Intelligent Video Smoother for Multimedia Communications

Maria C. Yuang, *Member, IEEE*, Po L. Tien, and Shih T. Liang

***Abstract*—Multimedia communications often require intramedia synchronization for video data to prevent potential playout discontinuity resulting from network delay variation (jitter) while still achieving satisfactory playout throughput. In this paper, we propose a neural network (NN) based intravideo synchronization mechanism, called the intelligent video smoother (IVS), operating at the application layer of the receiving end system. The IVS is composed of an NN traffic predictor, an NN window determinator, and a window-based playout smoothing algorithm. The NN traffic predictor employs an on-line-trained back-propagation neural network (BPNN) to periodically predict the characteristics of traffic modeled by a generic interrupted Bernoulli process (IBP) over a future fixed time period. With the predicted traffic characteristics, the NN window determinator determines the corresponding optimal window by means of an off-line-trained BPNN in an effort to achieve a maximum of the playout quality (Q) value. The window-based playout smoothing algorithm then dynamically adopts various playout rates according to the window and the number of packets in the buffer. Finally, we show that via simulation results and live video scenes, compared to two other playout approaches, IVS achieves high-throughput and low-discontinuity playout under a mixture of IBP arrivals.**

***Index Terms*—Back-propagation neural network (BPNN), interrupted Bernoulli process (IBP), intramedia synchronization, multimedia communications, network delay variation.**

I. Introduction

RECENT EVOLUTION in high-speed communication technology enables the deployment of distributed multimedia applications combining a variety of media data, such as text, audio, graphics, images, and full-motion video [15]. For supporting distributed multimedia communications, researchers have encountered various design problems including intermedia and intramedia synchronization [17], [19]. In particular, intramedia synchronization for video data 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], [8], we tackle the problem from the end system perspective.

Several existing intramedia synchronization methods, which perform at the end system, exhibit various performance merits. They can be categorized into one of three categories: *static delay-based*, *dynamic feedback-based*, and *dynamic delay-based*. Static delay-based methods preserve playout continuity by buffering massive packets at the receiving end system [13], [14] or delaying the playout time of the first packet received [3], [12], [13], [18]. These methods have been shown to be feasible but at the expense of a drastic decrease in playout throughput. On the other hand, dynamic feedback-based methods [11], [17] perform intramedia synchronization through adjusting the source generation rate by means of sending feedback from receiving end systems. These methods are effective, but are unviable for most live-source applications.

Unlike the two methods described above, the dynamic delay-based method [9] employs reduced playout rates if the current number of packets in the playout buffer falls below a given threshold, which is analytically computed in advance in accordance with a predetermined arrival process. This method has been shown to be viable; however, it may result in the misjudgment of playout rates should the traffic arrival fail to follow the predetermined arrival process.

In this paper, we propose a neural network (NN) based intravideo synchronization mechanism, called the intelligent video smoother (IVS), operating at the application layer of the receiving end system. The IVS is composed of an NN traffic predictor, an NN window determinator, and a window-based playout smoothing algorithm. The source traffic to IVS is modeled as any discrete-time interrupted Bernoulli process (IBP) with unknown probabilistic parameters.

Initially, the NN traffic predictor employs an on-line-trained back-propagation neural network (BPNN) to periodically predict two traffic characteristics (mean busy period and mean idle period) of an IBP arrival over a future fixed time period. With the predicted traffic characteristics, the NN window determinator determines the corresponding optimal *window* by means of an off-line-trained BPNN, in an attempt to achieve a maximum of the playout quality (Q) value defined as a function of mean playout throughput and playout discontinuity. The *window-based* playout smoothing algorithm then dynamically adopts various playout rates according to the window and the number of packets in the buffer. Finally, we show simulation results which demonstrate that compared to two other playout approaches, IVS achieves superior Q under a mixture of IBP arrivals.

The remainder of this paper is organized as follows. Section II presents the main concept and the architecture of the IVS system. Section III describes the NN traffic predictor. Under a predicted traffic arrival, since the off-line training data for the determination of the optimal window are collected by performing the playout smoothing algorithm, the window-based playout smoothing algorithm is first introduced in

Manuscript received February 1996; revised July 1, 1996.

The authors are with the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.

Publisher Item Identifier S 0733-8716(97)00482-4.

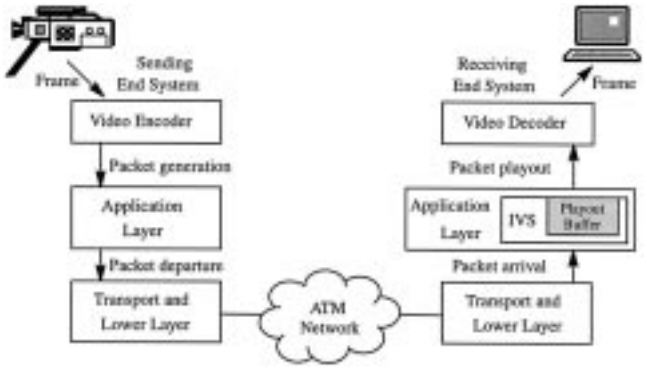

Fig. 1. Protocol stack around IVS.

Section IV. The determination of the optimal window through the NN window determinator is then provided in Section V. Section VI shows performance comparisons and experimental results of the entire IVS system. Finally, conclusion remarks are given in Section VII.

II. Intelligent Video Smoother

A. Concept

The protocol stack on which IVS is established is shown in Fig. 1. Video frames are often captured, encoded, and repacketized into fixed-size packets. These packets are in turn sent through a transport network including lower layers of the sending end system and the relay network, such as an asynchronous transfer mode (ATM) network [5], until reaching the receiving end system. Upon receiving packets which are assumed to arrive in accordance with any IBP (described below) with unknown probabilistic parameters, IVS determines the playout time at which packets are transferred from the IVS playout buffer to the decoder from which frames are resumed and playbacked.

It is worth noting that IVS 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 front end of the decoder. In the case of supporting a primitive compression-less decoder card, IVS indispensably furnishes intramedia synchronization by directly treating captured fixed-size frames as packets. The IVS can also support sophisticated synchronization-equipped video decoder systems, such as the moving pictures expert group (MPEG) [1], [6], [15], [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 [1]. 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

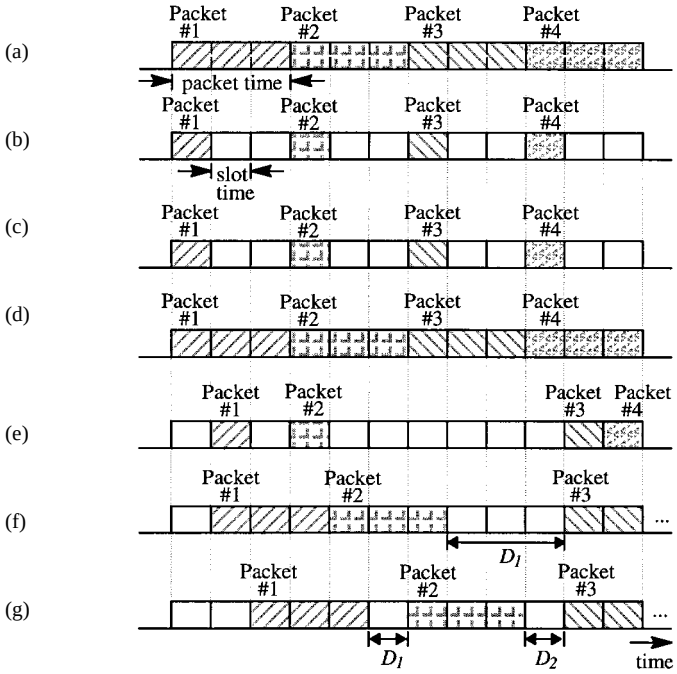

Fig. 2. Concept of IVS. (a) Generation of packets at the application layer of the sending end system (2.32 Mb/s). (b) Packets processed and sent by the transport layer (7 Mb/s). (c) Ideal reception of packets at the transport layer of the receiving end system. (d) Ideal playout of packets at the application layer of the receiving end system. (e) Imperfect reception of packets at the transport layer of the receiving end system due to delay jitters. (f) Playout of packets at the application layer without synchronization. (g) Playout of packets at the application layer via IVS.

playout discontinuity. The IVS thus suitably facilitates as a traffic smoother preventing these two problems.

The design principle of IVS is further illustrated via an example shown in Fig. 2. In the example, any constant delay, such as the interface delay between adjacent layers and the propagation delay throughout the network, is ignored. The time axis in IVS is slotted by the processing and arrival of a single packet from the adjacent lower layer, namely the transport layer. Moreover, we assume that disregarding the packetizing overhead, video packets are generated at the application layer of the sending end system [Fig. 2(a)] at a rate of 2.32 Mb/s (30 frames/s $\times$ 220 cells/frame [15] $\times$ 44 bytes/cell $\times$ 8 bits/byte). These packets are in turn sent to the transport layer operating at a rate of approximately 7 Mb/s [4], [23], [Fig. 2(b)].

Define $\mathcal{F}$ as the ratio of the generation or playout of a packet, referred to as the *packet time*, to the processing of a single packet at the transport layer, referred to as the *slot time*

$$\mathcal{F} = \frac{\text{packet time}}{\text{slot time}}. \tag{1}$$

For the example given in Fig. 2, $\mathcal{F} = 3$ (7 Mb/s/2.32 Mb/s). It is worth noting that packets are "played out" from IVS at a maximum rate of $1/\mathcal{F}$, i.e., 1/3 in this example, to prevent the decoder buffer from overflowing in the case of supporting MPEG.

Packets are finally received at the transport layer of the receiving end system. Ideally, as shown in Fig. 2(c), the interarrival times of packets at the transport layer of the receiver are the same as the interdeparture times of packets

at the transport layer of the sending end system. In that case, packets are played out without discontinuity at the maximum rate, as shown in Fig. 2(d). However, due to delay jitters [16] induced in networks, different packets yield different end-to-end delays [Fig. 2(e)] causing playout discontinuity. Define the *variance of discontinuity* (VOD) as

$$\text{Variance of discontinuity (VOD)} = E\{(D_i - E[D_i])^2\} \tag{2}$$

where E is the expectation function and D_i is the ith discontinuity duration during the playout. For example, D_1 in Fig. 2(f) is three slots in length, whereas both D_1 and D_2 in Fig. 2(g) are one slot long. Fig. 2(f) and (g) depict the playout of packets without and with intravideo synchronization, respectively.

Moreover, playout discontinuity can be reduced at the expense of a rise in playout delay or a decrease in playout throughput. Define the mean playout throughput (MPT) as

$$\text{Mean playout throughput (MPT)} = E\left[\frac{1}{\text{packet sojourn time}}\right] \tag{3}$$

where the *packet sojourn time* is defined as the elapsed time between packet arrival and departure in IVS. In this example, the sojourn times of Packet #2 in Fig. 2(f) and (g) are two and four slots, respectively. Apparently, for any high-burstiness arrival, the playout of packets without intravideo synchronization achieves the highest MPT at the expense of an increase in the VOD, as shown in Fig. 2(f). In contrast, as shown in Fig. 2(g), the playout of packets with intravideo synchronization exhibits a lower VOD but at the expense of a decrease in the MPT. The IVS has thus been designed to achieve a minimum of VOD and a maximum of MPT. Define playout quality (Q) as a function of the MPT and VOD

$$\text{Playout quality}\,(Q) = f(MPT, VOD). \tag{4}$$

Three issues have been raised in the design of IVS. First, how can future traffic be foreseen? Second, how can the playout rates be determined achieving a maximum of Q value? Finally, how can one select a Q which can be soundly associated with the perceptional demand of the video application under consideration? The solutions to these three issues are addressed in the following sections after the source traffic model and the architecture of IVS are first presented in the next two subsections.

B. Source Traffic Model

The source traffic to IVS is modeled by a generic discrete-time IBP [7], which has been widely accepted to model the traffic which is bursty in nature. The process alternates between the busy and the idle states, as shown in Fig. 3. Notice that rather than confine the source traffic model to a given IBP, IVS adopts a *generic* IBP allowing any combination of transitional probabilities. In the figure, α defines the probability of switching from the busy to the idle state and β defines the opposite probability. Moreover, in any time slot packets arrive in a rate of $\lambda_1 = \lambda$ during the busy state and in a rate of $\lambda_0 = 0$ during the idle state. That is, one packet is generated

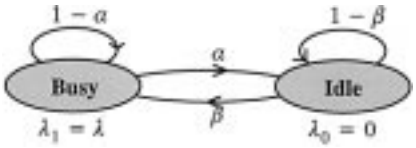

Fig. 3. Source traffic model.

with probability λ per time slot during the busy state, and no packet is generated during the idle state. The steady-state probability of being at each state, denoted as Π_{busy} and Π_{idle}, can be computed using $\Pi = \Pi P$, where $\Pi = [\Pi_{busy}, \Pi_{idle}]$, as

$$\Pi_{busy} = \frac{\beta}{\alpha + \beta}$$

and

$$\Pi_{idle} = \frac{\alpha}{\alpha + \beta}. \tag{5}$$

In our work, we approximate this source traffic distribution by four characteristics: *mean packet rate* ($\overline{P}$), *mean busy period* ($\overline{B}$), *mean idle period* ($\overline{I}$), and *burstiness* (b) [16]. Accordingly, for an IBP arrival defined by parameters α, β, and λ, the four traffic characteristics in terms of α, β, and λ are given by

$$\overline{P} = \Pi_{busy} \times \lambda = \frac{\beta \times \lambda}{\alpha + \beta}$$

$$\overline{B} = \frac{1}{\alpha}$$

$$\overline{I} = \frac{1}{\beta}$$

and

$$b = \frac{1}{\Pi_{busy}} = \frac{\alpha + \beta}{\beta}. \tag{6}$$

Table I summarizes nine different traffic arrivals with various $\overline{P}$'s and $\overline{I}$'s under a fixed $\overline{B}$, which are used throughout the rest of the paper. For ease of illustration, arrival rate λ is assumed to be one in all cases. As a result, $\overline{P}$ becomes a function of $\overline{B}$ and $\overline{I}$, i.e., $\overline{P} = \overline{B}/(\overline{B} + \overline{I})$. These two traffic characteristics, namely $\overline{B}$ and $\overline{I}$, as will be shown, are to be predicted by the NN traffic predictor of IVS. Notice that for traffic arrivals exhibiting b higher than six, we observe that playout discontinuity or underflowing of the decoder buffer can no longer be avoided regardless of the consideration of synchronization.

C. System Architecture

IVS is composed of three major components (see Fig. 4): NN traffic predictor, NN window determinator, and the window-based playout smoothing algorithm. For a future fixed time interval, the NN traffic predictor employs a BPNN to predict two traffic characteristics, $\overline{B}$ and $\overline{I}$, of the traffic arrival over this interval. With the predicted traffic characteristics, the NN window determinator determines the corresponding optimal window achieving a maximum of the Q value by means of an off-line pre-trained BPNN. The window-based playout smoothing algorithm then dynamically adopts various playout rates according to the window and the number of

TABLE I
NINE TRAFFIC ARRIVALS

IBP arrival			Mean Packet Rate	Mean Busy	Mean Idle	Burstiness
α	β	λ	($\overline{P}$ packets/slot)	Period ($\overline{B}$ slots)	Period ($\overline{I}$ slots)	(b)
0.333	0.142	1	0.300	3	7	3.33
0.333	0.125	1	0.273	3	8	3.67
0.333	0.111	1	0.250	3	9	4.00
0.333	0.100	1	0.230	3	10	4.33
0.333	0.090	1	0.214	3	11	4.67
0.333	0.083	1	0.200	3	12	5.00
0.333	0.077	1	0.188	3	13	5.33
0.333	0.071	1	0.176	3	14	5.67
0.333	0.066	1	0.167	3	15	6.00

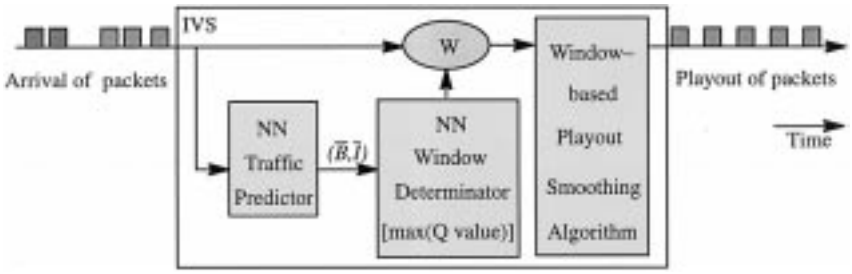

Fig. 4. IVS system architecture.

packets in the buffer within this interval. The complete process repeats for the next future time interval until the end of the connection. In the following sections, each component is described in detail followed by the demonstration of experimental results of the entire IVS system.

III. NEURAL NETWORK TRAFFIC PREDICTOR

Substantially, we have discovered several strengths of NN's with respect to the training of traffic distributions. On the whole, while the off-line learning of traffic distributions has been shown to be feasible and straightforward, the on-line training [22] of highly bursty traffic is more challenging. In principle, a viable video smoother should refrain from predicting specific but perhaps biased local traffic behavior. In lieu, it should adopt more general traffic characteristics (e.g., *mean* behavior) in an attempt to capture both local and global traffic behavior, still without susceptibly suffering from playout quality degradation should the traffic be occasionally imperfectly predicated. The NN traffic predictor, coupled with the window-based playout mechanism, has been designed to satisfy this need.

The NN traffic predictor employs an on-line trained BPNN to predict $\overline{B}$ and $\overline{I}$ of the traffic over a fixed future time duration based on traffic characteristics taken from a set of overlapping past time intervals. More explicitly, the NN is modeled, as shown in Fig. 5, as

$$\hat{M}(t_c, FI) = \mathrm{NN}_f\{M(t_c, [PI]_n, C), WG\}. \qquad (7)$$

In the equation, NN_f denotes the NN function and WG represents the weight matrix of the links between neurons. $M(t_c, [PI]_n, C)$ denotes the n sets of input vectors ($\overline{B}$'s and

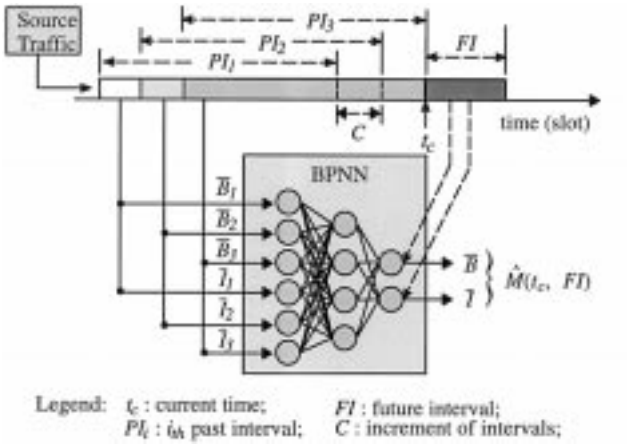

Fig. 5. NN traffic predictor.

$\overline{I}$'s) representing the traffic characteristics respectively taken from n overlapping past time intervals each of which is of same length and of distance C from the adjacent interval, up to the present time t_c. $\hat{M}(t_c, FI)$ denotes the output vector ($\overline{B}$ and $\overline{I}$) representing the traffic characteristics over the time duration FI. At any of the following time instants, $\cdots t_c - FI, t_c, t_c + FI, \cdots$, say $t_c + FI$, in addition to predicting future traffic as described above, the NN also performs the back-propagation training operation by updating the WG based on the traffic measurements ($\overline{B}$ and $\overline{I}$) over the past time duration $t_c + FI$.

The selections of the PI, FI, and C are crucial to the performance of the NN traffic predictor. Generally, we have observed that the larger the number of overlapping PI's

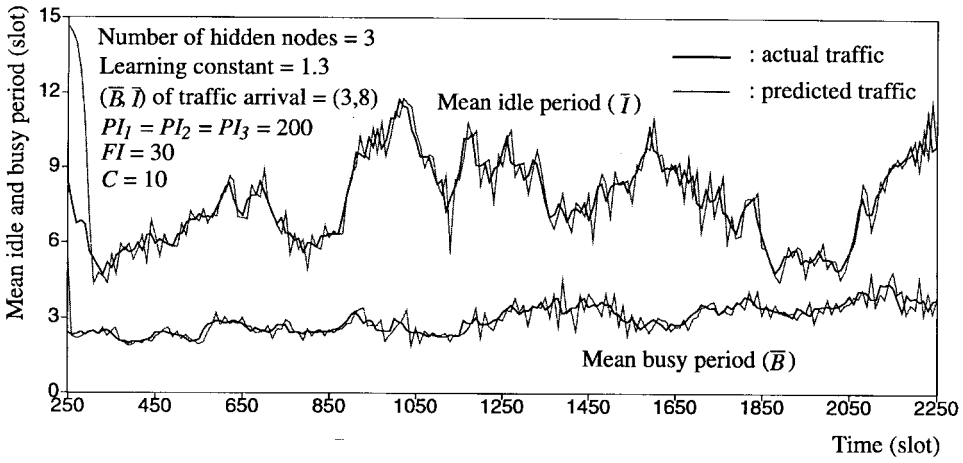

Fig. 6. Comparison of actual and predicted traffic.

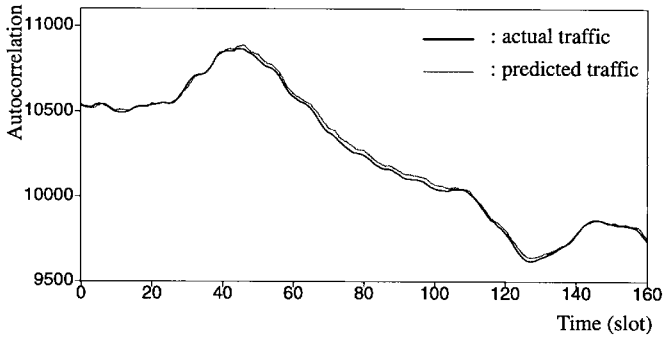

Fig. 7. Autocorrelation function of actual and predicted traffic.

and the smaller the C is, the more precisely traffic characteristics $\overline{B}$ and $\overline{I}$ can be predicted. Moreover, decreasing the FI yields more accurate prediction but imposes higher computational overhead. In contrast, increasing the FI incurs inferior prediction and delayed training of the NN. However, we have surprisingly noticed that as the FI falls below a value, inaccurate prediction is revealed. In a subsequent section, we show this phenomenon by demonstrating the Q value with respect to the FI. Our goal is to offer the determination of an appropriate FI aiming to achieve acceptable Q at the expense of reasonable computational overhead.

Fig. 6 draws comparisons of the mean idle and busy periods ($\overline{I}$ and $\overline{B}$) between the actual and predicted traffic assuming IBP arrival ($\overline{P}$, $\overline{B}$, $\overline{I}$) = (0.273, 3, 8). In this experiment, we employed a three-layer NN with three hidden nodes and a learning constant of 1.3. We have observed that the variation of traffic characteristics is greatly captured by the predictor. In addition, the NN traffic predictor demands only a reasonably short adapting period before being able to accurately and stably predict traffic arrivals, as shown by the sharp curve drop within the initial 300 time slots in Fig. 6. In Fig. 7, we show the autocorrelation function of the traffic produced by the NN traffic predictor compared with that of the actual traffic, for the same traffic arrival. Both figures justify superior characterization and learning of traffic behavior by the NN traffic predictor.

IV. Window-Based Playout Smoothing Algorithm

Generally, IVS dynamically adopts various playout rates according to the window (W) (described later) and the current number of packets in the playout buffer. For example, given a window size of 18 slots long, a maximum playout rate (i.e., immediate playout without delay) is applied if the number of packets in the playout buffer equals or exceeds 18 (slot time) $\div$ 3 (slot time/packet time) $= 6$ packets. Otherwise, if the number of packets is less than six, a reduced playout rate is applied for the playout of the next packet in the buffer. In this example, if there are five packets in the buffer, these five packets (15 slots of playout time) are to be evenly played out within 18 slots. That is, the remaining three slots should be evenly spread in six gaps among packets within the window. Consequently, the playout of the next packet incurs $\lceil 3/6 \rceil$ slot of a delay. Upon finishing the playout of this packet, the number of packets in the buffer is re-examined and the new playout time of the next packet is redetermined. The same procedure repeats until the end of the connection. It is worth noticing that the playout without intravideo synchronization corresponds to the playout through IVS using a window size of $\mathcal{F}$, i.e., $W = \mathcal{F}$. In this case, employing window size $W = 3$ corresponds to the playout without synchronization. Fig. 8 depicts the detailed playout smoothing algorithm.

To examine the effect of the window size on the VOD and MPT of the playout based on the playout smoothing algorithm under a variety of traffic arrivals, we carried out an experiment via simulation. Results are plotted in Figs. 9 and 10. Unsurprisingly, both the VOD and MPT decrease with the window size. As shown in Fig. 9, to gain an acceptably low VOD, traffic of higher burstiness requires larger window sizes. However, as shown in Fig. 10, to achieve a satisfactory MPT, high-burstiness traffic requires smaller window sizes. Namely, increasing the window size results in a reduction in the VOD but at the expense of a decrease in the MPT, and vice versa.

In principle, the optimal window size should be selected by balancing the rise in the VOD against the fall in the MPT. Two problems now arise. First, how can one define the combinatorial function of VOD and MPT, i.e., the Q

Fig. 8. Video playout smoothing algorithm.

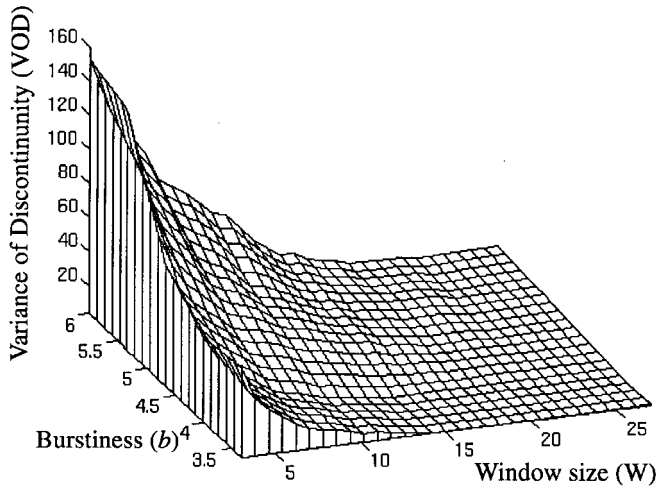

Fig. 9. VOD attained based on the window-based playout smoothing algorithm.

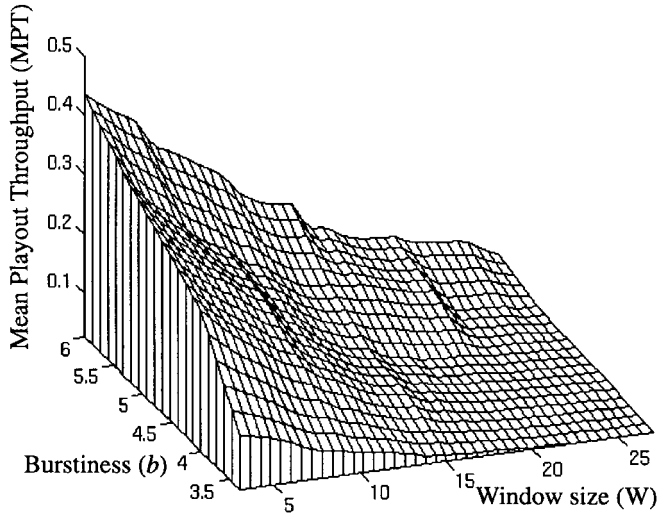

Fig. 10. MPT achieved based on the playout smoothing algorithm.

(previously defined as a function of the VOD and MPT), so as to associate it with the perceptional requirement in mind?

TABLE II
FOUR Q ASSOCIATED WITH FOUR PERCEPTIONAL REQUIREMENTS

Type	Definition	MPT Significance	VOD Significance
Q_1	$\frac{MPT}{VOD^2}$	Fair	High
Q_2	$\frac{MPT}{VOD}$	Fair	Fair
Q_3	$\frac{MPT}{VOD^{1/2}}$	Fair	Low
Q_4	$\frac{MPT^{3/2}}{VOD^{1/2}}$	High	Low

The second problem is then how to determine the optimal window size in an attempt to achieve a maximum of Q value? Notice that formally solving the first problem is beyond the scope of this paper. Nevertheless, in the next section, we define and examine four different Q's which are used to correspond to four perceptional requirements. The second problem is afterward discussed in great detail in the same section.

V. NN WINDOW DETERMINATOR

Multimedia applications often have different grades of perceptional requirements in terms of VOD and MPT. For example, while teleconferencing systems demand stringent MPT's, video-on-demand systems require bounded VOD's. To quantize perceptional requirements, we define four different types of Q's (Q_1, Q_2, Q_3, and Q_4) exhibiting various significances of VOD and MPT, as summarized in Table II. It is worth noting that these four Q types are ranked in order of an increasing significance of MPT and decreasing significance of VOD.

For each type of Q, based on the playout smoothing algorithm, we attained normalized Q values (between 0–1) under all traffic arrivals and window sizes. Results of four types of Q's are plotted in Fig. 11(a)–(d), respectively. Saliently, we discover that under any given traffic arrival, the optimal window (the window achieving a maximum of Q value) declines from Q_1 through Q_4. For example, the optimal window size for Q_1 through Q_4, under the traffic arrival of a burstiness of 6, drops from 23, 18, 15, until 4. This is because achieving a maximum of Q value entails a smaller window size under an increasing weight of MPT and a decreasing weight of VOD. The result agrees with that revealed in Figs. 9 and 10.

To determine the optimal window size in real time for any Q type, we design a NN-based window determinator which has been off-line trained via the experimental results from Fig. 11. Fig. 12 depicts the optimal window size for the Q_2 type under a variety of $\overline{B}$'s and $\overline{I}$'s. Clearly, as shown in the figure, the optimal window size W increases with the burstiness of the traffic arrival. We also surprisingly discover that W is also dependent on $\overline{I}$ other than the burstiness. For instance, optimal window sizes W's are 22, 30, and 36, respectively, for three arrivals $(\overline{B}, \overline{I}) = (3, 15)$, $(\overline{B}, \overline{I}) = (4, 20)$, and $(\overline{B}, \overline{I}) = (5, 25)$ all exhibiting the same burstiness $(b = 6)$. Specifically, the higher the $\overline{I}$ the larger the W.

The NN window determinator uses a three-layer fully connected NN and the back-propagation learning algorithm. During the off-line training phase, the input signals to the NN

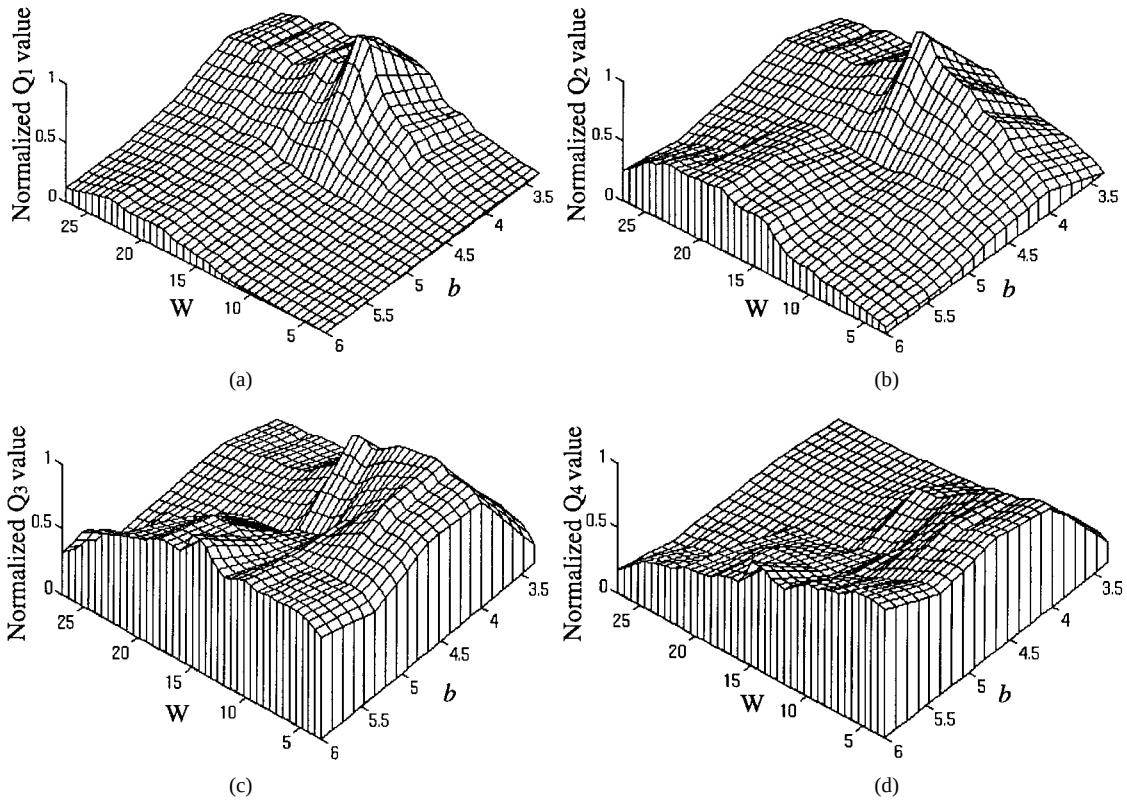

Fig. 11. Four Q's corresponding to four perceptional requirements. (a) Fair MPT and high VOD. (b) Fair MPT and fair VOD. (c) Fair MPT and low VOD. (d) High MPT and low VOD.

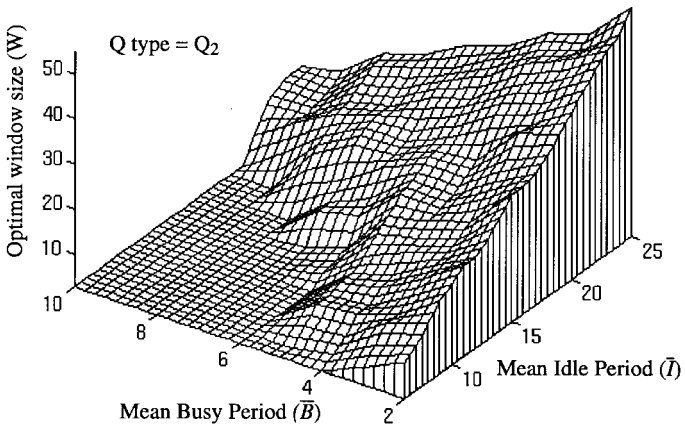

Fig. 12. Optimal window achieving a maximum of Q_2 value.

window determinator are characteristics $\overline{B}$ and $\overline{I}$, of the traffic predicted by the NN traffic predictor. The desired output is the corresponding optimal window achieving a maximum of the Q value. All the weights of the NN are then learned at the end of the training phase. With the determined weights, during the on-line operation phase the optimal window can then be efficiently and precisely determined for any given traffic arrival.

VI. Experimental Results of the IVS System

We experimented on the entire IVS system via simulation. In the experiment, we considered the Q_2 type and assumed that the ratio ($\mathcal{F}$) of packet time to slot time is three and any traffic is comprised of a mixture of IBP arrivals. To demonstrate the viability of the IVS system, we employed three playout approaches: dynamic-window-based (IVS), static-window-based, and synchronization—less (i.e., $W = 3$ in this case).

The dynamic-window-based approach corresponds to the playout through the IVS system. In this case, we applied a variety of FI's to the NN traffic predictor. For any given FI, say $FI = 50$ for instance, the NN traffic predictor predicted $\overline{B}$ and $\overline{I}$ within the first 50-slot interval. The NN window determinator determined the optimal window for this interval. Based on the optimal window attained, the playout smoothing algorithm then performed the playout of packets within the interval. The same procedure repeated for the next 50-slot interval until all 1450 slots have been playbacked.

The static-window-based approach corresponds to the deployment of previously surveyed dynamic delay-based method [9] which was regarded as one of the most promising approaches should the traffic follow the pre-assumed arrival process. It is worth noting that the optimal static window is logically identical to the threshold [9]. In this case, we experimented on all different windows for the entire playout duration assuming the arrival process is known in advance, and selected the one yielding a maximum of Q_2 value. Finally, for the playout without synchronization, a maximum playout rate (i.e., playout without any delay) was employed.

Fig. 13 shows the Q_2 value of the playout based on these three playout approaches. In the experiment, we adopted three types of arrivals each of which is composed of a mixture of IBP arrivals with the same burstiness. In addition, we employed different PI's under different FI's in the NN

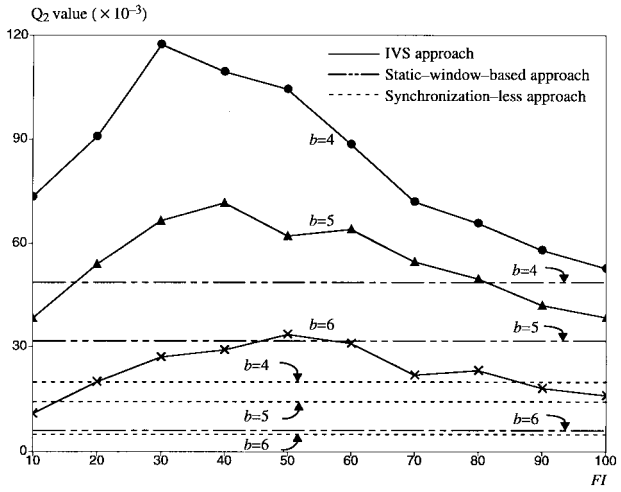

Fig. 13. Comparisons of Q_2 values achieved based on three playout approaches.

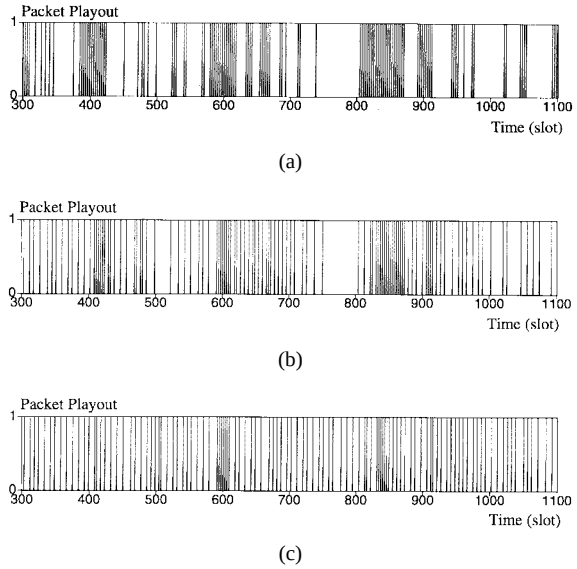

Fig. 14. Playout of video packets over time. (a) Synchronization-less playout (MPT = 0.393; VOD = 147.05; $Q_2 = 2.67 \times 10^{-3}$). (b) Static window-based playout ($W = 18$; MPT = 0.151; VOD = 28.27; $Q_2 = 5.34 \times 10^{-3}$). (c) Dynamic window-based (IVS) playout (MPT = 0.121; VOD = 3.593; $Q_2 = 33.68 \times 10^{-3}$).

traffic predictor. The figure shows that on the basis of the IVS approach, the Q_2 value first grows with the FI. This result is reasoned by the fact that excessively short FI's incur poor traffic prediction resulting from being deceived by the local burstiness. However, as the FI increases to the degree that both local and global traffic characteristics can be greatly captured, the Q_2 value then declines with the FI. This is because shorter FI yields better prediction, and higher dynamism (i.e., frequent adjustment of WG) leads to better playout quality. Furthermore, the Q_2 value achieved based on the static-window approach is invariantly lower than that based on the IVS approach. Unsurprisingly, the playout without synchronization results in the lowest Q_2 value.

Fig. 14 depicts the playout of video packets over time using an FI of 50-slot long under a combination of three back-to-back IBP arrivals: $(\overline{B}, \overline{I}) = (2, 10)$ in time period (0–650), $(\overline{B}, \overline{I}) = (3, 15)$ in time period (650–1050), and $(\overline{B}, \overline{I}) = (4, 20)$ in time period (1050–1450). In the figure, "1" represents the playout of a packet, whereas "0" represents the lack of any packet being played out. Fig. 14(a) exhibits the synchronization-less playout achieving the highest MPT but the poorest VOD. Fig. 14(b) depicts the playout based on the static-window approach. The playout incurs a lower MPT but achieves a better VOD. Fig. 14(c), by dynamically applying optimal windows in each interval over time, IVS achieves the most superior playout yielding a maximum of the Q_2 value.

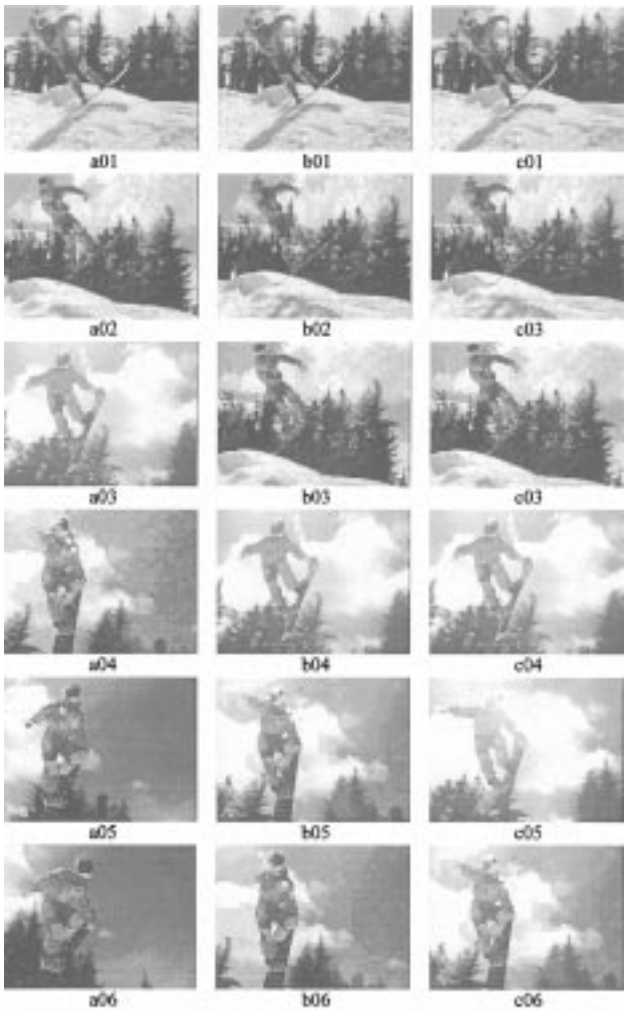

Fig. 15. Playout of snow-skiing scenes based on synchronizationless (a01–a06), static-window-based (b01–b06), and IVS (c01–c06) approaches.

We further carried out an experiment via a simulation on the playout of a series of snow-skiing scenes by means of the three playout approaches. First of all, using the generated simulation results shown in Fig. 14, we attained the playout epochs (in time slot) of video packets numbered from 1–60, as shown in Table III(a). For example, the playout of Packet #1, based on the synchronization-less, static-window, and dynamic-window (IVS) approaches, takes place at time slots 400, 416, and 418, respectively. Then, we captured a series of 34 consecutive skiing scenes (from a snow-skiing video program) corresponding to packets #20–#53. Within these 34 scenes, we playbacked 18 scenes every 14 time slots starting from the 567th time slot. These 18 scenes are referred to as a01–a18,

TABLE III
VIDEO PACKETS AND CORRESPONDING SKIING SCENES. (a) PLAYOUT EPOCHS FOR VIDEO PACKETS. (b) PLAYOUT WITHOUT SYNCHRONIZATION. (c) PLAYOUT BASED ON STATIC-WINDOW APPROACH. (d) PLAYOUT BASED ON DYNAMIC-WINDOW APPROACH (IVS)

P#	P_{SL}	P_{SW}	P_{IVS}	P#	P_{SL}	P_{SW}	P_{IVS}	P#	P_{SL}	P_{SW}	P_{IVS}	P#	P_{SL}	P_{SW}	P_{IVS}
1	400	416	418	16	521	523	533	31	603	618	626	46	668	697	746
2	403	419	425	17	524	534	540	32	606	622	634	47	684	703	752
3	406	422	433	18	529	545	548	33	609	627	643	48	687	710	759
4	409	425	441	19	539	554	554	34	612	633	650	49	693	721	767
5	412	429	451	20	543	565	563	35	615	640	659	50	711	728	777
6	415	434	462	21	568	570	570	36	618	645	669	51	715	739	788
7	418	440	469	22	571	581	581	37	633	650	675	52	738	750	796
8	421	447	477	23	579	591	591	38	636	656	682	53	803	805	805
9	424	458	486	24	582	595	595	39	639	662	689	54	806	815	813
10	451	469	493	25	585	598	598	40	643	666	696	55	809	822	818
11	471	473	499	26	588	601	601	41	653	669	703	56	812	827	824
12	477	479	504	27	591	604	604	42	656	673	711	57	815	832	831
13	480	482	509	28	594	607	607	43	659	678	720	58	818	835	836
14	486	488	516	29	597	610	610	44	662	684	728	59	821	838	839
15	498	500	526	30	600	614	618	45	665	691	737	60	824	841	842

(a)

S#	P_{SL}	P#	S#	P_{SL}	P#
a01	567	20	a10	693	49
a02	581	23	a11	707	49
a03	595	28	a12	721	51
a04	609	33	a13	735	51
a05	623	36	a14	749	52
a06	637	38	a15	763	52
a07	651	40	a16	777	52
a08	665	45	a17	791	52
a09	679	46	a18	805	53

(b)

S#	P_{SW}	P#	S#	P_{SW}	P#
b01	567	20	b10	693	45
b02	581	22	b11	707	47
b03	595	24	b12	721	49
b04	609	28	b13	735	50
b05	623	32	b14	749	51
b06	637	34	b15	763	52
b07	651	37	b16	777	52
b08	665	39	b17	791	52
b09	679	43	b18	805	53

(c)

S#	P_{IVS}	P#	S#	P_{IVS}	P#
c01	567	20	c10	693	39
c02	581	22	c11	707	41
c03	595	24	c12	721	43
c04	609	28	c13	735	44
c05	623	30	c14	749	46
c06	637	32	c15	763	48
c07	651	34	c16	777	50
c08	665	35	c17	791	51
c09	679	37	c18	805	53

(d)

Legend: P#: packet number;
S#: scene number;
P_{SL}: epoch (in slot time) of playout based on synchronization–less approach;
P_{SW}: epoch (in slot time) of playout based on static–window approach;
P_{IVS}: epoch (in slot time) of playout based on dynamic–window approach (IVS);

b01–b18, and c01–c18 in the cases of synchronization-less, static-window-based, and IVS approaches, respectively. The packet numbers, playout epochs, and corresponding scene numbers are summarized in Table III(b)–(d). In the table, for instance, at time slot 567, the scene displayed based on the synchronization-less playout is a01 or Packet #20. After an elapsed time of 56 (14×4) slots ($567 + 56 = 623$), the scene displayed becomes a05 or Packet #36, as shown in Table III(b). On the other hand, if the static-window or IVS approach is applied, 56 slots after the 567th time slot, the scene displayed becomes b05 (Packet #32) or c05 (Packet #30), as shown in Table III(c) and III(d).

These three sets of 18 skiing scenes are exhibited in Figs. 15–17. First of all, comparing the gesture of the skier between a10 and a11, and a12 and a13 in Figs. 16 and 17, we have observed that synchronization-less approach causes noticeable playout discontinuity. Second, focusing on the landing of the skier from a14–a17 in Fig. 17, we reveal a severe playout pause. Third, despite the superior playout at the beginning of the scenes from b01–b03 as shown in Fig. 15, the static-window approach still yields unacceptable playout discontinuity toward the end of scenes (b15–b17 in Fig. 17). Finally, compared to both approaches, IVS achieves superior playout for the entire series of scenes. Notice that even though the playout has been gradually delayed in IVS, the playout of c18 (Packet #53) is synchronized with that of a18 and b18. These observations justify that IVS achieves high quality playout with inevitable but acceptable delays.

VII. CONCLUSION

In this paper, we have proposed an NN-based intravideo synchronization mechanism, i.e., IVS. The IVS is composed of an NN traffic predictor, an NN window determinator, and a window-based playout smoothing algorithm. The NN traffic predictor employs an on-line-trained BPNN to predict the mean busy period and mean idle period of any mixture of IBP traffic distributions. The NN window determinator then determines the corresponding optimal window achieving a maximum of Q_2 value, i.e., the ratio of mean playout throughput to variance of discontinuity, by means of an off-line-trained BPNN. The window-based playout smoothing algorithm then dynamically adopts various playout rates according to the window and the number of packets in the buffer. Finally, we have shown that via simulation results and live video scenes,

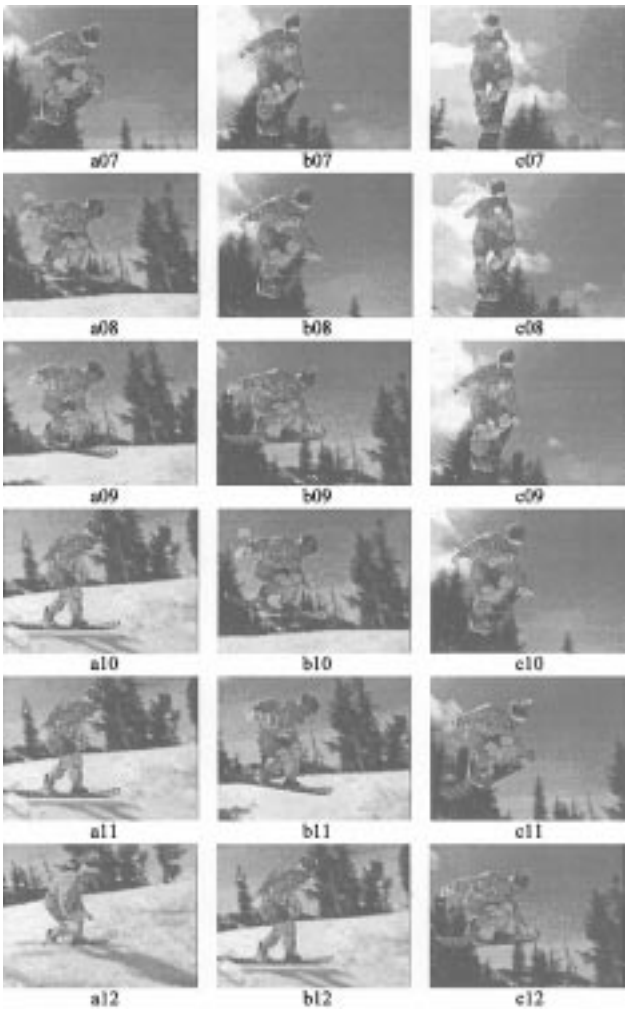

Fig. 16. Playout of snow-skiing scenes based on synchronizationless (a07–a12), static-window-based (b07–b12), and IVS (c07–c12) approaches.

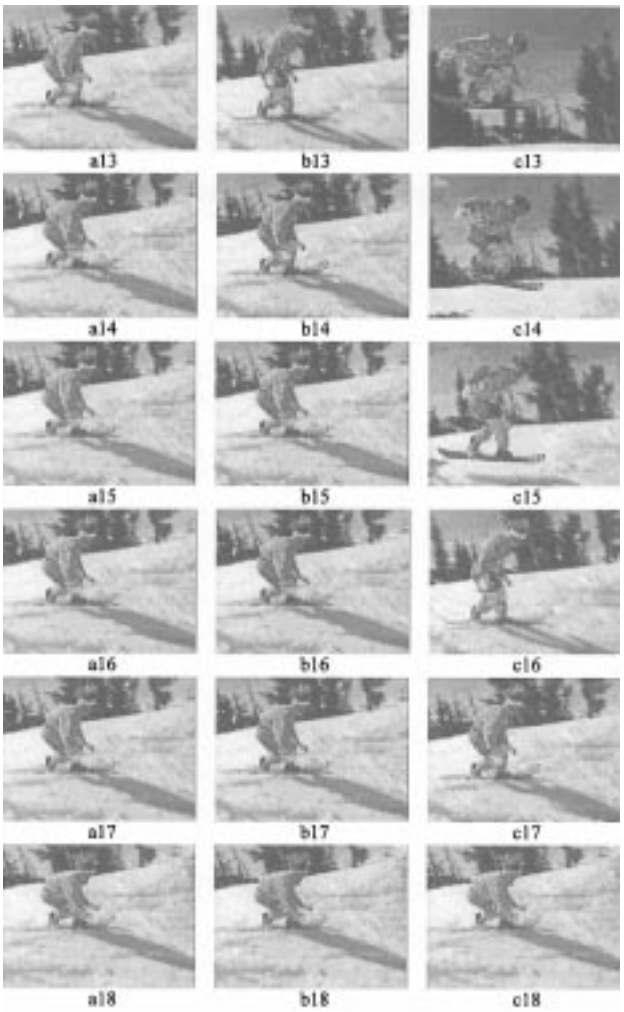

Fig. 17. Playout of snow-skiing scenes based on synchronizationless (a13–a18), static-window-based (b13–b18), and IVS (c13–c18) approaches.

compared to two other playout approaches, IVS achieves high-throughput and low-discontinuity playout.

ACKNOWLEDGMENT

The authors would like to express their thanks to anonymous reviewers, the editors, Prof. C. T. Lin, and Prof. H. M. Hang for their valuable comments.

REFERENCES

[1] ISO/IEC 13818-2, MPEG-2—Information Technology—Generic Coding of Moving Pictures and Associated Audio, Part 2: Video, Annex C.
[2] G. Barberis, "Buffer sizing of a packet-voice receiver," *IEEE Trans. Commun.*, vol. COM-29, no. 2, pp. 152–156, Feb. 1981.
[3] J. Bolot, "End-to-end packet delay and loss behavior in the internet," in *Proc. ACM SIGCOMM*, 1993, pp. 289–298.
[4] D. Clark *et al.*, "An analysis of TCP processing overhead," *IEEE Commun. Mag.*, 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," *Commun. ACM*, vol. 34, pp. 305–313, Apr. 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 J. Select. Areas Commun.*, vol. 9, no. 3, pp. 394–401, 1991.
[8] H. Kanakia, P. Mishra, and A. Reibman, "An adaptive congestion control scheme for real time packet video transport," *IEEE/ACM Trans. Networking*, vol. 3, no. 6, pp. 671–682, Dec. 1995.
[9] M. Yuang, S. Liang, Y. Chen, and C. Shen, "Dynamic video playout smoothing method for multimedia applications," in *Proc. IEEE ICC'96.*
[10] D. McDysan and D. Spohn, *ATM—Theory and Applications.* New York: McGraw-Hill, 1995.
[11] T. Little and A. Ghafoor, "Multimedia synchronization protocols for broadband integrated services," *IEEE J. Select. Areas Commun.*, vol. 9, no. 9, pp. 1368–1382, Dec. 1991.
[12] W. Montgomery, "Techniques for packet voice synchronization," *IEEE J. Select. Areas Commun.*, vol. SAC-1, no. 6, pp. 1022–1028, Dec. 1983.
[13] W. Naylor and L. Kleinrock, "Stream traffic communication in packet switched networks: Destination buffering considerations," *IEEE Trans. Commun.*, vol. COM-30, no. 12, pp. 2527–2534, Dec. 1982.
[14] C. Nicolaou, "An architecture for real-time multimedia communication systems," *IEEE J. Select. Areas Commun.*, vol. 8, no. 3, pp. 391–400, Apr. 1990.
[15] P. Pancha and M. Zarki, "MPEG coding for variable bit rate video transmission," *IEEE Commun. Mag.*, vol. 32, no. 5, pp. 54–66, May 1994.
[16] R. Onvural, *Asynchronous Transfer Mode Networks—Performance Issues*, 2nd ed. Norwood, MA: Artech, 1995.
[17] S. Ramanathan and P. Rangan, "Feedback techniques for intra-media continuity and inter-media synchronization in distributed multimedia systems," *The Computer J.*, 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," in *Proc. IEEE INFOCOM*, 1994.
[19] P. Rangan, S. Kumar, and S. Rajan, "Continuity and synchronization in MPEG," *IEEE J. Select. Areas Commun.*, vol. 14, no. 1, pp. 52–60, 1996.

[20] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," Internet Draft, Nov. 20, 1995.
[21] A. Tarraf, I. Habib, and T. Saadawi, "Intelligent traffic control for ATM broadband networks," *IEEE Commun. Mag.*, vol. 33, no. 10, pp. 76–82, 1995.
[22] A. Tarraf and I. Habib, "A novel neural network traffic enforcement mechanism for ATM networks," *IEEE J. Select. Areas Commun.*, vol. 12, no. 6, pp. 1088–1096, 1994.
[23] M. Yuang, J. Liu, and C. Shay, "BATS: A high-performance transport system for broadband applications," *Proc. Local Computer Networks*, 1994.

Maria C. Yuang (M'91) received the B.S. degree in applied mathematics from the National Chiao Tung University, Taiwan, R.O.C., in 1978, the M.S. degree in computer science from the University of Maryland, College Park, MD, in 1981, and the Ph.D. degree in electrical engineering and computer science from the Polytechnic University, Brooklyn, NY, in 1989.

From 1981 to 1990, she was with AT&T Bell Laboratories and Bell Communications Research (Bellcore), where she was a Member of the Technical Staff working on high-speed networking and protocol engineering. She has been an Associate Professor, Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., since 1990. Her current research interests include high speed networking, multimedia communications, performance modeling and analysis, and ATM network management.

Po L. Tien was born in Taiwan, R.O.C., in 1969. He received the B.S. degree in applied mathematics and the M.S. degree in computer and information science from the National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1992 and 1995, respectively, where he is currently a Ph.D. candidate in the Department of Computer Science and Information Engineering.

His current research interests include high speed networking, multimedia communications, and applications of artificial neural networks.

Shih T. Liang was born in Taiwan, R.O.C., in 1968. He received the B.S. degree in computer information science from the Tunghai University, Taiwan, R.O.C., in 1990, and the M.S. and Ph.D. degrees in computer science and information engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1992 and 1996, respectively.

His current research interests include high speed networking, multimedia communications, and performance modeling and analysis.