# Predictive Dynamic Bandwidth Allocation for Efficient Transport of Real-Time VBR Video over ATM

Song Chong, San-qi Li, *Member, IEEE,* and Joydeep Ghosh

***Abstract*— This paper presents a novel approach to dynamic transmission bandwidth allocation for transport of real-time variable-bit-rate video in ATM networks. Video traffic statistics are measured in the frequency domain: The low-frequency signal captures the slow time-variation of consecutive scene changes while the high-frequency signal exhibits the feature of strong frame autocorrelation. Our queueing study indicates that the video transmission bandwidth in a finite-buffer system is essentially characterized by the low-frequency signal. We further observe in typical JPEG/MPEG video sequences that the time scale of video scene changes is in the range of a second or longer, which localizes the low-frequency video signal in a well-defined low-frequency band. Hence, in a network design it is feasible to implement dynamic allocation of video transmission bandwidth using on-line observation and prediction of scene changes. Two prediction schemes are examined: recursive least square method and time delay neural network method. A time delay neural network with low-complexity high-order architecture, called "pi-sigma network," is successfully used to predict scene changes. The overall dynamic bandwidth-allocation scheme presented in this paper is shown to be promising and practically feasible in obtaining efficient transmission of real-time video traffic.**

## I. Introduction

ATM technology offers a great flexibility of transmission bandwidth allocation to accommodate diverse demands of individual connections. A major application in ATM networks is to provide real-time loss-free transmission of variable-bit-rate (VBR) video. A key issue in video transmission design is to find an effective transmission bandwidth for guaranteed quality-of-services (QOS). Consider a single-link finite-buffer system shown in Fig. 1. By stochastic modeling, video traffic can be represented by a stationary random process. The notion of *effective bandwidth*, measured in cells per unit time, is then equivalent to the minimum transmission bandwidth allocated to the input traffic subject to QOS constraints. Limited analytical solutions are available on transmission bandwidth evaluation, usually with simplified input traffic models and under the asymptotic assumption of large buffer size and small loss rate [1], [2]. Finding the effective bandwidth for video is especially difficult for the following two reasons. First, a real VBR video traffic exhibits highly bursty and nonstationary properties that greatly complicate the queueing analysis. Second, the effective bandwidth must be designed to handle the worst-case input scenario in order to avoid buffer blocking or excessive delay. This extreme case is difficult to predict due to its infrequent occurrence and its dependence on individual sources.

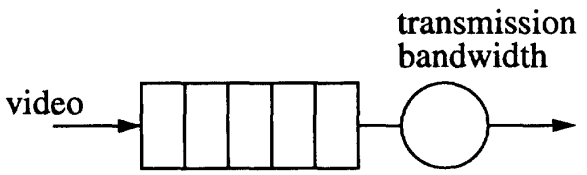



Fig. 1. A single-link finite-buffer system.

In practice, the transmission bandwidth requirement needs to be assessed using on-line traffic measurement, particularly in *live* services. The study in [3] indicates that one of the most important input statistics to measure for queueing analysis is power spectrum. Two basic concepts were discovered in [4] on examining input traffic in the frequency domain. First, the transmission bandwidth in a finite-buffer system is essentially determined by the low-frequency flow of input traffic. Second, the low-frequency flow basically stays intact as it travels through the finite-buffer system.

In this paper, we introduce the concept of dynamic bandwidth allocation, which is a major difference of our approach from most existing techniques. Instead of allocating a static effective bandwidth, we propose to adaptively change the transmission bandwidth using on-line measurement of video demand. For experimental studies, we choose data sequences coded from the movie "Star Wars" by using JPEG/MPEG compression techniques [6], [7]. Such a full-motion entertainment movie represents a class of difficult video services to support in ATM networks. Our statistical analysis shows that video traffic is well-separated into two frequency regions: The low-frequency signal captures the slow time-variation of consecutive scene changes while the high-frequency signal exhibits the feature of strong frame autocorrelation. It is the low-frequency signal that essentially determines the on-line demand of video transmission bandwidth. In other words, once the transmission bandwidth is adaptively changed to accommodate the low-frequency signal, the whole video signal can be transmitted via finite-buffer systems with negligible queueing delay or cell loss.

Manuscript received September 13, 1993; revised June 7, 1994. This work was supported in part by NSF under Grants NCR-9015757 and ECS-9307632, and by Texas Advanced Research Program under Grant TARP-129. This paper was presented in part at IEEE INFOCOM'94, Toronto, Ontario, Canada, June 1994, and at the 1993 International Conference on Artificial Neural Networks in Engineering (ANNIE'93), St. Louis, MO, November 1993.

The authors are with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA.

IEEE Log Number 9406087.

There is a tradeoff between transmission efficiency and protocol processing efficiency in the design of a dynamic bandwidth-allocation scheme. On one hand, a higher transmission efficiency can always be achieved by more frequent adaptation of the bandwidth to the low-frequency signal. On the other hand, since dynamic bandwidth allocation is often performed at the network layer based on global traffic measurement, the frequency of bandwidth adaptation is limited by the network-protocol processing time. Our study indicates that the low-frequency video signal typically stays in a well-defined frequency region, i.e., $\omega < 2\pi$ rad. The corresponding time scale of scene changes will be in the range of a second or longer. Hence, our analysis shows that the video-transmission bandwidth only needs to be adapted at the interval of several hundred milliseconds, which is feasible in a practical network design. Note that the increased bandwidth demand of a video virtual circuit (VC) can be accommodated by various network protocols. For a link-layer protocol, the schemes such as adaptive link capacity partitioning among individual VC's, low-priority cell discarding and blocking (regulating) of non-real-time VC's, can be employed. For a network-layer protocol, a dynamic (or periodic) rerouting scheme can be used to obtain the extra bandwidth without violating the other VC's QOS [4], [5]. A technique of frame-by-frame dynamic bandwidth allocation has been studied recently for the transmission of an MPEG video source [7]. While its on-line bandwidth-estimation scheme is simple, its application to control of network-wide video traffic flow is limited by the frequent adaptation of bandwidth.

A key question then is how to effectively predict abrupt scene changes in the incoming video traffic using on-line low-frequency flow measurement. A better prediction scheme allows a relatively longer lead time to predict any scene changes, which otherwise will cause buffer congestion if the bandwidth is not properly adapted. Two prediction schemes are proposed and compared in this paper. One is based on the recursive least square (RLS) method. From linear filter theory, the RLS adaptive algorithm has the advantage of fast convergence and robustness over the least mean square (LMS) adaptive algorithm [10]. The other scheme takes the artificial neural network (ANN) approach. In particular, we choose a low-complexity high-order architecture, called "pi-sigma network (PSN)," for the construction of a time-delay neural network (TDNN) [12].

Each scheme has its own strength and weakness in prediction design. In contrast to the RLS method, the ANN approach requires a training stage with off-line computation and its solution may not be generally applied. However, the ANN approach has the advantage of much less on-line computation time and no initial transient state for convergence. Also, the PSN-TDNN scheme is a high-degree polynomial prediction method whereas the RLS scheme is a linear prediction method. In our application we find that the PSN-TDNN scheme, trained on one video segment, can generally be applied to other video segments collected from different scenes. Furthermore, the prediction lead time of the PSN-TDNN scheme is found to be longer than that of the RLS scheme for achieving virtually identical queueing performance.

Several examples of using ANN approaches for on-line prediction can be found in the literature, including financial forecasting in stock markets, electric-load forecasting in power networks, traffic prediction in transportation networks and fault prediction in process control [13]–[15]. An ANN approach was also proposed for call-admission control and link-capacity allocation in ATM networks [16], [17]. This paper is the first attempt to use the ANN approach for bandwidth prediction in multimedia communication environment. Our experimental study shows the efficiency and robustness of the PSN-TDNN scheme to predict scene changes in VBR video.

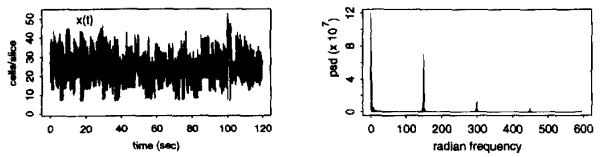
One can implement the dynamic bandwidth allocation in two different operations. In synchronous operation, the bandwidth is periodically adapted at a fixed adaptation interval based on the prediction of video demand. In asynchronous operation, the bandwidth will be adapted if and only if the demand exceeds a preassigned level. The asynchronous operation can significantly reduce the adaptation frequency at the cost of increasing transmission bandwidth.

The paper is organized as follows. In Section II, we introduce the concept of dynamic bandwidth allocation and demonstrate its superior performance over the static allocation. Both RLS and PSN-TDNN prediction schemes are described and their complexities are examined in Section III. Also in Section III, we study the prediction performance of the two shemes in video application with emphasis on the PSN-TDNN scheme. The queueing performance is obtained in Section IV as the two prediction schemes are used for the dynamic bandwidth allocation. The paper is concluded in Section V.

## II. Dynamic Bandwidth Allocation

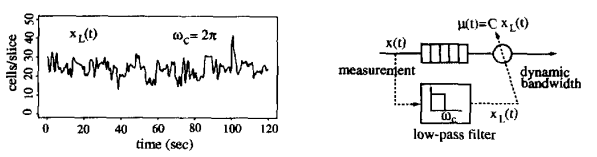
None of the analytical models available today can adequately represent VBR video traffic. Here we choose a full-motion movie, "Star Wars" released from Bellcore [6], as a testbed for our study. It is coded by a JPEG compression technique, i.e., 8×8 discrete cosine transform (DCT) and Huffman coding without motion compensation. The average bit rate is 5.3 Mb/s. The original data are recorded in bytes per slice (1.4 ms) for approximately 2 h. There are 16 lines per slice, 30 slices per frame, and 24 frames per second. The entire data sequence is divided into 60 consecutive pages for approximately two minutes per page. In ATM network application, bytes are converted into cells with each cell consisting of 44 bytes of video signal plus 9 bytes of protocol overhead.

Our interest here is in the dynamic behavior of the video cell sequence. Fig. 2(a) shows a typical 2-min JPEG video traffic (page 56 of "Star Wars") measured in cells per slice, denoted by $x(t)$ at time $t$. The maximum number of cells per slice within this segment is 52.9 and the average is 24.3 cells per slice. Also shown in Fig. 2(b) is the corresponding power spectrum. Two key observations are made about the JPEG video power spectrum. First, the spectral spikes that appear at $24\times 2\pi$ rad and its harmonics represent the frame autocorrelation. Second, the rest of the video power, located in a very low frequency band typically less than $2\pi$ rad, captures the strong correlation of scene changes.




(a) (b)

Fig. 2. (a) A typical 2-min JPEG video traffic where the mean rate and peak rate are, respectively, 24.3 and 52.9 cells/slice (page 56). (b) The corresponding power spectrum.




(a) (b)

Fig. 3. (a) Low-frequency video signal filtered at $\omega_c = 2\pi$ rad, where the mean rate and peak rate are, respectively, 24.3 and 42.0 cells/slice (page 56). (b) Ideal dynamic bandwidth allocation by low-frequency signal.

The recent study in [4] indicates that the effective bandwidth in a finite-buffer system is essentially determined by the low-frequency characteristics of input traffic. Especially for video, the effective bandwidth must be designed to cope with the worst scenario of consecutive scene changes, which is difficult to predict when the connection is initially set up. Note that VBR video traffic possesses highly bursty and nonstationary properties. An effective approach is to dynamically allocate transmission bandwidth using the on-line observation of video scene changes. On applying the above 2-min signal $x(t)$ to a low-pass filter at the cutoff frequency $\omega_c = 2\pi$ rad, Fig. 3(a) shows the filtered signal $x_L(t)$ that characterizes the scene changes. The average and peak input rates of $x_L(t)$ are equal to 24.3 and 42.0 cells/slice, respectively. While a variety of low-pass filters are available, here we choose a class of finite impulse-response filters with a Kaiser window. The time unit in the digital-filtering process is one slice. As one will see, it is this $x_L(t)$ that essentially captures the on-line demand of video transmission bandwidth. For simplicity, we first consider an ideal situation where the transmission bandwidth is instantaneously changed with $x_L(t)$, as described in Fig. 3(b). In other words, the transmission bandwidth at time $t$, denoted by $\mu(t)$, is defined as a function of $x_L(t)$. We further assume

$$\mu(t) = Cx_L(t) \tag{1}$$

where $C$ is a control parameter. From the facts $E[\mu(t)] = CE[x_L(t)]$ and $E[x_L(t)] = E[x(t)]$, one can get $C = \rho^{-1}$ where $\rho$ denotes the average link utilization. Since the bandwidth is instantaneously adapted by the filtered signal, the impact of video scene changes on the queueing process itself has now been removed. Note that we need $C > 1$ for $E[\mu(t)] > E[x(t)]$, i.e., some extra transmission bandwidth is required for the finite buffer to effectively absorb the rest of the high-frequency signal.

Let us now examine the effect of low-frequency video signal on queueing performance. Consider a queueing system with infinite buffer capacity to transmit the above 2-min video using the dynamic bandwidth-allocation scheme at different cutoff frequencies. The reason for using an infinite buffer here is to measure both buffer size requirement and worst-case delay according to different cutoff frequencies. The control parameter $C$ is fixed at 1.25, which is equivalent to the average link utilization $\rho = 0.8$. The simulation results are summarized in Table I, where $\overline{q}$, $\sigma_q$ and $q_{max}$ represent the mean, standard deviation, and maximum of queue length in cell unit, respectively. Note that while the video signal was recorded in the format of number of cells per slice, the queueing process must be evolved in the time unit of cell-transmission slot. In converting the time unit from slice to slot, we consider two extreme scenarios. One is for the "worst scenario" where all the cells generated in each slice are assumed to arrive at the beginning of the first slot in that slice. The other is for the "best scenario" where the cell arrivals in each slice are assumed to be evenly distributed on all the slots of that slice. The queueing solutions in the two extremes provide us the upper and lower bounds of the exact queueing solution. When we choose $\omega_c = 2\pi\tau^{-1}$ where $\tau$ denotes one slice interval, the bandwidth will be adaptively changed with $Cx_L(t)$ in every slice interval. As a result, the queue will always be empty in the best scenario. In the worst scenario, we get $(\overline{q}, \sigma_q, q_{max})$=(9.0, 8.4, 52) due to the batch cell arrival at the first slot of each slice interval. When $\omega_c$ is reduced to $12\times2\pi$, the queueing solutions increase to $(\overline{q}, \sigma_q, q_{max})$=(10.2, 10.0, 129) in the worst scenario, which is contributed by the high-frequency video signal (i.e., the frame autocorrelation). In the extreme, one can take $\omega_c$ =0 to completely eliminate the dynamic bandwidth allocation. Then, due to the strong impact of low-frequency video signal, the queue is drastically increased as measured by $(\overline{q}, \sigma_q, q_{max})$=(257.5, 1208.5, 9610) in the worst scenario. It is obvious that the low-frequency video signal, which captures the slow time-variation of consecutive scene changes, dominates the queueing performance without implementing the bandwidth adaptation.

In practice, the transmission bandwidth cannot be adapted too frequently but is limited by the network-protocol process-

TABLE I
EFFECT OF CUTOFF FREQUENCY ON QUEUEING PERFORMANCE IN THE WORST (BEST) SCENARIO BY DYNAMIC BANDWIDTH ALLOCATION AT $C = 1.25$ (PAGE 56)

| $\omega_c$ (radians) | $\overline{q}$ (cells) | $\sigma_q$ (cells) | $q_{max}$ (cells) |
|---|---|---|---|
| $\tau^{-1} \times 2\pi$ | 9.0 (0) | 8.4 (0) | 52 (0) |
| $12\times2\pi$ | 10.2 (1.0) | 10.0 (4.2) | 129 (100) |
| $1\times2\pi$ | **11.1(2.6)** | **13.1(9.1)** | **224(195)** |
| 0 | 257.5 (247.7) | 1208.5 (1207.5) | 9610 (9580) |

ing time. In our case we choose $\omega_c = 2\pi$ at which the queueing performance is given by $(\overline{q},\ \sigma_q,\ q_{max})$=(11.1, 13.1, 224). For the average rate of the 2-min video, equal to 24.3 cells/slice, the maximum queue length of 224 cells at $C = 1.25$ (or $\rho = 0.8$) is equivalent to the maximum delay of 10 ms, which is expected to be tolerable in video services. Notice that the time-varying scale of the video signal in the low-frequency band $\omega < 2\pi$ is at least 1 s long, which makes the implementation of dynamic bandwidth allocation in a practical network design feasible.

Let us examine the performance improvement of dynamic bandwidth allocation over static allocation. The static allocation assigns a fixed transmission bandwidth to each connection during the entire service period. In order to prevent excessive delay or cell loss, the static bandwidth has to be designed to cope with the worst input scenario. The most conservative static allocation is to reserve bandwidth by the peak input rate, which would lead to the excessive use of transmission capacity with zero buffer size as in the circuit-switched design. In our case, the video signal was originally collected at slice interval. If the bandwidth is statically assigned by the peak input rate of the above 2-min video segment measured at slice interval, the link utilization will be as low as 0.46. The corresponding queueing solutions in the worst scenario are given by $(\overline{q},\ \sigma_q,\ q_{max})$=(5.2, 8.8, 52) (see Table II). The study in [4] indicates that the effective transmission bandwidth in a finite-buffer system is essentially determined by the peak of the properly filtered input rate. Consider the filtered-video input rate $x_L(t)$ at $\omega_c = 2\pi$. By the static allocation, the bandwidth will be assigned by the peak of $x_L(t)$, which leads to the link utilization $\rho = 0.58$ and queueing performance $(\overline{q},\ \sigma_q,\ q_{max})$=(6.9, 9.5, 138). In contrast, when the bandwidth is dynamically adapted with $Cx_L(t)$, where $C = 1.25$, the link utilization can be as high as $\rho = 0.80$ with a moderate increase of buffer capacity. As the results are compared in Table II, the dynamic bandwidth allocation designed at $\omega_c = 2\pi$ can effectively improve the video-transmisson efficiency. Also compared in Fig. 4 are the queue distribution functions of the corresponding three bandwidth-allocation schemes. Note that the improvement with dynamic allocation can be more significant if a longer video segment is chosen. This is because the static allocation scheme needs to assign the bandwidth equal to the filtered peak input rate of the chosen video segment. In practice, it is difficult to identify the filtered peak input rate of each individual video source at the call admission stage, which makes the static allocation scheme hardly implementable. By comparison, the dynamic bandwidth allocation is designed on the basis of on-line traffic measurement. Not only does it effectively improve the transmission efficiency, but also its implementation is highly feasible in practice.

Unlike JPEG compression, the MPEG compression technique exploits temporal (i.e., frame-to-frame) redundancy present in all video sequences [8]. There are three types of frames in an MPEG video depending on motion-compensation schemes: $I$ frame by intraframe encoding, $P$ frame by motion-compensated prediction, and $B$ frame by motion-compensated interpolation. For comparative study, both MPEG and JPEG video sequences are collected from the same 3.5-min segment

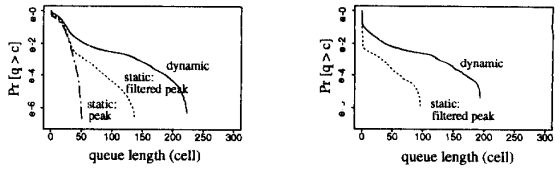



Fig. 4. Queue distributions in static and dynamic bandwidth allocations (page 56). (a) Worst scenario. (b) Best scenario.

TABLE II
QUEUEING PERFORMANCE IN STATIC AND DYNAMIC BANDWIDTH ALLOCATIONS IN THE WORST (BEST) SCENARIO (PAGE 56)

| | Static Allocation | | Dynamic Allocation |
|---|---|---|---|
| | peak rate (at $\omega_c = 2\pi\tau^{-1}$) | filtered peak rate (at $\omega_c = 2\pi$) | (at $\omega_c = 2\pi$) |
| $\rho$ | 0.46 | 0.58 | 0.80 |
| $\overline{q}$ | 5.2 (0) | 6.9 (0.1) | 11.1 (1.9) |
| $\sigma_q$ | 8.8 (0) | 9.5 (1.8) | 13.1 (9.1) |
| $q_{max}$ | 52 (0) | 138 (97) | 224 (195) |

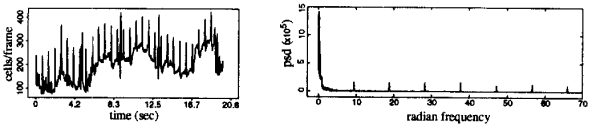



Fig. 5. (a) An MPEG video traffic. (b) The corresponding power spectrum.

(pages 2, 3) of the "Star Wars" [7]. In the MPEG coding, only $I, P$ frames were used since the interpolation for $B$ frames is not appropriate for real-time applications due to its noncausal property. In our experiment, each $I$ frame is followed by 15 consecutive $P$ frames and there are 24 $I$ and $P$ frames/s. The average rate of the MPEG and JPEG sequences is 2.3 Mb/s and 3.5 Mb/s, respectively. Fig. 5(a) shows a part of the MPEG video sequence, which is measured in cells per frame. As one can see, the burst of $I$ frames occurs periodically at the interval of 16 frames, which in some degree will cause the queueing-performance degradation. The power spectrum of the MPEG video sequence is also plotted in Fig. 5(b). The spectral spikes appeared at harmonic radian frequencies $9.4 \times k$ (i.e., $\frac{24}{16} \times 2\pi \times k$), $k = 1, 2, \cdots$, are due to the strong autocorrelation of periodic $I$ frames. In contrast, the JPEG video contains negligible powers in the frequency band $2\pi < \omega < 24 \times 2\pi$, as shown in Fig. 2(b).

Let us now examine the effect of the MPEG and JPEG compression schemes on queueing performance when the dynamic bandwidth allocation is applied. The control parameter $C$ is set at 1.25 in both MPEG and JEPG cases. The results are

TABLE III
COMPARISON OF QUEUEING PERFORMANCES OF MPEG AND JPEG VIDEOS AS A FUNCTION OF $\omega_c$ BY THE SAME DYNAMIC BANDWIDTH ALLOCATION AT $C = 1.25$

| | MPEG | | | JPEG | | |
|---|---|---|---|---|---|---|
| $\omega_c$ | $\bar{q}$ | $\sigma_q$ | $q_{max}$ | $\bar{q}$ | $\sigma_q$ | $q_{max}$ |
| $\tau^{-1} \times 2\pi$ | 0 | 0 | 0 | 0 | 0 | 0 |
| $1 \times 2\pi$ | 13.7 | 43.4 | 497 | 0.52 | 7.9 | 237 |
| 0 | 5487 | 9848 | 41020 | 5103 | 10295 | 36335 |

summarized in Table III. In the queueing simulation, the cell arrivals in each frame are assumed to be evenly distributed among the slots of that frame. The queueing performances are compared at three selected cutoff frequencies. When $\omega_c = 2\pi\tau^{-1}$, where $\tau$ denotes one frame interval, the queue is always empty in both MPEG and JPEG cases because the video sequences are collected in frame units. When $\omega_c = 2\pi$, we get ($\bar{q}$, $\sigma_q$, $q_{max}$) equal to (13.7, 43.4, 497) for the MPEG video and (0.52, 7.9, 237) for the JEPG video. Obviously, the $I$ frame burstiness (or powers at $9.4 \times k, k = 1, 2, \cdots$) in some degree has caused the queueing-performance degradation in the MPEG video. The maximum queue length increases from 237 in the JPEG case to 497 in the MPEG case. With such a moderate increase of buffer-capacity requirement, the proposed dynamic bandwidth allocation at cutoff frequency $\omega_c = 2\pi$ should also be applicable to the MPEG video. At $\omega_c = 0$, i.e., without dynamic allocation, the queueing performances in both MPEG and JPEG cases are drastically deteriorated due to the strong impact of low-frequency video signal, as shown in Table III.

In principle, the same dynamic bandwidth-allocation scheme can be applied to MPEG video transmission with moderate increase of buffer capacity. On the other hand, the motion-compensation technique used in the MPEG compression will greatly improve transmission efficiency. In the above example, the average transmission rate is each case is given by multiplying the average video rate by $C$. Therefore, the average transmission rate is equal to 1.25×2.3 Mb/s for the MPEG video whereas it is 1.25×3.5 Mb/s for the JPEG video. The following sections focus on JPEG video transmission since only few MPEG video sequences are available.

## III. BANDWIDTH PREDICTION

In the above section we have assumed that the transmission bandwidth $\mu(t)$ is instantaneously adapted by the filtered input rate $x_L(t)$. In reality, the bandwidth can only be adapted intermittently using on-line observation and prediction of $x_L(t)$; the adaptation interval cannot be too short since it is limited by the protocol processing time to allocate the desired bandwidth. Fig. 6 describes a realistic dynamic allocation scheme. Let the filtered input signal $x_L(t)$ be sampled at time unit $\Delta$. Denote the sampled signal by $x_L(n)$ at the $n$th $\Delta$ unit. The bandwidth is periodically adapted at the interval of $M\Delta$. There will be $D\Delta$ lead time for computation of the prediction algorithm and protocol processing of the bandwidth allocation. That is, if the prediction starts at the $n$th unit, the bandwidth will be adapted at the $(n + D)$th unit. Consequently, the next bandwidth adaptation will occur at the $(n + D + M)$th unit. The prediction of the input rate at each consecutive unit in the adaptation interval, denoted by $\hat{x}_L(n + D + l)$ at $l = 0, 1, \cdots, M$, is made on the basis of $(N + 1)$ consecutive observations collected at the $n$th unit. In other words, the predictions $\{\hat{x}_L(n + D), \hat{x}_L(n + D + 1), \cdots, \hat{x}_L(n + D + M)\}$ are made on the basis of $\{x_L(n - N), x_L(n - N + 1), \cdots, x_L(n)\}$. To cope with the worst input rate in the adaptation interval, the transmission bandwidth is assigned by

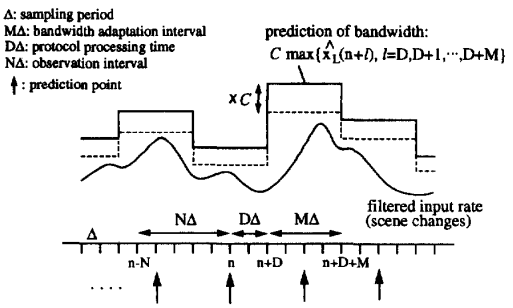



Fig. 6. Bandwidth prediction in a VBR video connection.

$$C \max \{\hat{x}_L(n+D), \hat{x}_L(n+D+1), \cdots, \hat{x}_L(n+D+M)\}. \quad (2)$$

The selection of $\Delta$ is dependent on the time-variation scale of $x_L(t)$. In our application, $x_L(t)$ is defined at $\omega_c = 2\pi$ to represent the video scene changes as indicated in Figs. 2(b), 3(a). The corresponding time-varying scale of $x_L(t)$ is at least 1 s long. Here we take the oversampling of $x_L(t)$ at $\Delta = 0.14$ s in order to capture any abrupt scene changes. Two prediction schemes are introduced in the following subsections with comparison of prediction performance and computational complexity.

### A. RLS-Based Prediction

We use the RLS algorithm to design an adaptive filter for traffic prediction [10]. The rate of convergence of the RLS algorithm is typically an order of magnitude faster than that of the LMS algorithm at the expense of increased computation. In contrast to the LMS algorithm, the rate of convergence of the RLS algorithm is insensitive to variations in the eigenvalue spread, defined as the ratio of the maximum-to-minimum eigenvalues of the correlation matrix of the input vector. The RLS algorithm also has to some extent the capability to track statistical variations in a nonstationary environment by setting the exponential forgetting factor less than unity [10]. As a result, here we choose the RLS algorithm for the on-line bandwidth prediction of video traffic.

Since each bandwidth adaptation requires computation of the predictions $\{\hat{x}_L(n+D), \hat{x}_L(n+D+1), \cdots, \hat{x}_L(n+D+M)\}$, the so-called *indirect prediction* approach is used to avoid redundant computation [9]. That is, instead of directly constructing $(M + 1)$ RLS prediction filters, we construct a single RLS filter to perform parameter estimation of a given autoregressive (AR) model of the time series. The $(M + 1)$ predictions are then obtained by converting the model into

the required predictor format. This is further described in the following.

Assume that the sampled time series of the filtered video is modeled by a deterministic AR process of order $N$, defined by

$$x_L(n) = \theta_1^* x_L(n-1) + \theta_2^* x_L(n-2) + \cdots + \theta_N^* x_L(n-N). \quad (3)$$

What we need to solve is the estimation of the unknown parameter vector $\vec{\theta}^* = [\theta_1^*, \theta_2^*, \cdots, \theta_N^*]^T$. At the $n$th time unit, let us define the input vector $\vec{u}(n) = [x_L(n-1), x_L(n-2), \cdots, x_L(n-N)]^T$, the desired output $d(n) = x_L(n)$, and the estimated parameter vector $\hat{\vec{\theta}}(n) = [\hat{\theta}_1(n), \hat{\theta}_2(n), \cdots, \hat{\theta}_N(n)]^T$. The on-line RLS estimation of $\vec{\theta}^*$ is then recursively expressed by

$$\vec{k}(n) = \frac{\lambda^{-1}\boldsymbol{P}(n-1)\vec{u}(n)}{1 + \lambda^{-1}\vec{u}^T(n)\boldsymbol{P}(n-1)\vec{u}(n)} \quad (4)$$

$$\hat{\vec{\theta}}(n) = \hat{\vec{\theta}}(n-1) + \vec{k}(n)(d(n) - \hat{\vec{\theta}}^T(n-1)\vec{u}(n)) \quad (5)$$

$$\boldsymbol{P}(n) = \lambda^{-1}\boldsymbol{P}(n-1) - \lambda^{-1}\vec{k}(n)\vec{u}^T(n)\boldsymbol{P}(n-1) \quad (6)$$

where $\lambda$ is the forgetting factor, $\boldsymbol{P}(n)$ denotes the inverse of the input correlation matrix, and $\vec{k}(n)$ represents the gain vector. For the initial condition at $n = 0$ we have $\hat{\vec{\theta}}(0) = \vec{0}$ and $\boldsymbol{P}(0) = \delta^{-1}\boldsymbol{I}$, where $\delta$ is a small positive constant and $\boldsymbol{I}$ represents an identity matrix. From the given AR model in (3), then each $l$−step ahead prediction, $l = D, D+1, \cdots, D+M$, is recursively obtained by

$$\hat{x}_L(n+l) = \hat{\vec{\theta}}^T(n)\vec{r}(n, l) \quad (7)$$

where $\vec{r}(n, l) = [\hat{x}_L(n+l-1), \hat{x}_L(n+l-2), \cdots, \hat{x}_L(n+1), x_L(n), x_L(n-1), \cdots, x_L(n+l-N)]^T$.

Let $b(n)$ represent the transmission bandwidth required during the adaptation interval $[(n+D)\Delta, (n+D+M)\Delta)$. Its prediction, denoted by $\hat{b}(n)$, is then given by

$$\hat{b}(n) = C \max\{\hat{x}_L(n+l),\ l = D, D+1, \cdots, D+M\}. \quad (8)$$

Using the *round-to-largest* rule, the real value of $\hat{b}(n)$ is quantized into an integer in cell unit. Such an adaptive bandwidth assignment is called *synchronous dynamic allocation* since it is adapted periodically at a fixed interval $M\Delta$. For low-complexity network management of bandwidth adaptation, one can also introduce an *asynchronous dynamic allocation* scheme, defined by

$$\hat{b}(n) = C \max\{\phi, \hat{x}_L(n+l),\ l = D, D+1, \cdots, D+M\} \quad (9)$$

where $\phi$ denotes a preassigned *nominal bandwidth.* Since the bandwidth is adapted if and only if the video prediction exceeds the nominal bandwidth, the asynchronous operation can significantly reduce the bandwidth adaptation frequency at the expense of increased transmission bandwidth. As an example, we set $\phi = E[x(t)] + (\mathrm{Var}[x(t)])^{1/2}$ in this paper. If such statistics are not available *a priori,* one can choose a reasonable value for $\phi$ with a tradeoff between transmission efficiency and protocol processing efficiency.

### *B. TDNN-Based Prediction*

As an alternative approach we introduce a TDNN-based prediction scheme [18], [19]. ANN's have adaptation capability that can accommodate nonstationarity. ANN's have generalization capability that makes them flexible and robust when faced with new and/or noisy data patterns. Once the training is completed, an ANN can be computationally inexpensive even if it continues to adapt on-line. Recently a high-order neural network has been developed [12], which approximates the input-output relationship by a high-degree polynomial while avoiding exponentially increasing computational and memory cost that had affected ordinary high-order nets [21]. This architecture, called the pi-sigma network (PSN), is selected as the basis of our TDNN prediction scheme.

Fig. 7 shows a TDNN based on $L$th-degree PSN with $(N+1)$ inputs and $(M+1)$ outputs. In conventional TDNN's, the architecture above tapped delay line in Fig. 7 is given by multilayered perceptron networks (MLP's) [18]–[20], which suffer slow training and relatively expensive on-line computation. The PSN architecture consists of a single hidden layer of $L \times (M+1)$ linear summing nits ($L$ summing units per output) and an output layer of $(M+1)$ product units. These product units make it possible to incorporate the capabilities of high-order networks while greatly reducing network complexity. The term "pi-sigma" comes from the fact that these networks use products of sums of input components instead of sums of products, as in ordinary high-order networks. The output from each product unit passes through a sigmoid activation function defined by $\sigma(x) = 1/(1+e^{-x})$. Unlike in MLP, the weights from the hidden layer to the outputs are fixed at 1. This property contributes to reducing training time substantially. The bias input is also fixed at 1. The purpose of this network is to find an approximate $L$th-degree relationship between the inputs $x_L(n), x_L(n-1), \cdots, x_L(n-N)$ and the desired outputs $x_L(n+D), x_L(n+D+1), \cdots, x_L(n+D+M)$. For convenience, we define input vector, desired output vector, and estimated output vector by $\vec{\xi}(n) = [\xi_0(n), \xi_1(n), \xi_2(n), \cdots, \xi_{N+1}(n)]^T$, $\vec{d}(n) = [d_0(n), d_1(n), \cdots, d_M(n)]^T$ and $\vec{y}(n) = [y_0(n), y_1(n), \cdots, y_M(n)]^T$. Correspondingly

$$\begin{aligned}
\vec{\xi}(n) &= [1, x_L(n), x_L(n-1), \cdots, x_L(n-N)]^T \\
\vec{d}(n) &= [x_L(n+D), x_L(n+D+1), \cdots, \\
&\qquad x_L(n+D+M)]^T \\
\vec{y}(n) &= [\vec{x}_L(n+D), \hat{x}_L(n+D+1), \cdots, \\
&\qquad x_L(n+D+M)]^T. \quad (10)
\end{aligned}$$

Note that $\vec{\xi}(n)$ is augmented by a bias input and $\vec{y}(n)$ is from the array of sigmoid functions. By training, the relationship will be stored through the network in the form of connectivity strengths or weights. Let $\vec{w}_j^l = [w_{0_j}^l, w_{1_j}^l, \cdots, w_{N+1_j}^l]^T$ be the weight vector for the $j$th hidden unit of the $l$th output where $l = 0, 1, \cdots, M$ and $j = 0, 1, \cdots, L-1$. The $l$th output $y_l(n)$

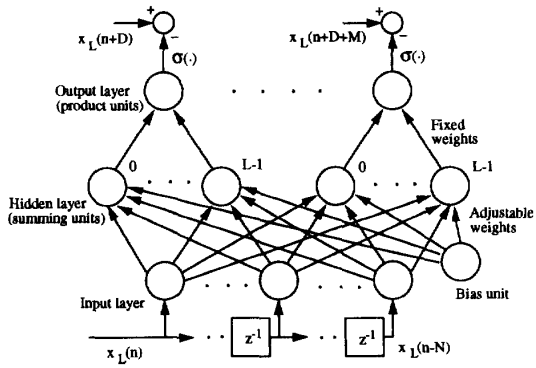



Fig. 7. An $L$th-degree PSN-TDNN.

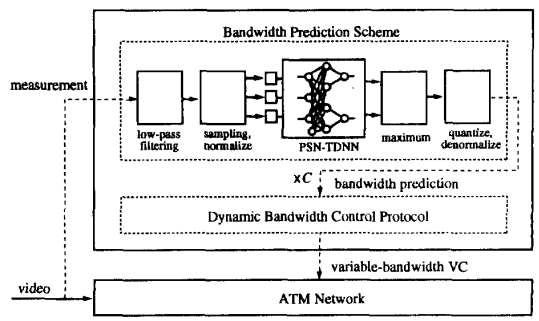



Fig. 8. Overall structure of dynamic bandwidth-allocation scheme using on-line bandwidth prediction.

at time $n$ is then expressed by

$$y_l(n) = \sigma\left[\prod_{j=0}^{L-1} \vec{w}_j^{lT}\vec{\xi}(n)\right] = \sigma\left[\prod_{j=0}^{L-1}\left(\sum_{k=1}^{N+1} w_{k_j}^l \xi_k(n) + w_{0_j}^l\right)\right]. \quad (11)$$

Note that this regression model of the PSN-TDNN is more general than AR model in that it realizes an $L$th-degree polynomial mapping from input to output. Furthermore, unlike in ordinary high-order nets, the high-degree approximation in a factorized form as in (11) greatly reduces the computational complexity.

The learning algorithm for the PSN is based on gradient descent on the estimated mean square error (MSE) surface in weight space. The MSE objective is given by

$$J_l = \frac{1}{p}\sum_{n=0}^{p-1}[d_l(n) - y_l(n)]^2, \qquad l = 0, 1, \cdots, M \quad (12)$$

where $p$ denotes the number of training patterns. By taking LMS-type approach, the weight update rule is given by

$$\delta\vec{w}_s^l(n) = \eta[d_l(n) - y_l(n)][y_l(n)]'\left[\prod_{j\neq s}\vec{w}_j^{lT}(n)\vec{\xi}(n)\right]\vec{\xi}(n) \quad (13)$$

for $l = 0, 1, \cdots, M$ where $[y_l(n)]'$ is the first derivative of the sigmoid function and $\eta$ is the learning rate. Here we have adopted an *asynchronous update rule,* which updates only a partial set of weights at a time instead of the overall weights [12]. There are total $L$ sets of weights, defined by $\{\vec{w}_j^l(n), l = 0, 1, \cdots, M\}$ at $j = 0, 1, \cdots, L-1$, each of which is associated with a hidden unit. Only one set, $\{\vec{w}_s^l(n), l = 0, 1, \cdots, M\}$, is chosen at time $n$ and updated by (13); the rest of the sets remain unchanged. This procedure is repeated in an asynchronous manner.

As in the RLS-based prediction, the transmission bandwidth required during the next adaptation interval $[(n+D)\Delta, (n+D+M)\Delta)$ is predicted by

$$\hat{b}(n) = C\max\{y_l(n), l = 0, 1, \cdots, M\}. \quad (14)$$

Also, the asynchronous allocation scheme is given by

$$\hat{b}(n) = C\max\{\phi, y_l(n), L = 0, 1, \cdots, M\}. \quad (15)$$

The overall structure of dynamic bandwidth-allocation scheme using on-line PSN-TDNN-based prediction is shown in Fig. 8. In practice, such a system can be implemented at the network access point to continuously provide the prediction of bandwidth changes to the network. The function of *dynamic bandwidth control protocol* is to allocate the predicted bandwidth to each VC, which can be implemented at different protocol layers. At the link layer, the dynamic allocation of predicted bandwidth can be achieved by adaptive partitioning of link capacity among individual VC's, low-priority cell discarding, or blocking (regulating) of non-real-time traffic streams. The adaptive-link-capacity partitioning scheme is to take advantage of *statistical gain* of multiplexed connections. Also, for hierarchically coded video connections [7], a low-priority cell-discarding scheme can be effectively employed. At the network layer, the dynamic allocation of predicted bandwidth can be achieved by dynamic (or periodic) rerouting schemes as proposed and implemented in [4], [5]. When a video VC request more bandwidth, by selectively rerouting some of the active VC's to the other paths being underutilized, the network can provide the extra bandwidth to the VC without violating the other VC's QOS. The design and evaluation of such dynamic-bandwidth-control protocols are beyond the scope of this paper.

### C. Computational Complexity

One disadvantage of the RLS prediction scheme is its computational complexity. The parameter estimation in (4)–(6) requires $2N^2 + 7N + 5$ multiplications and $N^2 + 4N + 3$ divisions per iteration. The bandwidth prediction in (7) needs $N\times(D+M)$ multiplications. Since the parameter estimation is computed at every $\Delta$ interval while the bandwidth prediction is computed at every $M\Delta$, the total complexity per $M\Delta$ is $M \times (2N^2 + 7N + 5) + N \times (D + M)$ multiplications and $M \times (N^2 + 4N + 3)$ divisions. The computational complexity can be reduced by using the so-called fast transversal filter (FTF) algorithm [11]. This algorithm attains the RLS solution with the same convergence properties as in the RLS algorithm

but at a computational cost that is competitive with the LMS algorithm. As a result, the complexity in parameter estimation is reduced to $7N + 12$ multiplcations plus 4 divisions per iteration; the total complexity per $M\Delta$ then becomes $M \times (7N+12)+N \times (D+M)$ multiplications and $4M$ divisions. In comparison, once it is trained, the PSN-TDNN scheme only requires $(M + 1) \times (N + 2) \times L$ multiplications per $M\Delta$. Although special computation for the sigmoid function is required, in practice the sigmoid function is usually replaced by a linear saturator or a lookup table.

### D. Prediction Performance

In the experimental study, we choose $\omega_c = 2\pi$ for the low-frequency signal $x_L(t)$ to represent the video scene changes according to the power spectral distribution in Fig. 2(b). The RLS scheme is designed with $(N, D, M) = (5, 1, 4)$ and the PSN-TDNN scheme is with $(N, D, M, L) = (5, 2, 4, 2)$, both of which are found to provide adequate prediction of scene changes in queueing performance. The prediction lead time is defined by $(D + M)\Delta$, which at $\Delta = 0.14$ s is equal to 0.7 s for the RLS scheme and 0.84 s for the PSN-TDNN scheme. The longer the lead time, the better the prediction scheme for achieving identical performance. Since the time-varying scale of the low-frequency video signal is at least one second long, we design the bandwidth-adaptation interval at $M\Delta = 0.56$ s, close to the Nyquist sampling interval.

In the initial stage of our experimental study, we also used an MLP-based TDNN for the prediction, but no significant performance advantage was observed over the PSN-based TDNN. In contrast with PSN-TDNN, MLP-TDNN suffers higher computational complexity and subsequent longer training period. For the training of the PSN-TDNN, we used a 2-min JPEG video segment (page 56) in Fig. 3(a), which was filtered at $\omega_c = 2\pi$. By scanning the filtered video segment along time, we collected 208 training examples. After adding a $2\Delta-$long offset, we again scanned the segment, thereby updating the 208 examples. By repeating this procedure three times, we obtained a total of 624 examples. Due to the dynamic range of the sigmoid function, the input data was normalized into [0, 1]. In the design of the PSN-TDNN, having the observation interval $N\Delta > 5\Delta$ or the degree $L > 2$ was found to simply add computational complexity with no significant performance improvement. It can be interpreted in that, in statistical estimation, increasing complexity of the model over some optimal point may degrade performance due to the *bias/variance dilemma* [22]. The learning rate was tuned at $\eta = 0.15$. The training was carried out on a SPARC-10 workstation for 24 min (CPU time) through 5000 epochs. In the design of the RLS scheme, the observation interval was also tuned at $N = 5$ and the forgetting factor was set at $\lambda = 0.9$.

The performance is measured by the prediction-error statistics. In our application, the transmission bandwidth is adapted at every $M\Delta$ interval. For $t \in [(n + D)\Delta, (n + D + M)\Delta)$, let $\hat{x}_{max}(t)$ be

$$\hat{x}_{\max}(t) = \max\{\hat{x}_L(n+l),\ l = D, D+1, \cdots, D+M\}. \quad (16)$$

TABLE IV
PREDICTION ERROR PERFORMANCE ON TRAINING SET (PAGE 56) AT $M\Delta + 0.56$ S WHERE $\bar{\epsilon}$, $\sigma_\epsilon$, $\epsilon_{max}$, AND $\epsilon_{min}$ ARE, RESPECTIVELY, THE MEAN, STANDARD DEVIATION, MAXIMUM, AND MINIMUM OF $\epsilon(t)$

| | PSN-TDNN | RLS | |
|---|---|---|---|
| | $(D+M)\Delta$=0.84 | $(D+M)\Delta$=0.7 | $(D+M)\Delta$=0.84 |
| $\bar{\epsilon}$ | 0.044 | 0.051 | 0.050 |
| $\sigma_\epsilon$ | 0.071 | 0.092 | 0.114 |
| $\epsilon_{max}$ | 0.541 | 0.658 | 0.740 |
| $\epsilon_{min}$ | -0.145 | -0.144 | -0.256 |

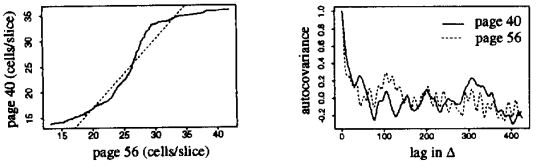
The relative prediction error at time $t$ is then defined by

$$\epsilon(t) = [\hat{x}_{\max}(t) - x_L(t)]/x_L(t).$$

Positive and negative values of $\epsilon(t)$ correspond to the overestimation and underestimation of the low-frequency video signal. Note that it is the underestimation that may cause the buffer congestion, while the overestimation can only result in the underutilization of the transmission bandwidth. This is why in (2) we have taken the maximum of the predictions in each adaptation interval for the bandwidth allocation.

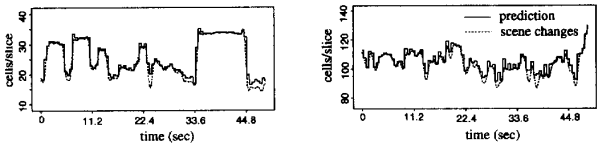
Table IV shows the prediction-error performance of the PSN-TDNN on the training-video segment (page 56). The performance on the same segment achieved by the RLS scheme is also shown. While the lead time of the PSN-TDNN scheme is 0.14 s longer than that of the RLS scheme, the performance is basically identical. In the RLS prediction-error statistics, initial transient performance for convergence was excluded. For comparison, also listed in Table IV is the performance of the RLS scheme at the lead time equal to 0.84 s. Obviously, this design of RLS scheme degrades the performance. It is observed that the RLS scheme has larger prediction-error variance than the PSN-TDNN scheme.

Let us now examine the general applicability of the PSN-TDNN scheme, trained by one video segment, to other video segments. Six 2-min filtered-video segments (pages 39–41, 55–57) were chosen as testing sets. Pages 39–41 were arbitrarily selected to represent statistically different scenes. For example, Fig. 9(a) shows the quantile-quantile (Q-Q) plot of page 40 and page 56 (the training set) at $\omega_c = 2\pi$. Since the Q-Q plot is largely deviated from the linear reference line, the probability distribution of page 40 is different from that of page 56. Also displayed in Fig. 9(b) are the autocovariance functions of the two segments, which are also quite different. Hence, the scene statistics of page 40 must be different from that of the training set. It is interesting to find that the PSN-TDNN scheme, trained on page 56, works very well on the other pages. Fig. 10(a) shows a part of the prediction curve to track the scene changes on page 40. Similar performance is observed on the other pages. Listed in Table V are the error statistics of the PSN-TDNN scheme on pages 40 and 56. The error statistics of the RLS scheme on page 40 are also included for comparson. Furthermore, in Table VI we compare the prediction performance of the two schemes for the overall 12 min. The generalization capability of the PSN-TDNN scheme is also tested on the prediction of a 2-min multiplexed video segment. Here we take the summation of five 2-min




(a) (b)

Fig. 9. (a) Q-Q plot of page 40 and page 56 (training set) at $\omega_c = 2\pi$. (b) Autocovariance functions.




(a) (b)

Fig. 10. Testing of PSN-TDNN. (a) On video segment with statistically different scenes (page 40). (b) On video multiplexed by pages 55–59.

TABLE V
PREDICTION ERROR PERFORMANCE ON A TESTING SET (PAGE 40) AT $M\Delta = 0.56$ S

| | on page 56 | on page 40 | |
|---|---|---|---|
| | PSN-TDNN | PSN-TDNN | RLS |
| | $(D+M)\Delta$=0.84 | $(D+M)\Delta$=0.84 | $(D+M)\Delta$=0.7 |
| $\bar{\epsilon}$ | 0.044 | 0.045 | 0.048 |
| $\sigma_\epsilon$ | 0.071 | 0.086 | 0.128 |
| $\epsilon_{max}$ | 0.541 | 0.597 | 1.875 |
| $\epsilon_{min}$ | -0.145 | -0.152 | -0.181 |

TABLE VI
TWELVE MINUTE PREDICTION-ERROR PERFORMANCE (ON PAGES 39–41, 55–57) AT $M\Delta = 0.56$ S

| | PSN-TDNN | RLS |
|---|---|---|
| | $(D+M)\Delta$=0.84 | $(D+M)\Delta$=0.7 |
| $\bar{\epsilon}$ | 0.044 | 0.045 |
| $\sigma_\epsilon$ | 0.072 | 0.098 |
| $\epsilon_{max}$ | 0.597 | 1.875 |
| $\epsilon_{min}$ | -0.171 | -0.188 |

video segments (pages 55–59) to represent the statistical multiplexing of five video sources. Similar performance is achieved as plotted in Fig. 10(b). In summary, we argue that the PSN-TDNN scheme, properly trained on a 2-min video segment, can be directly used to a certain extent on other video segments (which consist of statistically quite different scenes).

According to the analysis in Section III.C, the complexity of the RLS scheme is 385 multiplications and 192 divisions while the complexity of the PSN-TDNN scheme is only 70 multiplications per $4\Delta$. When the FTF algorithm is applied [11], the complexity of the RLS scheme is reduced to 213 multiplications and 16 divisions. In contrast, the on-line computation of the PSN-TDNN scheme is less than one-eighth of the RLS scheme.

The study in this section indicates that, in our application of video bandwidth prediction, the PSN-TDNN scheme is superior to the RLS scheme in terms of both prediction performance and computational complexity. In the next section we evaluate the video queueing performance when the two schemes are applied to the dynamic adaptation of transmission bandwidth.

## IV. PERFORMANCE EVALUATION OF DYNAMIC BANDWIDTH ALLOCATION

First, let us design a zero-loss transmission system to deliver a 12-min video segment (pages 39–41, 55–57). Assume that the buffer size should never exceed 300 cells. As described in Section II, the transmission bandwidth is essentially captured by the video scene changes located in a well-defined low-frequency band. We choose $\omega_c = 2\pi$ for the filtered video signal $x_L(t)$ to capture the scene changes. The transmission bandwidth is then allocated through the observation and prediction of $x_L(t)$, either dynamically or statically. In the static allocation, the bandwidth is assigned by $\max_t x_L(t)$, which is the maximum of $x_L(t)$ in the entire 12-min period. In practice, however, $\max_t x_L(t)$ is unknown. In the ideal dynamic allocation, the bandwidth is directly assigned by $Cx_L(t)$ as in (1), which is also unrealistic since the bandwidth cannot be instantaneously adapted. For the practical implementation of synchronous dynamic allocation, the bandwidth is periodically adapted by $C\hat{x}_{max}(t)$, where $\hat{x}_{max}(t)$ represents the maximum prediction of $x_L(t)$ in the next adaptation interval defined in (16). For the asynchronous dynamic allocation, the bandwidth is determined by $C\,max\{\phi, \hat{x}_{max}(t)\}$ where $\phi$ stands for a preassigned nominal video bandwidth. We assume $\phi = E[x(t)] + (Var[x(t)])^{1/2}$. As mentioned before, if such statistics are not available *a priori*, one can choose a reasonable value for $\phi$. Depending on $\phi$, there is a tradeoff between transmission efficiency and protocol-processing efficiency. Both RLS and PSN-TDNN schemes, designed in Section III at the adaptation interval $M\Delta = 0.56$ s, are used to evaluate $\hat{x}_{max}(t)$ in the synchronous/asynchronous dynamic allocation. We choose $C = 1.25$ for the dynamic allocation schemes.

Listed in Table VII are the transmission efficiency and queueing solutions with respect to each allocation scheme. Obviously, the transmission efficiency $\rho$ reaches its highest value at 0.80 by the ideal dynamic allocation, while its lowest value occurs at 0.54 by the static allocation. We also get $\rho = 0.74$ for the synchronous dynamic allocation and $\rho = 0.62$ for the asynchronous dynamic allocation. Note that one can get $\rho = C^{-1}$ for the ideal dynamic allocation. The rest of the values of the transmission efficiency are measured by simulation. Although both RLS and PSN-TDNN prediction schemes have achieved the same transmission efficiency ($\rho =$

TABLE VII
PERFORMANCE COMPARISON OF DIFFERENT BANDWIDTH-ALLOCATION SCHEMES FOR A FINITE-BUFFER ZERO-LOSS SYSTEM TO TRANSMIT 12-MIN VIDEO IN THE WORST SCENARIO

| | Static Allocation | Dynamic Allocation | | | | |
|---|---|---|---|---|---|---|
| | (filtered peak) | ideal | RLS (syn.) | PSN (syn.) | RLS (asyn.) | PSN (asyn.) |
| $\rho$ | 0.54 | 0.80 | 0.74 | 0.74 | 0.62 | 0.62 |
| $\bar{q}$ | 6.4 | 10.7 | 9.1 | 8.7 | 7.0 | 7.0 |
| $\sigma_q$ | 8.7 | 13.3 | 11.7 | 10.2 | 8.9 | 8.8 |
| $q_{max}$ | 138 | 283 | 269 | 231 | 202 | 186 |

TABLE VIII
PERFORMANCE COMPARISON OF SYNCHRONOUS DYNAMIC BANDWIDTH-ALLOCATION SCHEMES USING NONPREDICTIVE BANDWIDTH ESTIMATION, RLS BANDWIDTH PREDICTION, AND PSN-TDNN BANDWIDTH PREDICTION IN THE WORST SCENARIO (PAGE 39)

| | ideal | RLS | PSN | non-predictive estimation | | | |
|---|---|---|---|---|---|---|---|
| | ($C = 1.25$) | | | ($C = 1.25$) | | ($C = 1.30$) | |
| | | | | $(D+M)\Delta = 0.7$ | $(D+M)\Delta = 0.84$ | $(D+M)\Delta = 0.7$ | $(D+M)\Delta = 0.84$ |
| $\rho$ | 0.80 | 0.73 | 0.73 | 0.76 | 0.76 | 0.73 | 0.73 |
| $\bar{q}$ | 10.7 | 9.0 | 8.3 | 23.8 | 37.6 | 15.2 | 23.4 |
| $\sigma_q$ | 15.2 | 14.1 | 11.2 | 78.2 | 139.2 | 47.3 | 92.1 |
| $q_{max}$ | 218 | 269 | 231 | 877 | 1351 | 719 | 1041 |

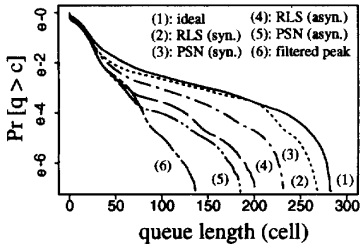



Fig. 11. Queue distributions in 12-min video transmission by different allocation schemes in the worst scenario.

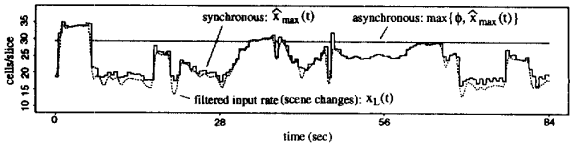



Fig. 12. A sample path of synchronous/asynchronous dynamic allocation (on page 41) using the PSN-TDNN scheme.

0.74 or 0.62), the queueng performance of the PSN-TDNN scheme is better than that of the RLS scheme as shown in Table VII. This is consistent to the prediction-performance comparison in Section III. DIsplayed in Fig. 11 is the queue distribution with respect to each allocation scheme. In Fig. 12, we show a sample path of the synchronous/asynchronous dynamic allocation using the PSN-TDNN scheme. As one can see, the asynchronous operation significantly reduces the frequency of bandwidth adaptation, which is desirable for low-complexity network management, but at the expense of increased transmission bandwidth. This study indicates the significant performance improvement due to dynamic allocation and the feasiblity of its implementation at a reasonably long adaptation interval such as 0.56 s for video transmission.

To further verify the significance of bandwidth prediction, we also study a nonpredictive bandwidth-estimation scheme where the bandwidth for the interval $[(n+D)\Delta, (n+D+M)\Delta)$ is approximated at time $n$ by the maximum of $(M+1)$ current observations, i.e., $C \max\{x_L(n-M), x_L(n-M+1), \cdots, x_L(n)\}$. The queueing performance by such a nonpredictive scheme is compared with those by the RLS and the PSN-TDNN schemes in the case of synchronous dynamic bandwidth allocation with an adaptation interval $M\Delta = 0.56$ s. We use the same RLS and PSN-TDNN schemes as designed in Section III. Specifically, the prediction lead time $(D+M)\Delta$ is set at 0.7 seconds for the RLS scheme and 0.84 seconds for the PSN-TDNN scheme. For comparison, we use the same lead time for the nonpredictive estimator. Listed in Table VIII are the results when $C = 1.25$. The RLS and PSN-TDNN schemes almost achieve the ideal queueing performance, whereas the nonpredictive scheme greatly degrades the queueing performance.

On the other hand, compared to $\rho = 0.76$ with the nonpredictive scheme, both RLS and PSN-TDNN schemes result in relatively lower utilization ($\rho = 0.73$), since they tend to overestimate the bandwidth. Hence, for fair comparison, we study another nonpredictive estimation case where the control parameter $C$ is set at 1.30 to achieve the same utilization as in the RLS and the PSN-TDNN schemes. By allocating more bandwidth in such a manner, the queueing performance by the nonpredictive scheme is improved to some extent. However, the resulting performance is still much worse than those in the RLS and the PSN-TDNN schemes.

Next, we consider a single transmisson trunk of fixed bandwidth $\mu$ to support five VC connections as shown in

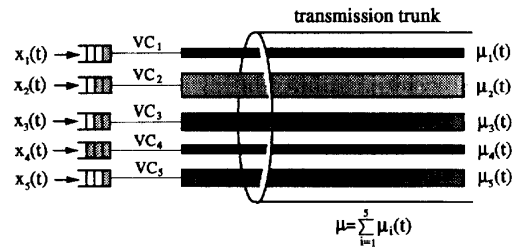



Fig. 13. Transmission of five video sources by dynamic sharing on a single ATM trunk.

Fig. 13. Each connection is associated with a separate buffer. The ATM traffic on each connection, denoted by $x_i(t)$ at $i = 1, 2, \cdots, 5$, is represented by a 2-min video source. We use pages 39–41, 55, and 56 of the JPEG "Star Wars" to individually represent each source. Let the transmission bandwidth of each VC be dynamically adapted based on the filtered $x_i(t)$'s. As in the previous example, we choose $\omega_c = 2\pi$ for the input filter and denote the filtered $x_i(t)$ by $x_{iL}(t)$. For the ideal dynamic sharing, the bandwidth of each VC is instantaneously changed by

$$\mu_i(t) = \frac{x_{iL}(t)\mu}{\sum_{j=1}^{5} x_{jL}(t)}, \qquad i = 1, 2, \cdots, 5.$$

For the synchronous dynamic sharing, similar to the definition of $\hat{x}_{\max}(t)$ in (16), we use $\hat{x}_{\max i}(t)$ to represent the maximum prediction of $x_{iL}(t)$ in each adaptation interval. The total bandwidth is then adaptively partitioned among the five VC's in every $M\Delta$ interval, according to

$$\mu_i(t) = \frac{\hat{x}_{\max i}(t)\mu}{\sum_{j=1}^{5} \hat{x}_{\max j}(t)}, \quad t \in [(n+D)\Delta, (n+D+M)\Delta).$$

Here we use the same RLS and PSN-TDNN schemes as designed in Section III, where the adaptation interval is fixed at 0.56 s. Assume that the overall trunk utilization is $\rho = 0.7$. Since the mean of the aggregate video traffic is 114.1 cells/slice, we have the total trunk bandwidth equal to $\mu = 163.0$ cells/slice. One slice corresponds to 1.4 ms. Listed in Table IX are the queueing solutions of each VC connection using different (synchronous) dynamic sharing policies in the worst scenario. As one can see, every dynamic sharing policy provides a fair service performance among the individual connections. It is also interesting to observe that the performance of the synchronous dynamic sharing is almost identical to that of the ideal one.

For comparison, we also consider some static sharing policies where the total bandwidth is statically divided by

$$\mu_i(t) = \frac{e_i\mu}{\sum_{j=1}^{5} e_j}, \qquad \forall t$$

where $e_i$ denotes a *static bandwidth measure* of the $i$th connection. For instance, $e_i$ can be the peak of the input traffic ($\max_t x_i(t)$), the peak of the filtered input ($\max_t x_{iL}(t)$), or the average input ($E[x_i(t)]$). The assumption of infinite buffer size is made for each connection since we are more interested in the worst-case queue (or delay) than cell loss. As shown in Table IX, the queueing solutions of the static sharing are highly unbalanced among the individual connections. In contrast, the queueing performance of the static sharing is much worse than that of the dynamic one. In summary, the temporal bandwidth demand of each video connection is essentially characterized by the scene changes, which are highly predictable through the on-line traffic measurement. One can therefore implement the dynamic sharing to significantly reduce the transmission bandwidth and buffer-capacity requirement.

TABLE IX
PERFORMANCE COMPARISON OF DIFFERENT SHARING POLICIES OF TRANSMISSION BANDWIDTH

| | Static Allocation | | | Dynamic Allocation | | |
|---|---|---|---|---|---|---|
| | peak | filtered peak | mean | ideal | RLS | PSN |
| $\bar{q}_1$ | 11.5 | 137.0 | 9.8 | 8.3 | 8.5 | 8.2 |
| $\sigma_{q_1}$ | 17.7 | 371.8 | 13.4 | 9.7 | 14.0 | 9.6 |
| $q_{max1}$ | 292 | 2179 | 214 | 171 | 445 | 177 |
| $\bar{q}_2$ | 30.5 | 203.9 | 604.1 | 10.5 | 11.1 | 11.1 |
| $\sigma_{q_2}$ | 122.0 | 672.7 | 1998.3 | 13.8 | 15.3 | 15.4 |
| $q_{max2}$ | 1753 | 4129 | 11737 | 183 | 178 | 159 |
| $\bar{q}_3$ | 247.0 | 138.3 | 247.0 | 8.5 | 8.1 | 8.5 |
| $\sigma_{q_3}$ | 1206.4 | 682.6 | 1206.4 | 9.3 | 8.7 | 9.6 |
| $q_{max3}$ | 9427 | 5656 | 9427 | 123 | 76 | 140 |
| $\bar{q}_4$ | 10.2 | 19.9 | 10.2 | 9.0 | 9.3 | 9.2 |
| $\sigma_{q_4}$ | 21.8 | 66.0 | 21.8 | 9.5 | 9.8 | 9.7 |
| $q_{max4}$ | 672 | 1490 | 672 | 131 | 113 | 135 |
| $\bar{q}_5$ | 106.9 | 16.3 | 78.6 | 9.2 | 9.0 | 10.0 |
| $\sigma_{q_5}$ | 670.6 | 89.7 | 507.1 | 10.2 | 9.4 | 15.5 |
| $q_{max5}$ | 6170 | 1282 | 5000 | 177 | 138 | 269 |

For the reasonably long time-varying scale of scene changes, we also expect that the same dynamic bandwidth-allocation scheme can be incorporated with network-wide control of video traffic flow. As mentioned, one feasible approach is a dynamic rerouting scheme [4], [5] based on global traffic measurements. When enough bandwidth is not available at a certain link, some of active VC's are selectively rerouted to the other paths which are being underutilized. By doing so, not only the per-connection bandwidth demand can be guaranteed but also the network-wide load balance can be achieved.

## V. CONCLUSION

This paper has presented a novel approach to dynamic bandwidth allocation for transport of real-time VBR video over ATM networks. The study indicates that the video-transmission bandwidth in a finite-buffer system is essentially characterized by the low-frequency signal that captures the time-variation of video scene changes. Furthermore, it is observed that the low-frequency video signal stays in a well-defined low-frequency band, typically $\omega < 2\pi$ radians. The dynamic bandwidth-allocation scheme proposed in this paper is based on on-line observation and prediction of the slowly time-varying

video scene changes. In the design of prediction scheme, both recursive least square method and time-delay neural network method are examined and compared. In particular, a time-delay neural network with low-complexity high-order architecture, called pi-sigma network, is found to be effective in predicting video scene changes. The proposed dynamic bandwidth-allocation scheme provides a new solution for efficient transmission of real-time video traffic with guaranteed quality-of-services.

## ACKNOWLEDGMENT

The authors thank M. Garrett at Bellcore and P. Pancha at University of Pennsylvania for providing them with the video clips used in this paper.

## REFERENCES

[1] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, pp. 968–981, Sept. 1991.

[2] A. Elwalid and D. Mitra, "Effective bandwidth of general Markovian traffic sources and admission control of high speed networks," *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 329–343, June 1993.

[3] S. Q. Li and C. Hwang, "Queue response to input correlation functions: Continuous spectral analysis," *IEEE/ACM Trans. Networking*, vol. 2, no. 6, pp. 678–692, Dec. 1993.

[4] S. Q. Li, S. Chong, C. Hwang, and X. Zhao, "Link capacity allocation and network control by filtered input rate in high speed networks," in *Proc. IEEE GLOBECOM'93*, Dec. 1993, pp. 744–750.

[5] B. P. Mohanty, C. G. Cassandras, and D. Towsley, "Performance comparison of routing algorithms in packet switched networks," in *Proc. IEEE GLOBECOM'90*, Dec. 1990, pp. 327–331.

[6] M. Garrett and M. Vetterli, "Congestion control strategies for packet video," *Fourth Int. Workshop Packet Video* (Kyoto, Japan), Aug. 1991.

[7] P. Pancha and M. El Zarki, "Bandwidth requirements of variable bit rate MPEG sources in ATM networks," in *Proc. IEEE INFOCOM'93*, Mar. 1993, pp. 902–909.

[8] D. Le Gall, "MPEG: A video compression standard for multimedia applications," *Commun. Assoc. Comput. Mach.*, vol. 34, no. 4, pp. 46–58, Apr. 1991.

[9] G. C. Goodwin and K. S. Sin, *Adaptive Filtering, Prediction and Control.* Englewood Cliffs, NJ: Prentice Hall, 1984.

[10] S. Haykin, *Adaptive Filter Theory.* Englewood Cliffs, NJ: Prentice-Hall, 1991.

[11] J. M. Cioffi and T. Kailath, "Fast, recursive-least-squares transversal filters for adaptive filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 304–337, 1984.

[12] J. Ghosh and Y. Shin, "Efficient high-order neural networks for classification and function approximation," *Int. J. Neural Syst.*, vol. 3, no. 4, pp. 323–350, 1992.

[13] B. Hammerstrom, "Neural networks at work," *IEEE Spectrum*, pp. 26–32, June 1993.

[14] D. Park *et al.*, "Electric load forecasting using an aritificial neural network," *IEEE Trans. Power Syst.*, vol. 6, no. 2, pp. 442–449, May 1991.

[15] H. Yang, T. Akiyama, and T. Sasaki, "A neural network approach to the identification of real-time origin-destination flows from traffic counts," in *Proc. Int. Conf. Artific. Intell. Applicat. Transportation Eng.*, June 1992, pp. 253–269.

[16] A. Hiramatsu, "ATM communications network control by neural networks," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 122–130, Mar. 1990.

[17] A. Hiramatsu, "Integration of ATM call admission control and link capacity control by distributed neural networks," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, pp. 1131–1138, Sept. 1991.

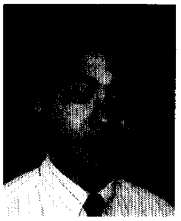
[18] J. Hertz, A. Krogh, and R. Palmer, *Introduction to the Theory of Neural Computation.* Reading, MA: Addison-Wesley, 1991.

[19] D. Hush and B. Horone, "Progress in supervised neural networks: What's new since Lippmann?" *IEEE Signal Processing Mag.*, pp. 8–38, Jan. 1993.

[20] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 23, pp. 359–366, 1989.

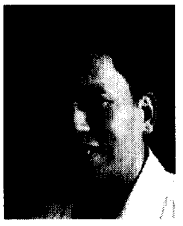
[21] C. Giles and T. Maxwell, "Learning, invariance, and generalization in a high-order neural network," *Appl. Optics*, vol. 26, no. 23, pp. 4972–4978, 1987.

[22] S. German, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, no. 1, pp. 1–58, 1992.



**Song Chong** received the B.S. and M.S. degrees from Seoul National University, Seoul, Korea, in 1988 and 1990, respectively, both in control engineering. He is working toward the Ph.D. degree in computer engineering at the University of Texas, Austin.

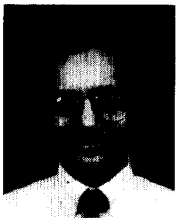
Since December 1994, he has been with AT&T Bell Laboratories, Holmdel, NJ, as a Member of the Technical Staff. His current research focuses on high-speed networking (ATM, B-ISDN) for multimedia communications with emphasis on traffic characterization, resource management, congestion control, QOS control, and routing. His research interests also include queueing systems, stochastic models, simulation, discrete event systems, artificial neural systems and their applications to design, modeling and performance analysis of computers and computer communication systems.



**San-qi Li** (M'86) received the B.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 1975, and the M.A.Sc. and Ph.D. degrees from the University of Waterloo, Waterloo, Ontario, Canada, in 1982 and 1985, respectively, all in electrical engineering.

From 1976 to 1980, he was with Posts and Telecommunications Academy of China as a Research Supervisor. From 1985 to 1989, he was an Associate Research Scientist and Principal Investigator for the Center for Telecommunications Research, Columbia University, New York, NY, where he was actively involved in conducting research, supervising doctoral students, and collaborating with faculty members. In September 1989, he joined the Department of Electrical and Computer Engineering at the University of Texas, Austin, where he is presently an Associate Professor. He has published over 80 refereed technical papers in the area of computer communication networks research. His current research interests include multimedia traffic analysis and control of high-speed networks.

Dr. Li received the Best Paper Award of the *1992 IEEE INFOCOM* Conference and has served as a member of the Technical Program Committee for *IEEE INFOCOM* Conference since 1988.



**Joydeep Ghosh** received the B.Tech degree in electrical engineering from IIT, Kanpur, India, in 1983. In 1988, he received the Ph.D. degree in computer engineering from the University of Southern California, Los Angeles (USC).

He is currently an Associate Professor in the Department of Electrical and Computer Engineering, University of Texas, Austin. His research interests are in the areas of artificial neural systems and highly parallel processing, with applications to image and signal processing, and has over 50 refereed publications in these areas. He has also coauthored six book chapters and edited two books: *Intelligent Engineering Systems Through Artificial Neural Networks* (ASME Press, 1993) and *Parallel Architectures for Image Processing* (1990).

Dr. Ghosh was the first student from the School of Engineering at USC to be awarded an "All-University Predoctoral Merit Fellowship" for four years. He has received several "best paper" awards, including the 1992 Darlington Award (for best journal paper) from the IEEE Circuits and Systems Society. He served as conference cochair for ANNIE'93 and ANNIE'94, as a member of the editorial board of IEEE Computer Society Press, and is currently an Associate Editor of *Pattern Recognition.*