On End-to-End Architecture for Transporting MPEG-4 Video Over the Internet

Dapeng Wu, *Student Member, IEEE*, Yiwei Thomas Hou, *Member, IEEE*, Wenwu Zhu, *Member, IEEE*, Hung-Ju Lee, *Member, IEEE*, Tihao Chiang, *Senior Member, IEEE*, Ya-Qin Zhang, *Fellow, IEEE*, and H. Jonathan Chao, *Senior Member, IEEE*

***Abstract*—With the success of the Internet and flexibility of MPEG-4, transporting MPEG-4 video over the Internet is expected to be an important component of many multimedia applications in the near future. Video applications typically have delay and loss requirements, which cannot be adequately supported by the current Internet. Thus, it is a challenging problem to design an efficient MPEG-4 video delivery system that can maximize the perceptual quality while achieving high resource utilization. This paper addresses this problem by presenting an end-to-end architecture for transporting MPEG-4 video over the Internet. We present a framework for transporting MPEG-4 video, which includes source rate adaptation, packetization, feedback control, and error control. The main contributions of this paper are: 1) a feedback control algorithm based on Real Time Protocol (RTP) and Real Time Control Protocol (RTCP); 2) an adaptive source-encoding algorithm for MPEG-4 video which is able to adjust the output rate of MPEG-4 video to the desired rate; and 3) an efficient and robust packetization algorithm for MPEG video bit-streams at the sync layer for Internet transport. Simulation results show that our end-to-end transport architecture achieves good perceptual picture quality for MPEG-4 video under low bit-rate and varying network conditions and efficiently utilizes network resources.**

***Index Terms*—Adaptive encoding, feedback control, Internet, MPEG-4, packetization, RTP/RTCP, video object.**

I. Introduction

With the success of the Internet and the emergence of a multimedia communication era, the new international standard MPEG-4 [13] is poised to become the enabling technology for multimedia communications in the near future. MPEG-4 builds on elements from several successful technologies, such as digital video, computer graphics, and the World Wide Web, with the aim of providing powerful tools in the production, distribution, and display of multimedia contents. With the flexibility and efficiency provided by coding a new form of visual data called visual object (VO), it is foreseen that MPEG-4 will be capable of addressing interactive content-based video services, as well as conventional storage and transmission of video.

Internet video applications typically have unique delay and loss requirements which differ from other data types. Furthermore, the traffic load condition over the Internet varies drastically over time, which is detrimental to video transmission [1], [2], [25]. Thus, it is a major challenge to design an efficient video delivery system that can both maximize users' perceived quality of service (QoS) while achieving high resource utilization in the Internet.

Since the standardization of MPEG-4, there has been little study on how to transport MPEG-4 video over Internet protocol (IP) networks. This paper presents an end-to-end architecture for transporting MPEG-4 video over IP networks. The objective of our architecture is to achieve good perceptual quality at the application level while being able to adapt to varying network conditions and to utilize network resources efficiently.

We first outline the key components in a video transport architecture, which include source-rate adaptation, packetization, end-to-end feedback control, and error control. Since the current Internet only offers best-effort service, it is essential for the end systems (sender and receiver) to actively perform feedback control so that the sender can adjust its transmission rate. Therefore, appropriate feedback control and source rate adaptation must be in place. Since bit-oriented syntax of MPEG-4 has to be converted into packets for transport over the network, an appropriate packetization algorithm is essential to achieve good performance in terms of efficiency and picture quality. Finally, it is necessary to have some error control scheme in place to minimize the degradation of perceptual video quality should packet loss occur during transport.

The main contributions of this paper are: 1) an MPEG-4 video encoding rate control algorithm, which is capable of adapting the overall output rate to the available bandwidth; 2) an efficient and robust packetization algorithm by exploiting the unique video object plane (VOP) feature in MPEG-4; and 3) an end-to-end feedback control algorithm which is capable of estimating available network bandwidth based on the packet loss information at the receiver.

Rate-adaptive video encoding has been studied extensively in recent years. The rate-distortion (R–D) theory is a powerful tool for encoding rate control. Under the R–D framework, there are two approaches to encoding rate control in the literature: the model-based approach and the operational R–D based approach. The model-based approach assumes various input

Manuscript received March 4, 1999; revised November 23, 1999. This paper was recommended by Associate Editor H. Gharavi.

D. Wu and H. J. Chao are with Polytechnic University, Department of Electrical Engineering, Brooklyn, NY 11201 USA.

Y. T. Hou is with Fujitsu Laboratories of America, Sunnyvale, CA 94086 USA.

W. Zhu and Y.-Q. Zhang are with Microsoft Research, China, 5F, Beijing Sigma Center, Beijing 100080, China.

H.-J. Lee is with Sarnoff Corporation, Multimedia Technology Laboratory, Princeton, NJ 08543-5300 USA.

T. Chiang is with the National Chiao Tung University, Department of Electronics Engineering, Hsinchu 30010, Taiwan.

Publisher Item Identifier S 1051-8215(00)07559-5.

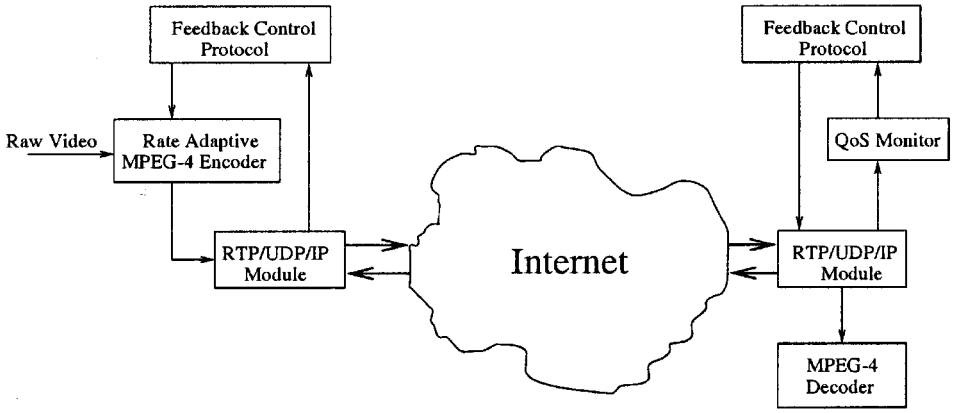


Fig. 1. An end-to-end architecture for transporting MPEG-4 video.

distribution and quantizer characteristics [4], [7], [8], [16], [27]. Under this approach, closed-form solutions can be obtained through using continuous optimization theory. On the other hand, the operational R–D based approach considers practical coding environments where only a finite set of quantizers is admissible [12], [14], [22], [30]. Under the operational R–D based approach, the admissible quantizers are used by the rate-control algorithm to determine the optimal strategy to minimize the distortion under the constraint of a given bit budget. To be specific, the optimal discrete solutions can be found through using integer programming theory. In this paper, we resort to the model-based approach and extend our previous work [4]. Different from the previous work [4], our source encoding rate control presented in this paper is based on the following new concepts and techniques: 1) a more accurate second-order R-D model for the target bit-rate estimation; 2) a dynamic bit-rate allocation among video objects with different coding complexities; 3) an adaptive data-point selection criterion for better modeling the updating process; 4) a sliding-window method for smoothing the impact of scene change; and 5) an adaptive threshold shape-control for better use of bit budget. This algorithm has been adopted in the international standard for MPEG-4 [13].

Prior efforts on packetization for video applications over the Internet include schemes for H.261 [26], H.263 [35], H.263+ [3], and a scheme for MPEG-1/2 [10]. These schemes are targeted at block-based video coding, but may not be applicable to or optimal for MPEG-4. Recent effort on RTP payload format for MPEG-4 elementary streams was discussed by Schulzrinne *et al.* [20]. However, it is not clear how to perform packetization for the MPEG-4 VOPs at the sync layer (SL) before passing onto the RTP layer. In this paper, we propose a packetization algorithm for MPEG-4 bit-streams at the SL, which achieves both efficiency and robustness by exploiting the VOP concept in MPEG-4 video.

Previous work on feedback control for Internet video includes that of Turletti and Huitema [25]. Since this algorithm employs a multiplicative rate increase, there is frequent and large rate fluctuation, which leads to large packet loss ratio. This paper extends the feedback control technique used in [25] for MPEG-4 video. In particular, we design an end-to-end feedback control algorithm for MPEG-4 video by employing the RTCP feedback mechanism and using packet loss as congestion indication.

To demonstrate the performance of our end-to-end transport architecture for MPEG-4 video, we perform simulations with raw video sequences. Simulation results show that our architecture is capable of transporting MPEG-4 video over the network with good perceptual quality under low bit-rate and varying network conditions and utilizes the network resources efficiently.

The remainder of this paper is organized as follows. In Section II, we outline key components for transporting video over the Internet. Section III presents our end-to-end feedback control algorithm based on RTCP. In Section IV, we discuss our design of an adaptive video encoding algorithm for MPEG-4. Section V presents our packetization algorithm for MPEG-4 VOP bit-streams at the SL. In Section VI, we use simulation results to demonstrate the performance behavior of MPEG-4 video under our transport architecture under varying network conditions. Section VII concludes this paper and points out future research directions.

II. An Architecture for Transporting MPEG-4 Video

In this section, we outline key components of transporting video over the Internet. We organize this section as follows. In Section II-A, we overview our end-to-end architecture. From Sections II-B–II-F, we briefly describe each component in our end-to-end architecture.

A. Overview

Fig. 1 shows our end-to-end architecture for transporting MPEG-4 video over the Internet. The proposed architecture is applicable to both pre-compressed video and live video. If the source is a pre-compressed video, the bit-rate of the stream can be matched to the rate enforced by our feedback control protocol through dynamic rate shaping [9] or selective frame discarding [34]. If the source is a live video, we use the MPEG-4 rate adaptation algorithm described in Section IV to control the output rate r of the encoder.

On the sender side, raw bit-stream of live video is encoded by an adaptive MPEG-4 encoder. After this stage, the compressed video bit-stream is first packetized at the SL and then passed through the RTP/UDP/IP layers before entering the Internet.

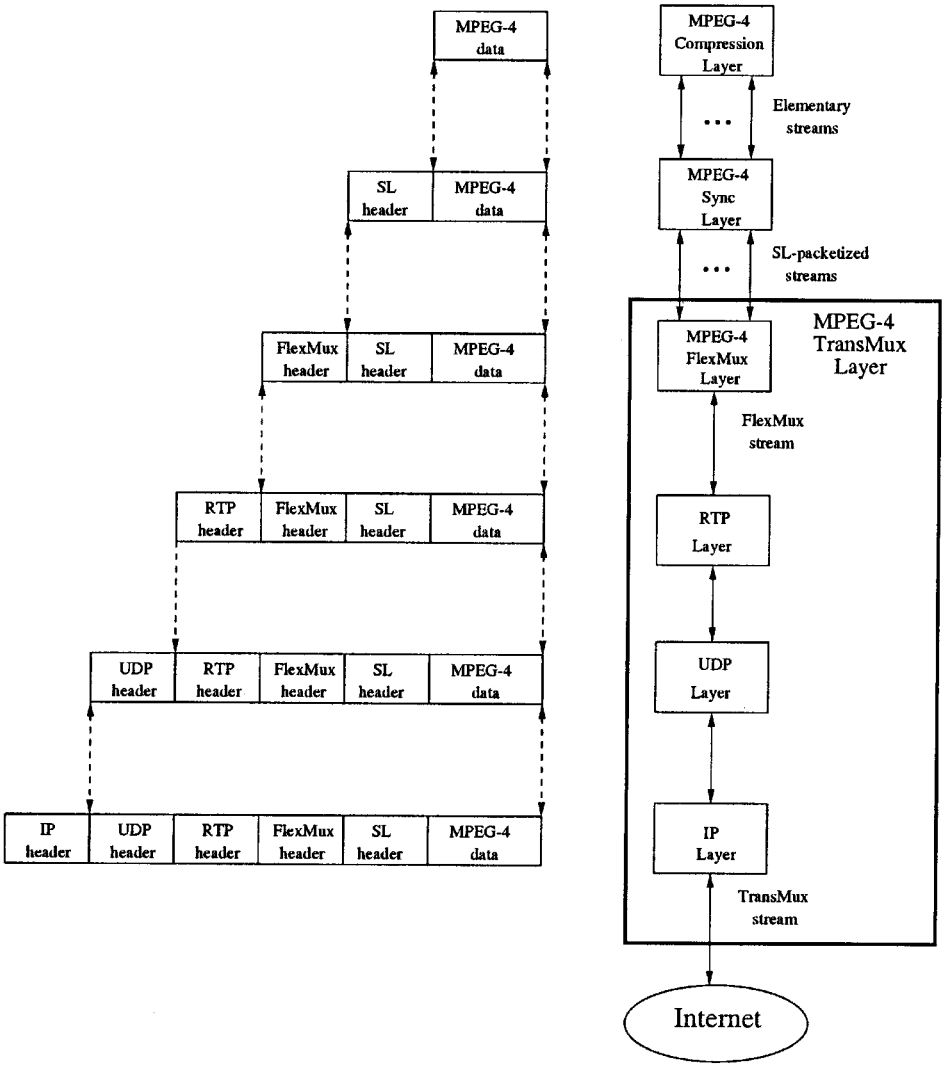


Fig. 2. Data format at each processing layer at an end system.

Packets may be dropped at a router/switch (due to congestion) or at the destination (due to excess delay). For packets that are successfully delivered to the destination, they first pass through the RTP/UDP/IP layers in reverse order before being decoded at the MPEG-4 decoder.

Under our architecture, a QoS monitor is kept at the receiver side to infer network congestion status based on the behavior of the arriving packets, e.g., packet loss and delay. Such information is used in the feedback-control protocol, which is sent back to the source. Based on such feedback information, the sender estimates the available network bandwidth and controls the output rate of the MPEG-4 encoder.

Fig. 2 shows the protocol stack for transporting MPEG-4 video. The right half of Fig. 2 shows the processing stages at an end system. At the sending side, the compression layer compresses the visual information and generates elementary streams (ESs), which contain the coded representation of the VOs. The ESs are packetized as SyncLayer (SL)-packetized streams at the SL. The SL-packetized streams provide timing and synchronization information, as well as fragmentation and random access information. The SL-packetized streams are multiplexed into a FlexMux stream at the TransMux Layer, which is then passed to the transport protocol stacks composed of RTP, UDP and IP. The resulting IP packets are transported over the Internet. At the receiver side, the video stream is processed in the reversed manner before its presentation. The left half of Fig. 2 shows the data format of each layer.

Fig. 3 shows the structure of MPEG-4 video encoder. Raw video stream is first segmented into video objects, then encoded by individual VO Encoder. The encoded VO bit-streams are packetized before multiplexed by stream multiplex interface. The resulting FlexMux stream is passed to the RTP/UDP/IP module.

The structure of MPEG-4 video decoder is shown in Fig. 4. Packets from RTP/UDP/IP are transferred to stream demultiplex interface and FlexMux buffer. The packets are demultiplexed and put into correspondent decoding buffers. The error concealment component will duplicate the previous VOP when packet loss is detected. The VO decoders decode the data in the decoding buffer and produce composition units (CUs), which are then put into composition memories to be consumed by the compositor.

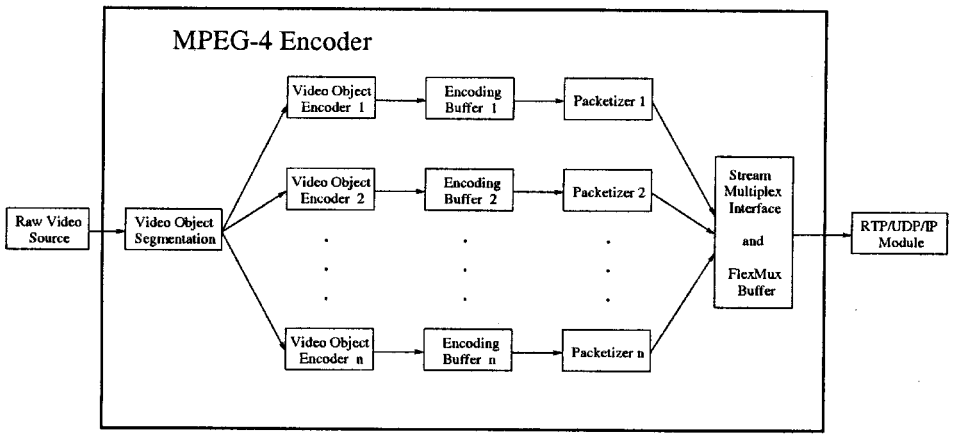


Fig. 3. Structure of MPEG-4 video encoder.

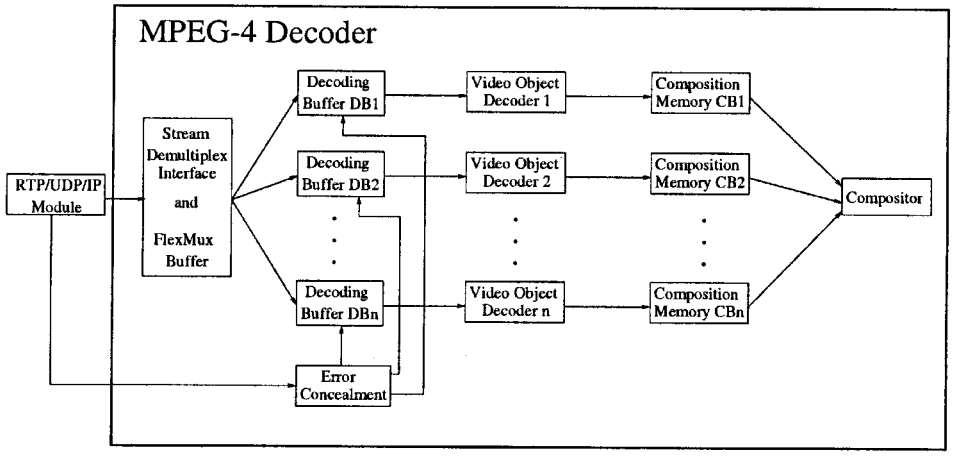


Fig. 4. Structure of MPEG-4 video decoder.

B. RTP/RTCP Protocol

Since TCP retransmission introduces delays that are not acceptable for MPEG-4 video applications with stringent delay requirement, we employ UDP as the transport protocol for MPEG-4 video streams. In addition, since UDP does not guarantee packet delivery, the receiver needs to rely on upper layer (i.e., RTP/RTCP) to detect packet loss.

RTP is an Internet standard protocol designed to provide end-to-end transport functions for supporting real-time applications [19]. RTCP is a companion protocol with RTP. RTCP designed to provide QoS feedback to the participants of an RTP session. In order words, RTP is a data transfer protocol, while RTCP is a control protocol.

RTP does not guarantee QoS or reliable delivery, but rather, provides some basic functionalities which are common to almost all real-time applications. Additional application-specific requirements are usually added on top of RTP in the form of application-specific profiles. A key feature supported by RTP is the packet sequence number, which can be used to detect packet loss at the receiver.

RTCP provides QoS feedback through the use of *sender reports* (SR) and *receiver reports* (RR) at the source and destination, respectively. In particular, RTCP keeps the total control packets to 5% of the total session bandwidth. Among the control packets, 25% are allocated to the sender reports and 75% to the receiver reports. To prevent control packet starvation, at least 1 control packet is sent within 5 s at the sender or receiver.

Fig. 5 shows our implementation architecture for RTP/UDP/IP layers. This module is a key component to realize our rate-based feedback control protocol and error control mechanism. From the sending part, the MPEG-4 encoder generates a packetized stream (FlexMux stream), which is turned into RTP packets. On the other hand, the information from feedback control protocol (sender side) is transferred to the RTCP generator. The resulting RTCP and RTP packets go down to the UDP/IP layer for transport over the Internet. On the receiving part, received IP packets are first un-packed by UDP/IP layer, then dispatched by filter and dispatcher to RTP and RTCP analyzers. RTP packets are unpacked by RTP analyzer and put into a buffer before being decoded for the purpose of loss detection. When packet loss is detected, the message will be sent to the error-concealment component. On the other hand, the RTCP analyzer unpacks RTCP packets

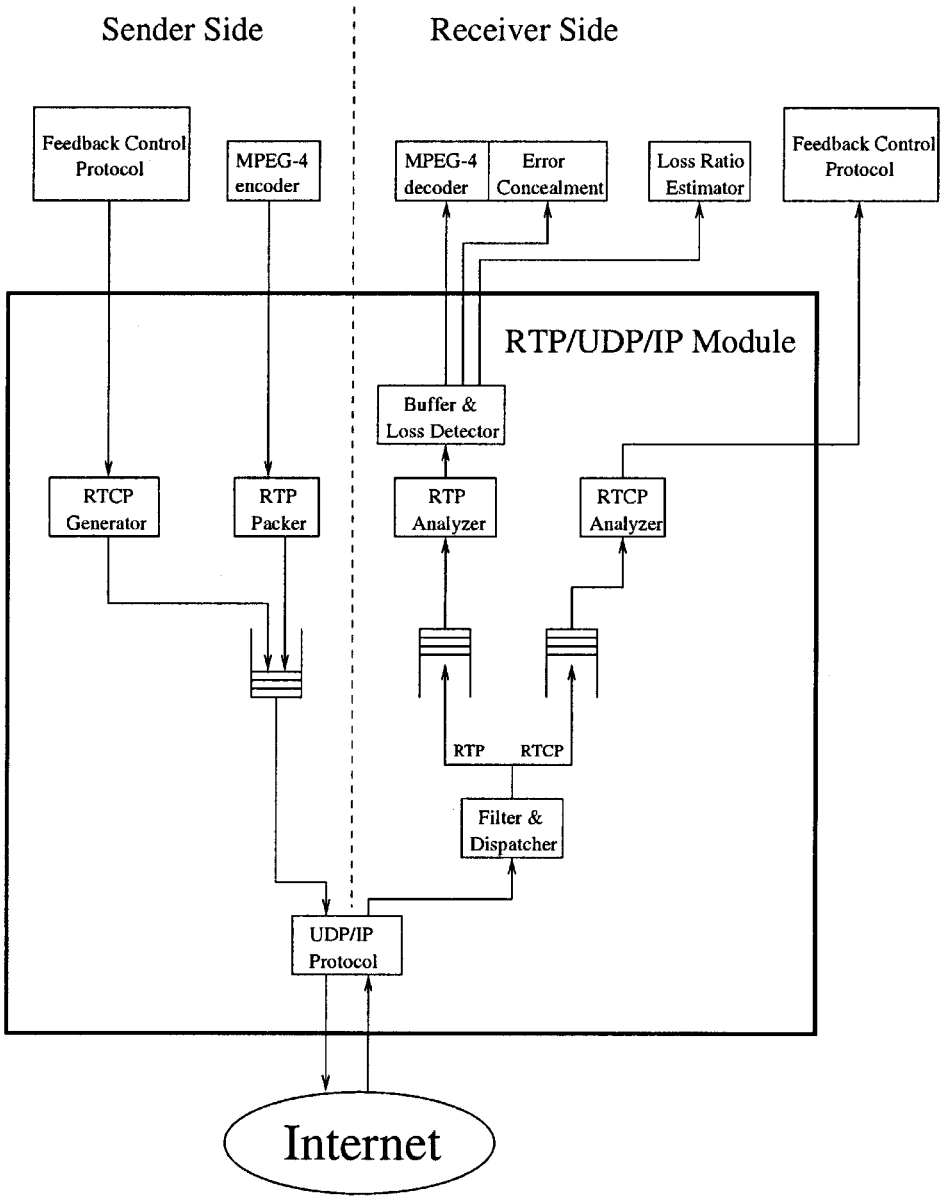


Fig. 5. Architecture of RTP/UDP/IP module.

and sends the feedback information to the Feedback Control Protocol component.

C. Feedback Control

Internet video applications typically have unique delay and loss requirements which differ from other data types. On the other hand, the current Internet does not widely support any bandwidth-reservation mechanism (e.g., RSVP) or other QoS guarantees. Furthermore, the available bandwidth not only is not known *a priori* but also varies with time. Therefore, a mechanism must be in place for MPEG-4 video source to sense network conditions so that it can encode the video with appropriate output rate.

Ideally, we would like to perform congestion indication and feedback at the point of network congestion, i.e., a bottleneck link at a switch/router.[1] Under such an environment, it is possible to design powerful rate-calculation algorithms at the switch and convey accurate available bandwidth information to the source [11]. Unfortunately, in the current Internet environment, IP switches/routers do not actively participate in feedback control.[2] All flow-control and error-recovery functions are left to the end systems and upper-layer applications. Under such an environment, we can only treat the Internet as a black box where packet loss and delay are beyond our control. Our design of the feedback-control algorithm will solely be on

[1]This is the case for the available bit-rate (ABR) service in ATM networks, where a switch actively participates in congestion control by inserting congestion and rate information in the flow-control packet (i.e., resource management (RM) cell).

[2]This is the so-called "minimalist" approach adopted by the early Internet designers, meaning that the complexity on the network side is kept as minimal as possible.

the end systems (sender and receiver) without any additional requirements on IP switches/routers.

In our architecture, we let the MPEG-4 video source gradually increase its transmission rate to probe available network bandwidth. Such a rate increase will first have the source's rate reach the available network bandwidth. Then the source rate will overshoot the available network bandwidth and fall into the congestion region. Congestion is detected by the receiver through packet loss/delay in the received packets. The receiver sends feedback RTCP packets to the source to inform it about congestion status. Once the source receives such a feedback, it decreases its transmission rate. The challenge of the feedback control lies in the proper design of such an algorithm so that a source can keep up with network bandwidth variation and thus the network is efficiently utilized. In Section III, we present the details of our feedback control algorithm that employs packet loss as congestion indication and uses RTCP control packet to convey congestion information.

D. Source-Encoding Rate Control

Since the feedback control described above needs the encoder to enforce the target rate, an MPEG-4 encoding rate-control algorithm must be in place. The objective of a source encoding rate-control algorithm is to maximize the perceptual quality under a given encoding rate. Such adaptive encoding may be achieved by the alteration of either or both of the encoder's quantization parameters and the video frame rate.

Traditional video encoders (e.g., H.261, MPEG-1/2) typically rely on altering the quantization parameter of the encoder to achieve rate adaptation. These encoding schemes perform coding with constant frame rate and does not exploit the temporal behavior of each object within a frame. Under these coders, alteration of frame rate is usually not employed, since even a slight reduction in frame rate can substantially degrade the perceptual quality at the receiver, especially during a dynamic scene change.

On the other hand, the MPEG-4 video encoder classifies each individual video object into VOP and encodes each VOP separately. Such isolation of video objects provides us with much greater flexibility to perform adaptive encoding. In particular, we may dynamically adjust target bit-rate distribution among video objects, in addition to the alteration of quantization parameters on each VOP.[3]

In Section IV, we will present a novel source encoding rate control algorithm for MPEG-4. Our algorithm is capable of achieving the desired output rate with excellent perceptual quality.

E. Packetization of MPEG-4 Bit Stream

Since MPEG-4 video stream has to be converted into packets for transport over the network, a packetization algorithm must be in place.

[3]To obtain better coding efficiency, it is fairly straightforward to encode different video objects at different frame rate. However, our experimental results show that significant quality deterioration is experienced in the "gluing" boundary of video objects. Thus, we find that encoding all the video objects at the same frame rate is a better alternative since this is less costly and achieves better video quality.

In Fig. 2, we showed the protocol stack for transporting MPEG-4 video. The RTP payload format for MPEG-4 elementary stream was proposed by Schulzrinne *et al.* [20]. However, it is not clear what kind of packetization algorithm should be employed at the SL before entering the RTP layer.

Due to the VOP property in MPEG-4, the packetization algorithm for MPEG-4 needs to be carefully designed for Internet transport and packetization algorithms for H.261/263 and MPEG-1/2 cannot be directly applied here. In Section V, we present a packetization algorithm for MPEG-4 video at the SL that achieves both efficiency and robustness.

F. Error Control

Lack of QoS support on the current Internet poses a challenging task for Internet video applications. In contrast to a wireless environment, where transmission errors are the main cause for quality deterioration, the degradation of picture quality in the Internet is attributed primarily to packet loss and delay. Therefore, error control/resilience solutions for a wireless environment [21], [32] are not applicable to the Internet environment.

Internet packet loss is mainly caused by congestion experienced in the routers. Furthermore, due to multi-path routing in the network, packets can be delayed or received out of sequence. Due to real-time requirements at the receiver, such delayed video packets may be considered as lost packets if their delays exceed a maximum threshold. Although MPEG-4 video stream can tolerate some loss, it does degrade the perceptual quality at the receiver. Therefore, error control and resilience mechanisms must be in place to maintain an acceptable perceptual quality.

Previous work on error control took two major approaches, i.e., forward error correction (FEC) and retransmission [1], [6], [18]. Danskin *et al.* [6] introduced a fast lossy Internet image transmission scheme (FLIIT). Although FLIIT eliminates retransmission delays and is thus able to transmit the same image several times faster than TCP/IP with equivalent quality, the joint source/channel coding (FEC approach) employed in FLIIT cannot be directly used in MPEG-4 video since MPEG-4 video coding algorithms are different from those used by FLIIT. Bolot *et al.* [1] also took the FEC approach which could not be applied to MPEG-4 video due to the difference between the two coding algorithms. Rhee [18] proposed a retransmission-base scheme, called periodic temporal dependency distance (PTDD). Although experiments had shown some utility of PTDD, the experiments were limited to small number of users. Since all retransmission-based error control schemes suffer from network congestion, the effectiveness of PTDD is questionable in a case where large number of video users employ PTDD within a highly congested network. Since FEC introduces a large overhead, it may not be applicable to very low bit-rate video with MPEG-4. We do not favor a retransmission-based error control scheme either, since large-scale deployment of retransmission-based scheme for transporting video may further deteriorate network congestion and cause a network collapse.

Another method to deal with error and loss in the transmission is error resilient encoding. The error-resilience mechanisms in the literature include re-synchronization marking, data partitioning, data recovery [e.g., reversible variable-length codes

(RVLC)], and error concealment [23], [24], [28], [29], [31], [33]. However, re-synchronization marking, data partitioning, and data recovery are targeted at error-prone environments like wireless channels and may not be applicable to the Internet environment. For video transmission over the Internet, the boundary of a packet already provides a synchronization point in the variable-length coded bit-stream at the receiver side. Since a packet loss may cause the loss of all the motion data and its associated shape/texture data, mechanisms such as re-synchronization marking, data partitioning, and data recovery may not be useful for Internet video applications. On the other hand, most error-concealment techniques discussed in [29] are only applicable to either ATM or wireless environments, and require substantial additional computation complexity, which is tolerable in decoding still images but not tolerable in decoding real-time video. Therefore, we only consider a simple error-concealment scheme that is applicable to Internet video applications.

With the above considerations, we employ a simple scheme for error control as follows. Packet loss is detected by the QoS monitor by examining the RTP packet sequence number at the receiver side (Fig. 1). In our implementation, we consider a packet as lost if it is delayed beyond the fourth packet behind it (although it may arrive at a future time). Here, according to the maximum playback delay, we choose the fourth packet as the threshold. The maximum playback delay can be specified by the user according to the requirement of the application. Our algorithm for error control consists of two parts. On the decoder side, when packet loss is detected, the data from the previously reconstructed VOP is simply repeated to recover the image regions corresponding to the lost packets. On the encoder side, the encoder periodically encodes intra-VOP so as to suppress error propagation.

III. End-to-End Feedback Control Protocol

As discussed in Section II-C, the switches/routers in the current Internet do not provide the source with explicit rate feedback information about available network bandwidth. We can only estimate network available bandwidth at the end system indirectly through delay [e.g., roundtrip time (RTT)] or packet loss ratio. It has been shown by Dabbous in [5] that the throughput of connections using RTT-based feedback is lower than the throughput of connections such as TCP connections using loss-based feedback. Therefore, we choose to employ packet loss ratio measured at the receiver as feedback information in our scheme.

Consistent with the RTP/RTCP standard [19], we let the source periodically send one RTCP control packet for every N_s RTP packets. The receiver sends a feedback RTCP control packet to the source upon receiving N_r packets or at least once every 5 s. The returning RTCP packet contains the packet loss ratio P_{loss}, which the receiver observed during the N_r packet time interval since the previous feedback RTCP packet. Rate control actions are taken by the encoder upon receiving a backward RTCP packet. Therefore, the interval between successive rate control at source is approximately equal to the interval between successive backward RTCP packets.

The following is our feedback-control protocol at sender and receiver, where IR, MR, and PR are the initial rate, minimum rate, and peak rate of the sender, respectively. AIR is the additive increase rate, α is the multiplicative decrease factor, and $P_{\text{threshold}}$ is the threshold for packet loss ratio.

Algorithm 1(Feedback-Control Protocol)

Sender Behavior

- The sender starts to transmit at the output rate r: = IR, which is greater than or equal to its minimum rate MR; each RTP data packet contains a packet sequence number.
- For every N_s transmitted RTP data packets, the sender sends a forward RTCP control packet;
- Upon the receipt of a backward RTCP packet with the packet loss ratio P_{loss} from the receiver, the output rate r at the source is adjusted according to the following rule:

```
if (P_loss ≤ P_threshold)
    r: = min{(r + AIR), PR};
else
    r: = max{(α × r), MR}.
```

Receiver Behavior

- The receiver keeps track of the sequence number in the RTP header of the arriving packets;
- Upon receiving N_r packets or at most 5 s, the receiver sends a feedback RTCP packet to the source containing packet loss rate P_{loss} it observes during this time interval.

During a control action, the feedback control algorithm (Algorithm 1) adjusts the output rate r of the MPEG-4 encoder in an attempt to maintain the packet loss ratio P_{loss} below the threshold $P_{\text{threshold}}$. Unlike the scheme by Turletti and Huitema in [25], where a multiplicative increase rate adjustment is employed when a feedback RTCP packet indicates that there is no congestion, we employ additive increase in Algorithm 1, which is a conservative rate increase approach to adapt to available network bandwidth. Our experience shows that a multiplicative increase usually brings much larger source rate oscillation and more packet loss in a large network than a conservative rate adjustment such as additive increase. On the other hand, we employ multiplicative decrease in Algorithm 1 should the source find that the P_{loss} is larger than threshold in the returning RTCP packet. We find that such swift rate reduction at the source is necessary to shorten the congestion period and reduce packet loss.

IV. Adaptive Encoding Rate Control (ARC) for MPEG-4 Video

In this section, we design an adaptive encoding algorithm for MPEG-4 so that the output rate of the encoder can match the estimated rate by our feedback control protocol in Section III.

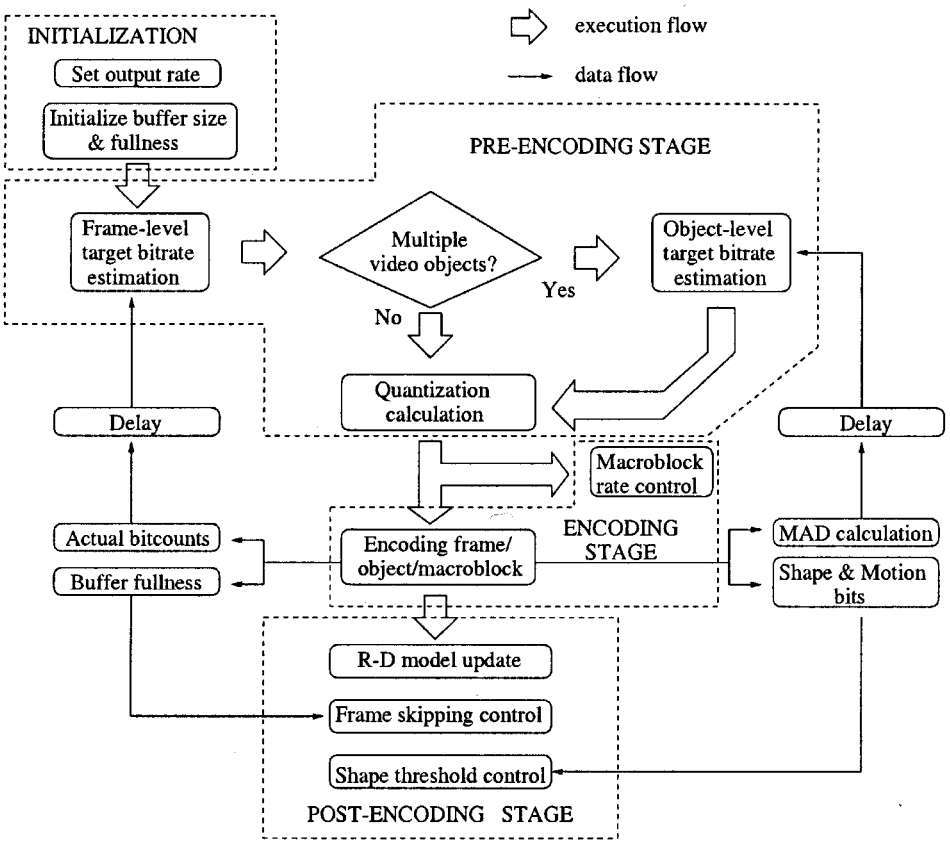


Fig. 6. Block diagram of our encoding rate-control scheme.

Our ARC scheme is based on the following new concepts and techniques:

1) a more accurate second-order R-D model for the target bit-rate estimation;
2) a dynamical bit-rate allocation among video objects with different coding complexities;
3) a sliding-window method for smoothing the impact of scene change;
4) an adaptive data-point selection criterion for better model updating process;
5) an adaptive threshold shape control for better use of bit budget;
6) a novel frame-skipping control mechanism.

The ARC scheme provides an integrated solution with three different coding granularities, including: 1) frame-level; 2) object-level; and 3) macroblock-level. ARC is able to achieve a more accurate target bit-rate allocation under the constraints of low latency and limited buffer size than the first-order R-D model. In addition to the frame-level rate control, the ARC scheme is also applicable to the macroblock-level rate control for finer bit allocation and buffer control, and multiple VO's rate control for better VO presentation when more network bandwidth is available. The block diagram of our ARC scheme is depicted in Fig. 6. Table I lists the notations which will be used in this section.

We organize this section as follows. In Section IV-A, we present the theoretical foundation behind ARC scheme. Sections IV-B–IV-E present the details of the four stages in ARC scheme: 1) initialization; 2) pre-encoding; 3) encoding; and 4) post-encoding.

A. Scalable Quadratic R-D Model

In our previous work [4], a model for evaluating the target bit-rate is formulated as follows:

$$B_i = a_1 \times Q_i^{-1} + a_2 \times Q_i^{-2} \tag{1}$$

where

B_i total number of bits used for encoding the current frame i;
Q_i quantization level used for the current frame i;
a_1, a_2 first- and second-order coefficients.

Although the above R-D model can provide the theoretical foundation of the rate-control scheme, it has the following two major drawbacks. First, the R–D model is not scalable with video contents. The model was developed based on the assumption that each video frame has similar coding complexity, resulting in similar video quality for each frame. Second, the R–D model does not exclude the bit counts used for coding overhead including video/frame syntax,

TABLE I
NOTATIONS

B_i	:	the total number of bits used for encoding frame i.
Q_i	:	the quantization level used for encoding frame i.
a_1	:	the first-order coefficient in rate distortion model.
a_1^i	:	the first-order coefficient in rate distortion model for the ith VO.
a_2	:	the second-order coefficient in rate distortion model.
a_2^i	:	the second-order coefficient in rate distortion model for the ith VO.
H_i	:	the bits used for header, motion vectors and shape information of frame i.
M_i	:	the MAD of frame i.
β_t	:	the remaining available bit count for encoding the subsequent frames at time t.
$\mathcal{T}$	:	the duration of the video sequence.
$\mathcal{I}$	:	the number of bits actually used for the first I frame.
r	:	the bit-rate for the sequence.
R_t	:	the target bit count for the P frame at time t.
N_t	:	the remaining number of P frames at time t.
A_t	:	the actual bits used for the P frame at time t.
S	:	the weighting factor in target bit estimation.
F_t	:	the current buffer fullness at time t.
γ	:	the buffer size.
r_f	:	the frame rate of the source video.
m	:	the safety margin.
C	:	the channel output rate.
V_t^i	:	the target bit budget for the ith VO at time t.
H_t^i	:	the overhead bit count for the ith VO at time t.
M_t^i	:	the MAD for the ith VO at time t.
α_i	:	the threshold for shape coding for the ith VO.

motion vectors, and shape information. The bits used for these nontexture information are usually a constant number, regardless of its associated texture information.

To address the above problems, we introduce two new parameters (i.e., MAD and nontexture overhead) into the second-order R–D model in (1). That is

$$\frac{B_i - H_i}{M_i} = a_1 \times Q_i^{-1} + a_2 \times Q_i^{-2}$$

where H_i is the bits used for header, motion vectors and shape information, M_i is the mean absolute difference (MAD), which is computed after the motion compensation for the luminance component (i.e., Y component). If a_1 and a_2 are known, B_i and Q_i can be obtained based on the technique described in our previous work [4]. How to obtain a_1 and a_2 will be described in Section IV-E-1.

The proposed R–D model is scalable with video contents since it uses the index of video coding complexity such as MAD. In addition, the proposed R–D model is more accurate due to the exclusion of the constant overhead bits.

B. Initialization Stage

In the initialization stage, the major tasks the encoder has to complete with respect to the rate control include:

1) subtracting the bit counts for the first I frame from the total bit counts;
2) initializing the buffer size based on the latency requirement;
3) Initializing the buffer fullness in the middle level.

Without loss of generality, we assume that the video sequence is encoded in the order of the first I frame and subsequent P frames. At this stage, the encoder encodes the first I frame using the initial quantization parameter (QP) value, which is specified as an input parameter. Then the remaining available bits for encoding the subsequent P frames can be calculated by

$$\beta_t = \mathcal{T} \times r - \mathcal{I}$$

where

- β_t remaining available bit count for encoding the subsequent P frames at the coding time instant t;
- $\mathcal{T}$ duration of the video sequence (in seconds);
- r bit rate for the sequence (in bits per second);
- $\mathcal{I}$ number of bits actually used for the first I frame.

Thus, the channel output rate is β_0/N_0, where N_0 is the number of P frames in the sequence or in a GOP.

The setting of buffer size is based on the latency requirement specified by the user. The default buffer size is set to $r \times 0.5$ (i.e., the maximum accumulated delay is 500 ms). The initial buffer fullness is set at the middle level of the buffer (i.e., $r \times 0.25$) for better buffer management.

C. Pre-Encoding Stage

In the pre-encoding stage, the tasks of the rate control scheme include: 1) estimation of the target bits; 2) further adjustment of the target bits based on the buffer status for each VO; and 3) quantization parameter calculation.

The target bit count is estimated in the following phases: 1) frame-level bit-rate estimation; 2) object-level bit-rate estimation if desired; and 3) macroblock-level bit-rate estimation if

desired. At the frame-level, the target bit count for a P frame at time $(t+1)$, R_{t+1}, is estimated by

$$R_{t+1} = \frac{\beta_t}{N_t} \times (1-S) + A_t \times S$$

where N_t is the remaining number of P frames at time t, A_t is the actual bits used for the P frame at time t (i.e., the previous P frame). Note that S is the weighting factor to determine the impact of the previous frame on the target bit estimation of the current frame, which can either be determined dynamically or set to a constant number. The default value of S is 0.05 in our experiments.

To get a better target bit-rate estimation, we need to consider buffer fullness. Hence the target bit-rate estimation can be further adjusted with the following equation:

$$R_t := \frac{F_t + 2 \times (\gamma - F_t)}{2 \times F_t + (\gamma - F_t)} \times R_t \tag{2}$$

where F_t is the current buffer fullness at time t and γ is the buffer size. The adjustment in (2) is to keep the buffer fullness at the middle level to reduce the chance of buffer overflow or underflow. To achieve the constant video quality for each video frame or VO, the encoder must allocate a minimum number of bits, which is denoted as r/r_f, where r and r_f are the application's bit rate and frame rate of the source video, respectively. That is

$$R_t := \max\left\{\frac{r}{r_f}, R_t\right\}.$$

Then the final adjustment is made to predict the impact of R_t on the future buffer fullness. To be specific, a safety margin is employed to avoid the potential buffer overflow or underflow. We denote the safety margin as m, which is preset before encoding. To avoid buffer overflow, if $(R_t + F_t) > (1-m) \times \gamma$, then the target bit-rate is decreased and becomes

$$R_t = (1-m) \times \gamma - F_t.$$

On the other hand, to avoid buffer underflow, if $(R_t + F_t - C) < m \times \gamma$, then the target bit-rate is increased and becomes

$$R_t = C - F_t + m \times \gamma$$

where $C = \beta_0 / N_0$ is the channel output rate.

In the case of multiple VOs, we use the following dynamic target bit allocation besides the above frame-level bit allocation.

C.1) Dynamic Target Bit-Rate Distribution Among VOs: A straightforward way to allocate target bit-rate for each VO is to give a certain fixed number of bits to each VO without considering its complexity and perceptual importance. Obviously, this simple scheme has some serious drawbacks. For example, it is possible that the background VO may have bits left unused while the foreground VO requires more.

We propose a new bit allocation method for multiple VOs as follows. Based on the coding complexity and perceptual importance, we let the distribution of the bit budget be proportional to the square of the MAD of a VO, which is obtained empirically. That is, given the target bit budget R_t at the frame-level, the target bit budget V_t^i for the ith VO at time t is

$$V_t^i = K_t^i \times R_t$$

where

$$K_t^i = \frac{\mathrm{MAD}_t^i \times \mathrm{MAD}_t^i}{\sum_{k=1}^{n} \mathrm{MAD}_t^k \times \mathrm{MAD}_t^k}$$

and n is the number of VOs in the coding frame at time t. The proposed method is capable of adaptively adjusting the bit budget for each VO with respect to content complexity and perceptual importance.

In addition to distribution of bit budget among VOs in the spatial domain, one may also consider the composition of VOs in the temporal domain. Since each VO has different coding complexity (e.g., low or high motion), it is straightforward to encode the VOs at different frame rate for better coding efficiency. However, our experiments show that there is significant quality deterioration in the "gluing" boundary of VO. Thus, encoding the VOs at the same frame rate is chosen for our rate control algorithm.

After the target bit budget R_t or V_t^i has been obtained, the required quantization parameter for the frame or the ith VO can be calculated through using the technique described in our previous work [4].

D. Encoding Stage

At the encoding stage, the encoder has to complete the following major tasks: 1) encoding the video frame (object) and recording all actual bit-rate and 2) activating the macroblock-layer rate control if desired.

If either the frame- or object-level rate control is activated, the encoder compresses each video frame or video object using QP as computed in the pre-encoding stage. However, some low-delay applications may require strict buffer regulations, less accumulated delay, and perceptual-based quantization scheme. A macroblock-level rate control is necessary but costly at low rate since there is additional overhead if quantization parameter is changed within a frame. For instance, in MPEG-4, the macroblock (MB) type requires three more bits to indicate the existence of the differential quantization parameter (i.e., dquant). Furthermore, two bits need to be sent for dquant. For the same prediction mode, an additional five bits are required for transmission in order to change QP. In the case of encoding at 10 kbits/s, 7.5 fps, QCIF resolution, the overhead can be as high as $99 \times 5 \times 7.5 = 3.7$ kbits/s. If only 33 MBs are encoded, the overhead is $33 \times 5 \times 7.5 = 1.2$ kbits/s. Thus, there will be about 10% loss in compression efficiency at low bit-rate encoding. At high bit-rate, the overhead bit count becomes less significant than the residual bit count.

E. Post-Encoding Stage

In the post-encoding stage, the encoder needs to complete the following tasks: 1) updating the correspondent quadratic R-D

model for the entire frame or an individual VO; 2) performing the shape-threshold control to balance the bit usage between shape information and texture information; and 3) performing the frame-skipping control to prevent the potential buffer overflow or underflow.

1) R–D Model Update: After the encoding stage, the encoder has to update each VO's respective R–D model based on the following formula. For the ith VO

$$\frac{V_t^i - H_t^i}{M_t^i} = \frac{a_1^i}{Q_t^i} + \frac{a_2^i}{(Q_t^i)^2}$$

where

V_t^i actual bit count used for the ith VO at time t;
H_t^i overhead bit count used for syntax, motion and shape coding for the ith VO at time t;
M_t^i MAD for the ith VO at time t.

Note that in the case of MB rate control, the quantization parameter is defined as the average of all encoded MBs. To make our R–D model more accurate to reflect the video contents, the R–D model updating process consists of the following three steps.

Step 1)—Selection of Data Points: The data points selected will be used as the data set to update the R–D model. The accuracy of the model depends on the quality and quantity of the data set. To address this issue, we use a sliding-window based data selection mechanism.

If the complexity changes significantly (i.e., high motion scenes), a smaller window with more recent data points is used. By using such a mechanism, the encoder is able to intelligently select those representative data points for R–D model updates. The selection of data points is based on the ratio of "scene change" between the current frame (object) and the previous encoded frame (object). To quantify the amount of scene change, various indices such as MAD or SAD, or their combinations can be used. A sophisticated weighting factor can also be considered.

For the sake of lower implementation complexity, we only use MAD as an index to quantify the amount of scene change in our rate-control scheme. More specifically, if one segment of the video content tends to have higher motion scene (i.e., increasing coding complexity), then a smaller number of data points with recent data are selected. On the other hand, for video content with a lower motion scene, a larger number of data points with historic data are selected. Algorithmically, we have Size of sliding window $:= (\mathrm{MAD}_t/\mathrm{MAD}_{t-1}) \times$ MAX_SLIDING_WINDOW if $\mathrm{MAD}_{t-1} > \mathrm{MAD}_t$; otherwise, Size of sliding window $:= (\mathrm{MAD}_{t-1}/\mathrm{MAD}_t) \times$ MAX_SLIDING_WINDOW, where t is a time instant of coding and MAX_SLIDING_WINDOW is a preset constant (e.g., 20 in our experiments).

Due to the introduction of MAD, the proposed sliding-window mechanism is able to adaptively smooth the impact of scene change and helps to improve the quality of the data set, resulting in a more accurate model.

Step 2)—Calculation of the Model Parameters: Based on a_1 and a_2, the target bit-rate can be calculated for each data point within the sliding window obtained in Step 1. For those selected data points, the encoder collects quantization levels and actual bit-rate statistics. Using the linear regression technique, the two model parameters a_1 and a_2 can be obtained as follows:

$$a_2 = \frac{n\sum_{i=1}^{n}\frac{B_i - H_i}{M_i} - \left(\sum_{i=1}^{n} Q_i^{-1}\right)\left(\sum_{i=1}^{n}\frac{Q_i \times (B_i - H_i)}{M_i}\right)}{n\sum_{i=1}^{n} Q_i^{-2} - \left(\sum_{i=1}^{n} Q_i^{-1}\right)^2}$$

and

$$a_1 = \frac{\sum_{i=1}^{n}\frac{Q_i \times (B_i - H_i)}{M_i} - a_2 \times Q_i^{-1}}{n}$$

where n is the number of selected past frames, Q_i and B_i are the actual average quantization levels and actual bit counts in the past, respectively.

Step 3)—Removal of the Outliers from the Data Set: After the new model parameters a_1 and a_2 are derived, the encoder performs further refinement step to remove some bad data points. The bad data points are defined, in the statistical sense, as those data points whose difference between the actual bit budget and the target bit budget is larger than k standard deviation (e.g., $k = 1$ in our experiments). The target bit budget is recalculated based on the new model parameters obtained in Step 2, i.e., the actual bit budget B_i and the average quantization level Q_i. Thus, better model parameter updating can be achieved through the proposed adaptive data-point selection criterion.

2) Shape-Threshold Control: Since the bit budget is limited, it is essential to develop an efficient and effective way to allocate the limited bit budget for coding both shape and texture information in object-based video coding.

In MPEG-4, there are two ways to control the bit count used for the shape information: size-conversion process and shape-threshold setting. The size-conversion process adopted by MPEG-4 standard is used to reduce the amount of shape information for rate control and can be carried out on an MB basis [13]. On the other hand, the shape-threshold setting is a controllable parameter and is carried out on an object basis. The threshold value could be either a constant or dynamically changing.

In this paper, we propose an adaptive threshold shape control as follows. For a VO, if the actual bit count used for the nontexture information (e.g., shape information) exceeds its estimated bit budget, the encoder will increase the threshold value to reduce the bit count for nontexture coding at the expense of shape accuracy of a VO. Otherwise, the encoder decreases the threshold value to get better video shape accuracy. Initially, the threshold, α_i, for the ith VO is set to zero. Then it is increased by α_{step} if $H_i (= \mathrm{syntax}_i + \mathrm{motion}_i + \mathrm{shape}_i$ bits actually used for the ith VO in the previous frame) is greater than or equal to the target bit budget for the ith VO in a frame, V_i. Otherwise, it is decreased by α_{step}. To maintain the accuracy of the video shape to a certain degree (i.e, to avoid a negative threshold or an excessively large α), the α_i is bounded between 0 and $\alpha_{\max}$. The shape-threshold control mechanism is described as follows.

Algorithm 2 (Shape-Threshold Control)
if (syntax$_i$ + motion$_i$ + shape$_i$ $\geq V_i$) then
$\alpha_i := \max\{\alpha_{\max}\alpha_i + \alpha_{\text{step}}\}$
else
$\alpha_i := \min\{0, \alpha_i - \alpha_{\text{step}}\}$.

The proposed shape-threshold control mechanism is capable of adapting to the content and achieving good shape accuracy and texture quality.

3) Frame-Skipping Control: The objective of the frame-skipping control is to prevent buffer overflow. Once the encoder predicts that encoding the next frame would cause the buffer overflow, the encoder skips the encoding of the next frame. The buffer fullness will be decreased at the expense of lower frame rate. Although the frame skipping is an effective way to prevent buffer overflow, the overall perceptual quality may be reduced substantially, especially for contiguously frame skipping. To avoid the problem associated with contiguous frame skipping, we propose a frame-skipping mechanism as follows.

Before encoding the next frame, the encoder first examines the current buffer fullness and the estimated target bit-rate for the next frame. If the current buffer-fullness parameter plus the estimated frame bits for the next frame is larger than some predetermined threshold, called the safety margin (e.g., 80% of the buffer size), the next frame will be skipped. Note that the use of safety margin can reduce the contiguous frame skipping and can even be changed adaptively based on the coding context.

In the proposed predictive frame-skipping control, we use the actual bit count for the last frame. If the sum of the current buffer fullness (BufferFullness) and the actual bit count for the last frame minus channel output during a frame interval exceeds the safety margin of the buffer, we skip the next frame. After frame skipping, the buffer fullness is decreased by the channel output during a frame interval. The frame-skipping condition can be formulated as follows.

Algorithm 3 (Frame-Skipping Control Algorithm)
while ((BufferFullness + ActualBitCountsFor-
LastFrame
− ChannelOutputRate × FrameTimeInterval)
≥ BufferSize × SkipMargin) {
Skip the next frame;
BufferFullness := BufferFullness −
ChannelOutputRate × FrameTimeInterval.
}

The proposed frame skipping control helps to prevent buffer overflow and achieve graceful degradation of perceptual quality.

V. A Packetization Algorithm

Before the compressed video stream is transported over the packet-switched IP networks, it has to be packetized. At the sender side, the raw video is first compressed through using the encoding algorithm described in Section IV. Then the compressed MPEG-4 video stream is packetized at SL with timing and synchronization information, as well as fragmentation and random access information, before transferred to the TransMux Layer.

To date, the packetization process for MPEG-4 video ES at the SL has not been adequately addressed [20]. An appropriate packetization algorithm at this layer is essential for the efficient and robust transport of MPEG-4 video over the Internet. There are three packetization schemes in the literature. Le Leannec and Guillemot [15] used a fixed packet size for MPEG-4 video stream. Although this packetization scheme is very simple, an MB may be split into two packets, resulting dependency between two packets. Turletti and Huitema [26] proposed to use an MB as a packet. Under their scheme, no MB is split. Thus, loss of a packet only corrupts one MB, which enhances the error resilient capability of the video. For this reason, this packetization scheme was recommended by the Internet Engineering Task Force (IETF) [26]. To increase the efficiency, Zhu [35] proposed to use a GOB as a packet. Under such scheme, no GOB is split. Thus, loss of a packet only corrupts one GOB, which enhances the error resilient capability of the video. Therefore, Zhu's packetization scheme was also recommended by the IETF [35]. Different from the above work, we take advantage of the VOP property in MPEG-4 and design a SL packetization algorithm that offers both efficiency and robustness for Internet transport.

It is clear that the use of large packet size will reduce the total number of generated packets and overhead.[4] On the other hand, the packet size cannot be larger than the path maximum transit unit (MTU), which is defined to be the minimum of the MTUs along all the traversing links from the source to the destination. This is because that any packet larger than path MTU will result in IP fragmentation, which brings overhead for each fragmented packet. To make things worse, loss of one fragment packet will corrupt other fragment packets from the same original packet. Furthermore, for MPEG-4 video, we do not advise to packetize the data that contain information across two VOPs, since loss of such a packet will corrupt both VOPs. With these considerations, we choose packet size to be the minimum of the current VOP size and the path MTU. The path MTU can be found through the mechanism proposed by Mogul and Deering [17]. In the case when path MTU information is not available, the default MTU, i.e., 576 bytes, will be used.

When a VOP is too large to fit into a single packet, it is necessary to break it up into multiple segments and use multiple packets. We try to minimize both the number of packets generated for a given MPEG-4 bit-stream and the dependency between adjacent packets. The motivation for minimizing the number of packets is to minimize overhead while the motivation for minimizing the dependency between adjacent packets is to achieve robustness. If the MPEG-4 VOP header information is copied into each packet, such dependency among the packets can be removed. Since the size of an MB is always less than the path MTU, a packet should be composed of at least one MB.

Our packetization strategy is the following. If a complete VOP fits into a packet, then packetize such VOP with a single packet. Otherwise, we will try to packetize as many MBs as possible into a packet (with VOP header information copied into

[4]The overhead is 50-bytes long, which consists of 3 bytes of SL header, 3 bytes of FlexMux header, 16 bytes of RTP header, 8 bytes of UDP header, and 20 bytes of IP header [20].

TABLE II
SIMULATION PARAMETERS

End System	MPEG-4	MaxPL	4208 bits
		IR	10 Kbps
		AIR	0.5 Kbps
		PR	200 Kbps
		MR	5 Kbps
		α	0.95
		N_s	79
		N_r	25
		$P_{threshold}$	5%
		Buffer Size	1 Mbytes
	TCP	Mean Packet Processing Delay	300 μs
		Packet Processing Delay Variation	10 μs
		Packet Size	576 bytes
		Maximum Receiver Window Size	64K bytes
		Default Timeout	500 ms
		Timer Granularity	500 ms
		TCP Version	Reno
Switch		Buffer Size	10 Kbytes
		Packet Processing Delay	4 μs
		Buffer Management	Tail Dropping
Link	End System to Switch	Link Speed	10 Mbps
		Distance	1 km
	Switch to Switch	Distance	1000 km

each packet for the same VOP) without crossing over into the next VOP even if space is available in the last packet for the current VOP, i.e., MBs from consecutive VOPs are never put into the same packet. Our packetization method achieves both efficiency, which is essential for low bit-rate coding, and robustness to packet loss (due to strict boundary between VOPs among packets and copying of VOP header information into packets for the same VOP).

We first describe the functions and parameters used in our packetization algorithm.

1) BitCount is a counter that registers the number of bits read for current packetization process.
2) MaxPL, or Maximum payload length (in bits), equals to (path MTU $-$ 50 bytes) $\times$ 8.
3) VOP_start_code is a predefined code at the beginning of a VOP and is regarded as the boundary between two consecutive VOPs.

Our SL packetization algorithm is shown as follows.

```
Algorithm 4 (A Packetization Algorithm)
while (there is encoded data to be packetized)
  {
    search for next VOP_start_code and BitCount
  counts
    the number of bits of the video stream;
    if [(next VOP_start_code is found) and
  (BitCount −
       length of VOP_start_code ≤ MaxPL)] {
       /* Packetize by VOP boundary */
       packetize the bits before next
  VOP_start_code;
    }
   else if (BitCount − length of
  VOP_start_code >
     MaxPL) {
       /* Packetize by MBs. */
       Packetize as many MBs as possible
  without
       exceeding MaxPL and without crossing
  into
       next VOP;
       }
     else { /* Next VOP_start_code is not
  found, i.e.,
       end of video. */
       Packetize the remaining data.
       }
  }
```

Algorithm 4 starts with checking if there is encoded data to be packetized. First, we test if the VOP being read can be contained into a packet with the size no larger than MaxPL. If yes, the data being read will be passed to the next stage, i.e., RTP Packer for packetization. Otherwise, the algorithm goes to the MB level, that is, packetization is processed on the boundary of MBs. If VOP_start_code is not found and the number of bits read is less than MaxPL, which means reaching the end of the video stream, we packetize the remaining data.

Since a VOP is larger than an MB or a GOB, Algorithm 4 achieves higher efficiency than that of Turletti and Huitema [26] and that of Zhu [35]. Algorithm 4 also removes dependency between packets, which is the problem of the scheme by Le Leannec and Guillemot [15].

(a) (b)

Fig. 7. Video objects: (a) VO1 and (b) VO2 in "Akiyo" MPEG-4 video sequence.

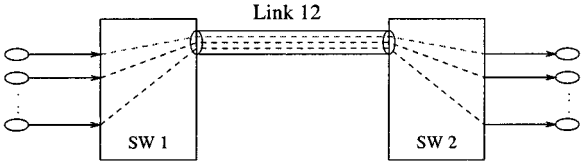


Fig. 8. A peer-to-peer network.

VI. Simulation Results

In this section, we implement our proposed architecture and algorithms on our network simulator and perform a simulation study on transporting MPEG-4 video over various benchmark network configurations. The purpose of this section is to demonstrate that our architecture and algorithms can 1) transport MPEG-4 video streams over the network with good perceptual picture quality under both low bit-rate and varying network conditions and 2) adapt to available network bandwidth and utilize it efficiently.

A. Simulation Settings

The network configurations that we use are the *peer-to-peer*, *parking lot*, and the *chain* network configurations. These network configurations have been used as standard configurations to test transport protocols in networking research community.

At the source side, we use the standard raw video sequence "Akiyo" in QCIF format for the MPEG-4 video encoder. The encoder performs MPEG-4 coding described in Section IV and adaptively adjusts its rate under our feedback control algorithm (Algorithm 1). The encoded bit-stream is packetized with our packetization algorithm (Algorithm 4) as well as RTP/UDP/IP protocol before being sent to the network. Packets may be dropped due to congestion in the network. For arriving packets, the receiver extracts the packet content to form the bit-stream for the MPEG-4 decoder. For a lost packet, the VOP associated with the lost packet will be discarded and a previous VOP will be copied over. For error control purpose, the source encoder encodes an Intra-VOP every 100 frames.

Table II lists the parameters used in our simulation. We use 576 bytes for the path MTU. Therefore, the maximum payload length, MaxPL, is 526 bytes (576 bytes minus 50 bytes of overhead) [20].

We run our simulation for 450 s for all configurations. Since there is only 300 continuous frames in "Akiyo" sequence available, we repeat the video sequence cyclically during the 450-s simulation run. In the simulations, we identify two VOs as VO1 (background) and VO2 (foreground) (see Fig. 7) and encoded them separately.

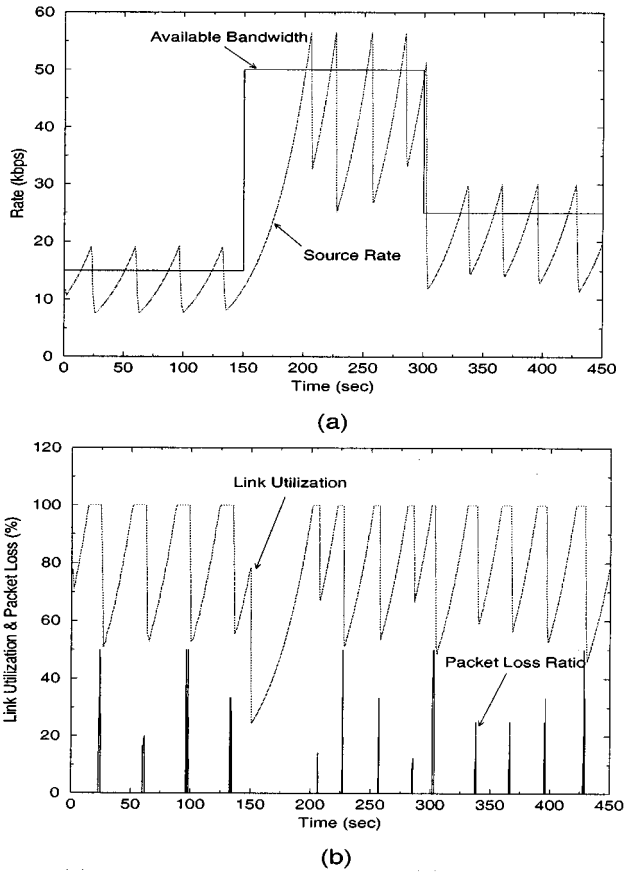


Fig. 9. (a) Source rate and link capacity. (b) Link utilization and packet loss ratio under peer-to-peer network.

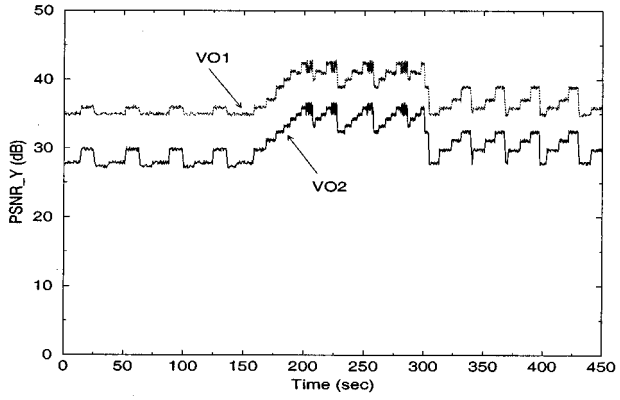


Fig. 10. PSNR of VOs at the receiver under peer-to-peer network.

B. Performance Under the Peer-to-Peer Configuration

We employ the standard peer-to-peer benchmark network configuration shown in Fig. 8 for the Internet environment (Fig. 1). We emphasize that such simple network configuration captures the fundamental property of a transport path within the Internet cloud since there is only one bottleneck link (i.e., the one with minimum bandwidth among all the traversing links) between the sender and the receiver. Furthermore, we stress that despite the multi-path and thus arriving packets out of sequence

(a) (b) (c)

Fig. 11. (a) Sample frame during [0, 150) s. (b) Sample frame during [150, 300) s. (c) Sample frame during [300, 450] s under peer-to-peer network.

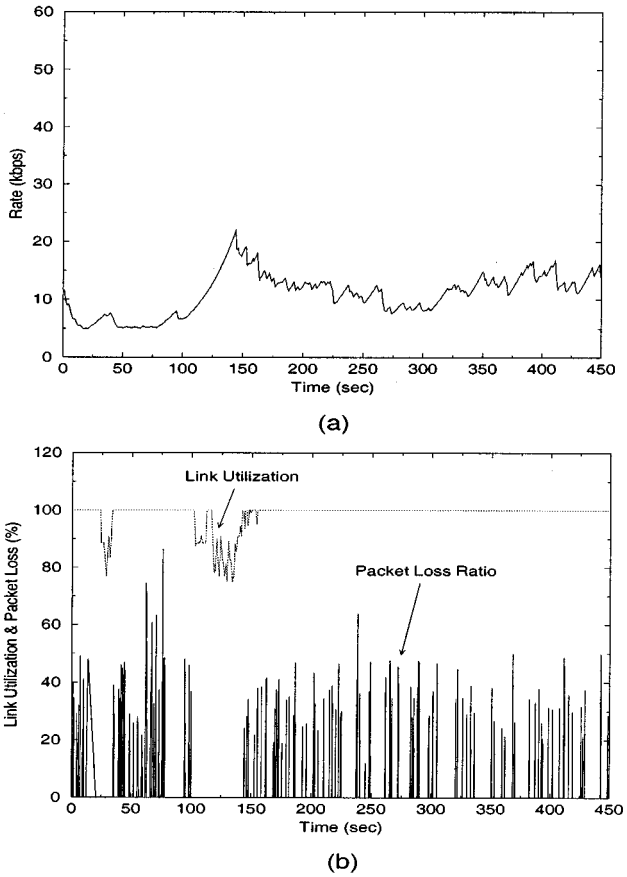


Fig. 12. (a) Source rate. (b) Link utilization and packet loss ratio under peer-to-peer network.

problem in the Internet, the validity and generality of our findings will not be compromised by the simple peer-to-peer network configuration since our architecture and algorithms are designed and implemented entirely on the end systems (sender and receiver). Therefore, a packet arriving after the threshold due to multi-path routing can just be treated as a lost packet at the destination and our architecture and algorithms remain intact under such scenario.

We organize our presentation as follows. Section VI-B-1 shows the performance of an MPEG-4 video under varying network bandwidth. In Section VI-B-2, we let MPEG-4 interact with competing TCP connections and show its performance.

1) MPEG-4 Video Under Varying Network Bandwidth: In this simulation, we only activate one MPEG-4 source under the peer-to-peer configuration (Fig. 8). The link capacity between the SW1 and SW2 varies from 15 kbits/s during [0, 150) s to 50 kbits/s during [150, 300) s to 25 kbits/s after 300 s (see Fig. 9).

Fig. 9(a) shows the network link bandwidth and source-rate behavior during the 450-s simulation run. We find that the source is able to adjust its output rate to keep track of the varying available network bandwidth. Fig. 9(b) shows the link utilization and packet loss ratio during the same simulation run, which is consistent with that shown in Fig. 9(a). The oscillation in source rate [Fig. 9(a)] and network utilization [Fig. 9(b)] are due to the propagation delay of the links and the binary nature of our feedback control algorithm. The source performs additive rate increase until it reaches the available link bandwidth. After that it overshoots it and results in congestion and packet loss. The packet loss is detected at the receiver and such information is conveyed to the source. Upon receiving such feedback, the source decreases its rate. Despite the oscillations, the average utilization of the bottleneck link is over 80%, which is a reasonably good result for feedback control in a wide area Internet (the inter-switch distance between SW1 and SW2 is 1000 km). Furthermore, we find that the average packet loss ratio is only 0.34%, which demonstrates the effectiveness of our feedback control algorithm.

A measure of the difference between the original video sequence and the received video sequence is the peak signal-to-noise (PSNR). Fig. 10 shows the PSNR of Y component of the MPEG-4 video at the receiver for the same simulation run as in Fig. 9. Fig. 10 is obtained by going through the following steps. First, the video sequence is reconstructed at the destination, where our simple error-concealment mechanism (i.e., copying previous VOP) is performed to conceal the effects of packet loss. Then, the PSNR is calculated for each reconstructed frame and plotted versus time.

To examine the perceptual quality of the MPEG-4 video, we play out the decoded video sequence at the receiver. Fig. 11 shows sample video frames at the receiver during [0, 150), [150, 300), and [300, 450] s time interval, respectively. The sample frames shown in Fig. 11 all show the same scene. We find that the video quality under these three different bit rates are all reasonably good, indicating the effectiveness of our end-to-end transport architecture and algorithms.

2) Interaction with Competing TCP Connections: We set the link capacity between SW1 and SW2 to be constant at 100 kbits/s in the peer-to-peer network (Fig. 8). In addition to one MPEG-4 video source, we activate 10 TCP connections to share the link bandwidth with the MPEG-4 video.

Fig. 12(a) shows source-rate behavior during the 450-s simulation run. We find that the source is able to adjust the source rate to dynamically share network bandwidth with other TCP connections. Since the time interval for TCP window adjustment is much smaller than that for MPEG-4 encoder-rate adjustment, the TCP connections are able to adjust to any remaining network bandwidth much faster than MPEG-4 and fully utilize overall network bandwidth. Fig. 12(b) shows the link utilization at Link12, which is 100% most of the time. Fig. 13 shows the PSNR of Y component of the MPEG-4 video at the receiver for the same simulation run. The average packet loss ratio in Fig. 13 is 0.95% Also, we find that the perceptual picture quality of the video at receiver is good.

C. Performance Under the Parking Lot Configuration

The parking lot network that we use is shown in Fig. 14, where path G1 consists of multiple flows and traverse from the first switch (SW1) to the last switch (SW5), path G2 starts from SW2 and terminates at the last switch (SW5), and so forth. Clearly, Link45 is the potential bottleneck link for all flows.

In our simulations, path G1 consists of four MPEG-4 sources and one TCP connection while paths G2, G3, and G4 all consist of five TCP connections, respectively. The capacity of each link between the switches is 400 kbits/s. All the TCP sources are persistent during the whole simulation run.

Fig. 15(a) shows source-rate behavior of the four MPEG-4 sources during the 450 second simulation run. We find that the sources are able to adjust the rates to keep track of the varying available network bandwidth. Fig. 15(b) shows the link utilization and packet-loss ratio of an MPEG-4 source during the same simulation run. The bottleneck link (Link45) is 99.9% utilized and the packet loss ratios for the four MPEG-4 sources are very low (with an average of 1.7%, 1.4%, 1.8%, and 1.5%, respectively). Fig. 16 shows the PSNR for the Y component of each VOs in one MPEG-4 video sequence at the receiver for the same simulation run.

D. Performance Under the Chain Configuration

The chain configuration that we use is shown in Fig. 17 where path G1 consisting of multiple flows and traverses from the first switch (SW1) to the last switch (SW4), while all the other paths traverse only one hop and "interfere" the flows in G1.

In our simulations, G1 consists of four MPEG-4 sources and one TCP connection while G2, G3 and G4 all consist of five TCP connections, respectively. The capacity of each link between the switches is 200 kbits/s. All the TCP sources are persistent during the whole simulation run.

Fig. 18(a) shows the encoding rates of the four MPEG-4 sources during the 450-s simulation run. We find that the sources are able to adjust the source rates based on the dynamic traffic behavior in the network. Fig. 18(b) shows the link utilization and packet loss ratio of an MPEG-4 source during the same simulation run. We find that the links are at least 98% utilized and the packet loss ratios for the four MPEG-4 sources are very small (with an average of 0.62%, 0.58%, 0.49%, and 0.46%, respectively). Fig. 19 shows the PSNR for

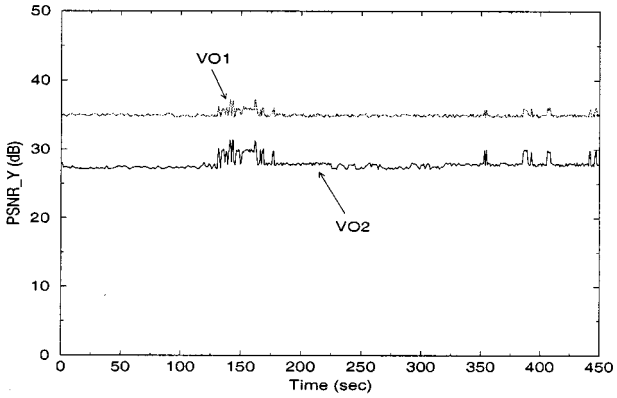


Fig. 13. PSNR of VOs at the receiver under peer-to-peer network.

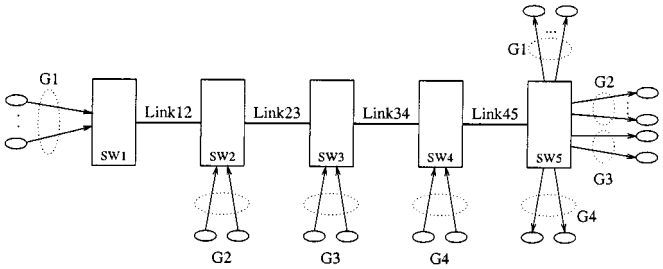


Fig. 14. A parking lot network.

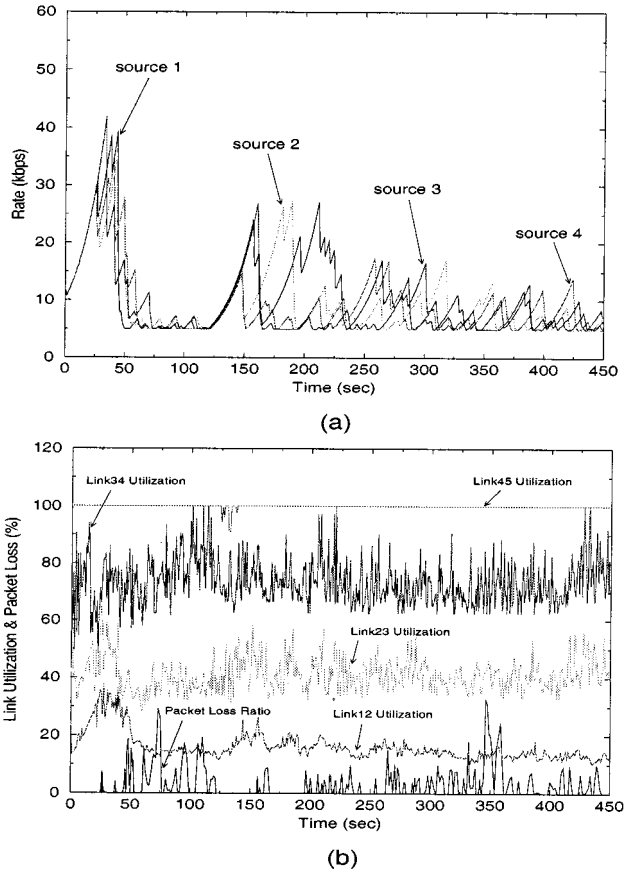


Fig. 15. (a) Source rates. (b) Link utilization and packet-loss ratio under parking lot network.

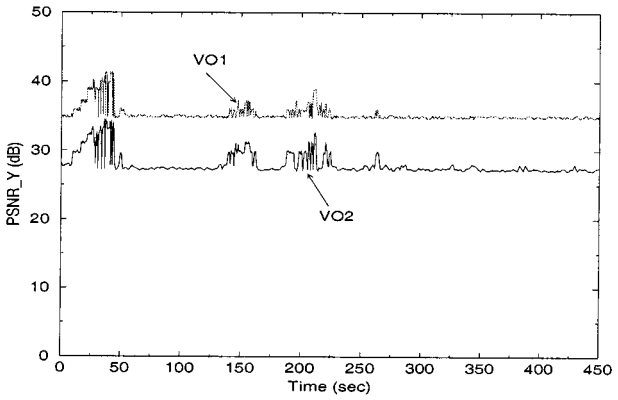


Fig. 16. PSNR of each VO of MPEG-4 video at receiver under parking lot network.

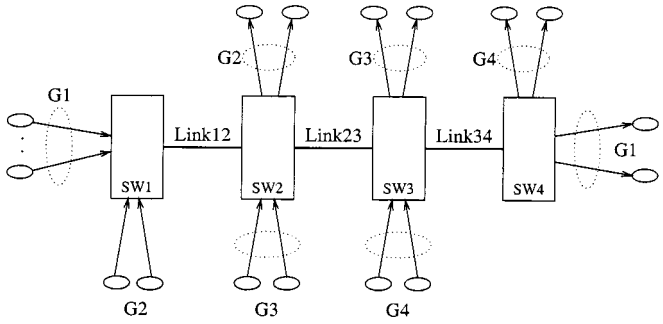


Fig. 17. A chain network.

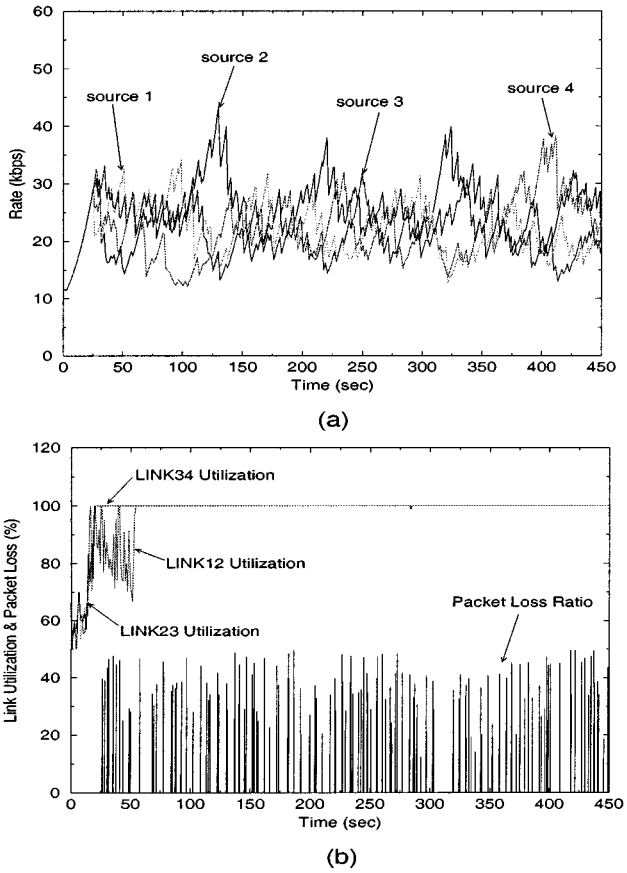


Fig. 18. (a) Source rates. (b) Link utilization and packet-loss ratio under chain network.

the Y component of each VO in one MPEG-4 sequence at the receiver for the same simulation run.

In summary, based on the extensive simulation results in this section, we conclude that our end-to-end transport architecture and algorithms can: 1) transport MPEG-4 video streams over the network with good perceptual picture quality under low bit-rate and varying network conditions and 2) adapt to available network bandwidth and utilize it efficiently.

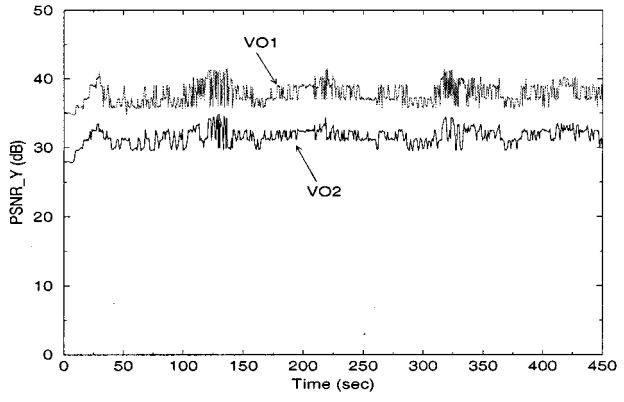


Fig. 19. PSNR of each VOs of the MPEG-4 video at the receiver under chain network.

VII. Concluding Remarks

The new MPEG-4 video standard has the potential of offering interactive content-based video services by using VO-based coding. Transporting MPEG-4 video is foreseen to be an important component of many multimedia applications. On the other hand, since the current Internet lacks QoS support and the available bandwidth, delay and loss vary over time, there remain many challenging problems in transporting MPEG-4 video with satisfactory video quality. To address these problems, this paper presents an end-to-end architecture for transporting MPEG-4 video over the Internet. The main contributions of this paper are listed as follows.

1) We outlined four key components in an end-to-end architecture for transporting MPEG-4 live video, which includes feedback control, source-rate adaptation, packetization, and error control. We stress that an architecture missing any of these components would not offer good performance for transporting MPEG-4 video over the Internet.
2) We presented an end-to-end feedback control algorithm employing RTCP feedback mechanism. We showed that our algorithm is capable of estimating available network bandwidth by measuring the packet loss ratio at the receiver. Since our feedback control algorithm is

implemented solely at end systems (source and destination), there is no additional requirement on Internet switches/routers.

3) We designed an encoding-rate control algorithm which is capable of adjusting the overall output rate of MPEG-4 video to the desired rate.
4) We designed a SL packetization algorithm for MPEG-4 video bit-streams. Our packetization algorithm was shown to achieve both efficiency and robustness for Internet MPEG-4 video.

Simulation results conclusively demonstrated that our proposed end-to-end transport architecture and algorithms for MPEG-4 are capable of providing good perceptual quality under low bit-rate and varying network conditions and utilizing network resources efficiently.

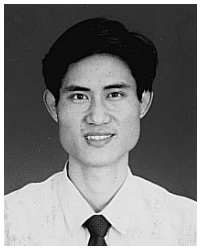
Our future work will focus on further extension of our architecture with greater performance and service support. One issue is the packet loss control and recovery associated with transporting MPEG-4 video. Another issue that needs to be addressed is the support of multicast for Internet video. Work is underway to extend our current transport architecture with multicast capability for MPEG-4 video.

ACKNOWLEDGMENT

The authors would like to thank Prof. Z.-L. Zhang of the University of Minnesota, Dr. Q.-F. Zhu of PictureTel Corporation, J. Huang of Polytechnic University, C. Albuquerque of the University of California at Irvine, W.-T. Tan of the University of California at Berkeley, R. Zhang of University of California at Santa Barbara, and Y. Wu of the University of Illinois at Urbana–Champaign for many fruitful discussions related to this work. Also, the authors wish to thank the anonymous reviewers for their constructive comments and suggestions that helped to improve the presentation of this paper.

REFERENCES

[1] J.-C. Bolot and T. Turletti, "Adaptive error control for packet video in the Internet," in *Proc. IEEE Int. Conf. Image Processing (ICIP'96)*, Lausanne, Switzerland, Sept. 1996.
[2] ——, "Experience with control mechanisms for packet video in the Internet," *ACM Comput. Commun. Rev.*, vol. 28, no. 1, Jan. 1998.
[3] C. Bormann, L. Cline, G. Deisher, T. Gardos, C. Maciocco, D. Newell, J. Ott, G. Sullivan, S. Wenger, and C. Zhu, "RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)," Internet Engineering Task Force, RFC 2429, Oct. 1998.
[4] T. Chiang and Y.-Q. Zhang, "A new rate control scheme using quadratic rate distortion model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 246–250, Feb. 1997.
[5] W. Dabbous, "Analysis of a delay-based congestion avoidance algorithm," in *Proc. 4th IFIP Conf. High-Performance Networking*, Dec. 1992.
[6] J. Danskin, G. Davis, and X. Song, "Fast lossy Internet image transmission," in *Proc. ACM Multimedia*, Nov. 1995.
[7] W. Ding and B. Liu, "Rate control of MPEG video coding and recording by rate-quantization modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 12–20, Feb. 1996.
[8] W. Ding, "Joint encoder and channel rate control of VBR video over ATM networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 266–278, Apr. 1997.
[9] A. Eleftheriadis and D. Anastassiou, "Meeting arbitrary QoS constraints using dynamic rate shaping of coded digital video," in *Proc. 5th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'95)*, April 1995, pp. 95–106.
[10] D. Hoffman, G. Fernando, and V. Goyal, "RTP payload format for MPEG1/MPEG2 video," Internet Engineering Task Force, RFC 2038, Oct. 1996.
[11] Y. T. Hou, S. S. Panwar, Z.-L. Zhang, H. Tzeng, and Y.-Q. Zhang, "Network bandwidth sharing for transporting rate-adaptive packet video using feedback," in *Proc. IEEE Global Telecommunications Conf. (GLOBECOM'98)*, Sydney, Australia, Nov. 8–12, 1998, pp. 1547–1555.
[12] C. Y. Hsu, A. Ortega, and A. R. Reibman, "Joint selection of source and channel rate for VBR transmission under ATM policing constraints," *IEEE J. Select. Areas Commun.*, vol. 15, pp. 1016–1028, Aug. 1997.
[13] ISO/IEC JTC 1/SC 29/WG 11, "Information technology—Coding of audio-visual objects, Part 1: Systems, Part 2: Visual, Part 3: Audio,", FCD 14 496, Dec. 1998.
[14] J. Lee and B. W. Dickenson, "Rate-distortion optimized frame type selection for MPEG encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 501–510, June 1997.
[15] F. Le Leannec and C. M. Guillemot, "Error resilient video transmission over the Internet," in *Proc. SPIE Visual Communications and Image Processing (VCIP'99)*, Jan. 1999.
[16] F. C. Martins, W. Ding, and E. Feig, "Joint control of spatial quantization and temporal sampling for very low bit-rate video," in *Proc. ICASSP'96*, vol. 4, May 1996, pp. 2072–2075.
[17] J. Mogul and S. Deering, "Path MTU discovery," Internet Engineering Task Force, RFC 1191, Nov. 1990.
[18] I. Rhee, "Error control techniques for interactive low-bit-rate video transmission over the Internet," in *Proc. ACM SIGCOMM*, Aug. 1998.
[19] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," Internet Engineering Task Force, RFC 1889, Jan. 1996.
[20] H. Schulzrinne, D. Hoffman, M. Speer, R. Civanlar, A. Basso, V. Balabanian, and C. Herpel, "RTP payload format for MPEG-4 elementary streams," Internet Engineering Task Force, Internet Draft, March 1998.
[21] R. Stedman, H. Gharavi, L. Hanzo, and R. Steele, "Transmission of subband-coded images via mobile channels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 15–26, Feb. 1993.
[22] H. Sun, W. Kwok, M. Chien, and C. H. J. Ju, "MPEG coding performance improvement by jointly optimizing coding mode decision and rate control," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 449–458, June 1997.
[23] R. Talluri, K. Oehler, T. Bannon, J. D. Courtney, A. Das, and J. Liao, "A robust, scalable, object-based video compression technique for very low bit-rate coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 221–233, Feb. 1997.
[24] R. Talluri, "Error-resilience video coding in the ISO MPEG-4 standard," *IEEE Commun. Mag.*, pp. 112–119, June 1998.
[25] T. Turletti and C. Huitema, "Videoconferencing on the Internet," *IEEE Trans. Networking*, vol. 4, pp. 340–351, June 1996.
[26] ——, "RTP payload format for H.261 video streams," Internet Engineering Task Force, RFC 2032, Oct. 1996.
[27] A. Vetro, H. Sun, and Y. Wang, "MPEG-4 rate control for multiple video objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 186–199, Feb. 1999.
[28] J. Villasenor, Y.-Q. Zhang, and J. Wen, "Robust video coding algorithms and systems," in *Proc. IEEE*, Oct. 1999, vol. 87, pp. 1724–1733.
[29] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communication: A review," *Proc. IEEE*, vol. 86, pp. 974–997, May 1998.
[30] T. Weigand, M. Lightstone, D. Mukherjee, T. G. Campbell, and S. K. Mitra, "Rate-distortion optimized mode selection for very low bit-rate video coding and the emerging H.263 standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 182–190, Apr. 1996.
[31] S. Wenger, G. Knorr, J. Ott, and F. Kossentini, "Error resilience support in H.263+," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 867–877, Nov. 1998.
[32] Y.-Q. Zhang, Y.-J. Liu, and R. Pickholtz, "Layered image transmission over cellular radio channels," *IEEE Trans. Veh. Technol.*, vol. 43, Aug. 1994.
[33] Y.-Q. Zhang and X. Lee, "Performance of MPEG codecs in the presence of errors," *J. Vis. Commun. Image Repres.*, vol. 5, pp. 379–387, Dec. 1994.
[34] Z.-L. Zhang, S. Nelakuditi, R. Aggarwa, and R. P. Tsang, "Efficient server selective frame discard algorithms for stored video delivery over resource constrained networks," in *Proc. IEEE INFOCOM*, Mar. 1999, pp. 472–479.
[35] C. Zhu, "RTP payload format for H.263 video streams," Internet Engineering Task Force, RFC 2190, Sept. 1997.



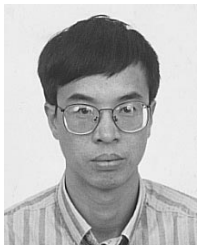
Dapeng Wu (S'98) received the B.E degree from Huazhong University of Science and Technology, Wuhan, China, and the M.E. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 1990 and 1997, respectively, both in electrical engineering. Since July 1997, he has been working toward the Ph.D. degree in electrical engineering at Polytechnic University, Brooklyn, NY.

During the summer of 1998 and part of 1999, he was with Fujitsu Laboratories of America, Sunnyvale, CA, conducting research on architectures and traffic management algorithms in integrated services (IntServ) networks and differentiated services (DiffServ) Internet for multimedia applications. His current interests are in the areas of next-generation Internet architecture, protocols, implementations for integrated and differentiated services, and rate control and error control for video streaming over the Internet.

Yiwei Thomas Hou (S'91–M'98) obtained the B.E. degree (*summa cum laude*) from the City College of New York in 1991, the M.S. degree from Columbia University, New York, in 1993, and the Ph.D. degree from Polytechnic University, Brooklyn, NY, in 1997, all in electrical engineering.

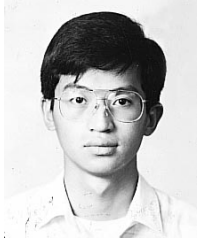
He was with AT&T Bell Labs, Murray Hill, NJ, during the summers of 1994 and 1995, working on implementations of IP and ATM inter-networking. He was with Bell Labs—Lucent Technologies, Holmdel, NJ, during the summer of 1996, working on network traffic management. Since September 1997, he has been a Researcher at Fujitsu Laboratories of America, Sunnyvale, CA. His current research interests are in the areas of quality of service (QoS) support for transporting multimedia applications over the Internet, and scalable architecture, protocols, and implementations for differentiated services.

Dr. Hou was awarded a National Science Foundation Graduate Research Traineeship for pursuing the Ph.D. degree in high-speed networking, and was recipient of Alexander Hessel Award for an outstanding Ph.D. dissertation in 1998 from Polytechnic University. He is a member of ACM and Sigma Xi.



Wenwu Zhu (S'92–M'97) received the B.E. and M.E. degrees from National University of Science and Technology, Changsha, China, in 1985 and 1988, respectively. He received the M.S. degree from Illinois Institute of Technology, Chicago, and the Ph.D. degree from Polytechnic University, Brooklyn, NY, in 1993 and 1996, respectively, both in electrical engineering.

From August 1988 to December 1990, he was with the Graduate School, University of Science and Technology of China (USTC), and Institute of Electronics, Academia Sinica (Chinese Academy of Sciences), Beijing. From July 1996 to October 1999, he was with Bell Labs—Lucent Technologies, Holmdel, NJ. He joined Microsoft Research, Beijing, China, in October 1999 as a Researcher.He has published over 50 papers in various referred conferences and journals. His current research interests are in the areas of video over IP and wireless networks, multimedia signal processing, and multimedia applications.



Hung-Ju Lee (M'96) received the B.S. degree from Tatung Institute of Technology, Taipei, Taiwan, in 1987, and the M.S. and Ph.D. degrees from the Texas A&M University, College Station, TX, in 1993 and 1996, respectively, all in computer science.

In 1996, he joined Sarnoff Corporation (formerly David Sarnoff Research Center), Princeton, NJ, as a Member of Technical Staff. He actively participates in ISO's MPEG digital-video standardization process, with particular focus on wavelet-based visual texture coding and scalable rate control for MPEG-4 video. His current research interests include image and video coding, and network resource management for multimedia applications.

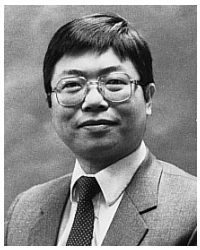
Dr. Lee received Sarnoff Technical Achievement Awards in 1998 for his contributions on the development of MPEG-4 rate control.

Tihao Chiang (S'91–M'95–SM'99) received the B.S. degree from the National Taiwan University, Taipei, Taiwan, in 1987, the M.S. and Ph.D. degrees from Columbia University, New York, in 1991 and 1995, respectively, all in electrical engineering.

He joined Sarnoff Corporation (formerly David Sarnoff Research Center), Princeton, NJ, in 1995 as a Member of Technical Staff, and later was promoted to Technology Leader and Program Manager. In September 1999, he joined the faculty at National Chiao-Tung University, Taiwan. Since 1992, he has actively participated in ISO's MPEG digital video coding standardization process, with particular focus on the scalability/compatibility issue. He is the co-chair for encoder optimization on the MPEG-4 committee. He has published over 30 technical journal and conference papers in the field of video and signal processing. His main research interests are compatible/scalable video compression, stereoscopic video coding, and motion estimation.

Ya-Qin Zhang (S'87–M'90–SM'93–F'97) joined Microsoft Research, Beijing, China, in January 1999 as the Assistant Managing Director of the company. He was previously the Director of Multimedia Technology Laboratory at Sarnoff Corporation (formerly David Sarnoff Research Center), Princeton, NJ. His laboratory is a world leader in MPEG2/DTV, MPEG4/VLBR, and multimedia information technologies. He was with GTE Laboratories Inc., Waltham, MA, and Contel Technology Center, VA, from 1989 to 1994. He has authored and co-authored over 150 refereed papers and 30 U.S. patents granted or pending in digital video, Internet multimedia, wireless and satellite communications. Many of the technologies he and his team developed have become the basis for start-up ventures, commercial products, and international standards. He has been an active contributor to the ISO/MPEG and ITU standardization efforts in digital video and multimedia.

Dr. Zhang was the Editor-In-Chief for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY from July 1997 to July 1999. He was a Guest Editor for the special issue on "Advances in Image and Video Compression" for the PROCEEDINGS OF THE IEEE (February 1995). He serves on the editorial boards of seven other professional journals and over a dozen conference committees. He received numerous awards, including several industry technical achievement awards and IEEE awards. He was awarded as the "Research Engineer of the Year" in 1998 by the Central Jersey Engineering Council for his "leadership and invention in communications technology, which has enabled dramatic advances in digital video compression and manipulation for broadcast and interactive television and networking applications."



H. Jonathan Chao (S'82–M'85–SM'95) received the B.S.E.E. and M.S.E.E. degrees from National Chiao Tung University, Taiwan, in 1977 and 1980, respectively, and the Ph.D. degree in electrical engineering from The Ohio State University, Columbus, OH, in 1985.

From 1985 to 1991, he was a Member of Technical Staff at Bellcore, NJ, where he conducted research in the area of SONET/ATM broadband networks and was involved in architecture designs and ASIC implementations, such as the first SONET-like Framer chip, ATM Layer chip, and Sequencer chip (the first chip-handling packet scheduling). Since January 1992, he has been a Professor in the Department of Electrical Engineering, Polytechnic University, Brooklyn, NY. His research interests include large-capacity packet switches and routers, packet scheduling and buffer management, and congestion flow control in IP/ATM networks.

Dr. Chao served as a Guest Editor for the *IEEE Journal on Selected Areas in Communications* special issues on "Advances in ATM Switching Systems for B-ISDN" (June 1997) and "Next Generation IP Switches and Routers" (June 1999). He is currently an Editor for IEEE/ACM TRANSACTIONS ON NETWORKING. He received Bellcore Excellence Award in 1987.