the discrete-time SBBP/G/1 queue with application to statistical multiplexer,” IEEE Journal on Selected Areas in Communications, Vol. 9, No. 3, pp. 394–401, 1991.
8. ISO/IEC 13818-2, MPEG-2—Information Technology—Generic Coding of Moving Pictures and Associated Audio, Part 2: Video, Annex C.
9. V. Jacobson, “Congestion avoidance and control,” Proc. 1988 ACM SIGCOMM, Aug. 1988, pp. 314–329.
10. H. Kanakia, P. Mishra, and A. Reibman, “An adaptive congestion control scheme for real time packet video transport,” IEEE/ACM Transactions on Networking, Vol. 3, No. 6, pp. 671–682, Dec. 1995.
11. J. Lin and J. Chen, “The design and implementation of TCP/IP—WinKing,” Document A3k11221-2, Institute for Information Industry (III), 1994.
12. T.D.C. Little and A. Ghafoor, “Multimedia synchronization protocols for broadband integrated services,” IEEE Journal on Selected Areas in Communications, Vol. 9, No. 9, pp. 1368–1382, Dec. 1991.
13. W.A. Montgomery, “Techniques for packet voice synchronization,” IEEE Journal on Selected Areas in Communications, Vol. SAC-1, No. 6, pp. 1022–1028, Dec. 1983.

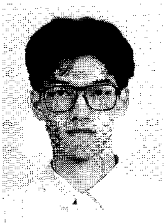
14. W.E. Naylor and L. Kleinrock, “Stream traffic communication in packet switched networks: Destination beffuring considerations,” IEEE Tansactions on Communications, Vol. COM-30, No. 12, pp. 2527–2534, Dec. 1982.
15. C. Nicolaou, “An architecture for real-time multimedia communication systems,” IEEE Journal on Selected Areas in Communications, Vol. 8, No. 3, pp. 391–400, April 1990.
16. P. Parcha and M.El Earki, “MPEG coding for variable bit rate vedio transmission,” IEEE Communications Magazine, Vol. 32, No. 5, pp. 54–66, May 1994.
17. S. Ramanathan and P.V. Rangan, “Feedback techniques for intra-media continuity and inter-media synchronization in distributed multimedia systems,” The Computer Jounal, Vol. 36, No. 1, pp. 19–31, 1993.
18. R. Ramjee, J. Kurose, and D. Towsley, “Adaptive playout mechanisms for packetized audio applications in wide-area networks,” IEEE INFOCOM, 1994.
19. P. Rangan, S. Kumar, and S. Rajan, “Continuity and synchronization in MPEG,” IEEE Journal on Selected Areas in Communications, Vol. 14, No. 1, pp. 52–60, 1996.
20. S.M. Ross, Stochastic Process, 1983.
21. H. Schulzrinne, “Voice communication across the internet: A network voice terminal,” Technical Report, Dept. of Computer Science, U. Massachusetts, Amherst, MA, July 1992.
22. M. Yuang, J. Liu, and C. Shay, “BATS: A high-performance transport system for broadband applications,” Proc. Local Computer Network, 1994.



Maria C. Yuang received the B.S. degree in Applied Mathematics from the National Chiao Tung University, Taiwan, in 1978; the M.S. degree in Computer Science from the University of Maryland, College Park, Maryland, in 1981; and the Ph.D. degree in Electrical Engineering and Computer Science from the Polytechnic University, Brooklyn, New York, in 1989. From 1981 to 1990, she was with AT&T Bell Laboratories and Bell Communications Research (Bellcore), where she was a member of technical staff working on high speed networking and protocol engineering. She has been an associate professor in computer science and information engineering at the National Chiao Tung University, Taiwan, since 1990. Her current research interests include high speed networking, multimedia communications, performance modelling and analysis, and ATM network management.



Shih T. Liang was born in Taiwan, 1968. He received the B.S. degree in Computer Information Science from the Tunghai University, Taiwan, in 1990, and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from National Chaio Tung University, Taiwan, in 1992 and 1996, respectively. His current research interests include high speed networking, multimedia communications, and performance modelling and analysis.



Yu G. Chen was born in Taiwan, 1970. He received the B.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Chiao Tung University, Taiwan, in 1992 and 1997, respectively. His current research interests include network reliability analysis, ATM network management, high speed networking, and multimedia communications.