Broadcast Quality Video over IP

Sergio D. Servetto, *Member, IEEE*, and Klara Nahrstedt, *Member, IEEE*

***Abstract*—We consider the problem of designing systems for the transmission of video signals of the quality found in current television broadcasts, over high-speed segments of the public IP network. Our most important contribution is the definition of a network/coder interface for IP networks which gathers channel state information, and then sets parameters of the video coder to maximize the quality of the signal delivered to the receiver, while remaining fair to other data or video connections. This interface plays a role analogous to that of a Leaky Bucket controller, in that it specifies traffic shaping parameters which result in simultaneous good Quality-of-Service (QoS) for the source and good network performance. Since the network is not assumed to provide any form of QoS guarantee, fundamental to our construction is a hidden Markov model for the channel, based on which the interface solves a problem of optimal stochastic control, to decide how to configure the encoder. Other contributions are a) modifications to the standard Internet transport protocol, to make it suitable for the transport of delay-constrained traffic and to gather channel state information, and b) the design of an error-resilient video coder. Experimental studies reveal that the proposed system is able to stream video signals of the quality of current TV-broadcasts, among hosts in wide-area networks connected to the experimental vBNS backbone.**

***Index Terms*—Communication systems, computer networks, hidden Markov models, interactive TV, protocols, stochastic systems.**

I. Introduction

THE MAIN goal pursued in this paper is the design and implementation of a system to transmit in real time *high-quality* video signals over the public network. By high-quality we mean signals encoded at bit rates resulting in the reproduction quality attained by current TV-broadcasts, i.e., transmission of CCIR601 color signals at bit rates in the range 4–9 Mbits/s.

Systems for streaming video at such high bit rates are of interest, for example, to video-on-demand applications (such as web servers). Most existing streaming systems however typically operate at bit rates in the range of 40–50 Kbits/s, mainly because a significant portion of Internet users are connected via dialup lines whose capacity is limited to at most 56 Kbits/s (under the most optimistic assumptions). However, it is our belief that the limitations on channel capacity imposed by "the last mile" are only temporary. New technologies (e.g., DSL modems), promise the delivery of substantially higher data rates. Thus, from a research perspective, it makes sense to consider design issues for future systems free from the very low bit rate constraint. Furthermore, video coding at very low bit rates is a problem that has already received significant attention, for which a number of good solutions do already exist (e.g., H.263).

Note that this paper would be pointless if, provided with enough bandwidth, these existing low bit rate systems could be scaled up to stream high-quality signals. However, this is not the case. Streaming high-quality video poses technical challenges at all levels in a system design, and a new family of communication protocols and signal processing algorithms is required. Specifically, when freed from the low bit rate constraint, some new interesting problems are:

Communications and Control: What are realistic models for the network channel, that can be used by applications to effectively perform joint source/channel coding tasks? Given one such model, what are efficient algorithms to identify its parameters and complexity? How can channel states be estimated? What control strategies should be applied at the interface between the source and the channel, to ensure efficient usage of the network resources *and* a high quality signal reproduction?

Signal Processing: What are good ways of trading off compression efficiency and robustness to errors? This is particularly relevant in our context, since at middle/high bit rates some compression efficiency can often be sacrificed without a noticeable degradation of the decoded signal.

Information Theory: For each of the concrete problems discussed above there is a more fundamental version of the same problem. What are relevant properties of the channel under consideration? What is an appropriate definition of capacity for this channel? What performance bounds apply to coders that signal information across this channel?

It has been argued recently that "fifty years from now … telecommunications will likely be dominated by packet data/voice transmitted over wide-area networks like the Internet … the challenge will be to understand the network as a whole and to guarantee end-to-end quality of service" [2]. We believe the questions we pose (and start to answer) in this paper are very meaningful, well aligned with this vision.

A. Video Transmission over IP Networks

The problem of sending video over IP has essentially two main components: video data compression, and design of communication protocols, as illustrated Fig. 1.

1) Coding-Only View of the Problem: One popular approach consists of designing a low bit rate coder, possibly protecting

Manuscript received May 31, 2000; revised December 1, 2000. This work was supported in part by the Ray Ozzie Fellowship of the Department of Computer Science, University of Illinois at Urbana-Champaign, and by the National Science Foundation under Award CCR-96-23867 (CAREER) and Grant 9624396. Parts of this work were presented at ICIP'99. The associate editor coordinating the review of this paper and approving it for publication was Prof. Jenq-Neng Hwang.

S. D. Servetto was with Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA. He is now with the Laboratoire de Communications Audiovisuelles, Ecole Polytechnique Fédérale de Lausanne, IN-Ecublens, 1015 Lausanne, Switzerland.

K. Nahrstedt is with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA.

Publisher Item Identifier S 1520-9210(01)01861-2.

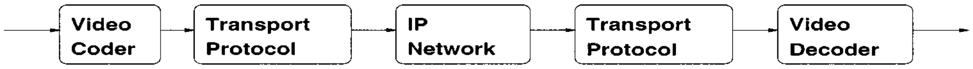

Fig. 1. Structure of a video streaming system.

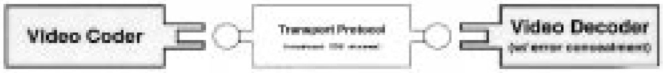

Fig. 2. Approach centered around coding problems. In this case, all the intelligence goes into the design of good data compression algorithms, but it is assumed that the network is a black-box—a fixed, standard pipe, with no differentiation of packets at the socket level or below.

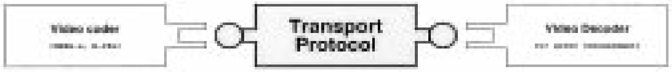

Fig. 3. Approach centered around the design of network protocols. In this case, all the intelligence goes into the design of good communication protocols, but it is assumed that the coder is a black-box—fixed, typically one of the standards (MPEG-x, H.26x).

the resulting bit stream with channel codes, and using one of the standard Internet transport protocols to transmit the resulting data stream. If the source bit rate is low enough and the channel is not too congested then it is possible to use TCP, in which case no errors occur and therefore there is no need for channel codes. Otherwise, UDP is used with a constant packet injection rate, and low-redundancy channel codes are used to protect against infrequent lost packets. Fig. 2 illustrates these concepts.

The main drawback of this approach is that, essentially, it does not deal well with the time-varying nature of the channel. To avoid having to deal with these time-variations, the channel is severely underutilized, by using a low bit rate coder [27]. This is because at higher injection rates, fluctuations in the packet loss rate make it very difficult to guarantee a low probability of decoding error. At higher bit rates, a careful matching of Reed–Solomon (RS) codes to the importance of different portions of an MPEG-2 stream has been proposed [1]. However, in this case it is unclear how to adjust the parameters of the RS codes for different channel conditions, leading to possibly overly conservative estimates of the required amount of redundancy. Furthermore, with UDP applications are responsible for performing their own congestion avoidance and control, which is a very bad idea, since potentially misbehaving sources may cause severe degradation of the overall network performance [7], [23], [37].

2) A Networking-Only View of the Problem: Another widely used approach consists of designing new transport protocols, but using either standard video coding algorithms which generate a fixed syntax for the compressed bit stream [4], or by using layered coding techniques [29], [38].

This approach has certain advantages over the one discussed above, the first and most obvious one being that flow control is part of the protocol, and therefore the resulting video flows are indeed good network citizens [29]. Besides, since the bit stream syntax is known, there are some games that can be played in the presence of congestion: for example, it is possible to put all the blocks along a given motion trajectory into a single packet, so that if this packet is lost the entire motion path is lost and therefore error propagation is limited [3]. Also, it is possible to retransmit packets selectively depending on, for example, whether a lost packet contains intra-coded blocks or not [9]. Fig. 3 illustrates these concepts.

The main drawback of this approach is that they are all limited in performance by the nature of the coders used. Video coders based on multiresolution techniques are inherently mismatched to a network which provides no form of packet differentiation. Plus, since the modified protocols cannot ensure error-free transmission, when errors do indeed occur, the quality of the decoded signals suffers severely because of lack of robustness in the coders used.

B. The Need for Joint Source/Channel Coding

Consider Fig. 1 again. One can easily recognize in that picture Shannon's structure for a communications system [33]: there is a module in charge of removing redundancy in the source (the video coder), there is a module in charge of protecting the data against possible channel impairments (the transmission protocol), there is a channel (the network), and there are modules that invert the previous steps. Under this light, it is clear that the approaches previously described are essentially based on the Source/Channel Separation Theorem [6, Ch. 8]: this is the first indicator of "trouble," since packet networks are perhaps the best example of a channel for which the hypotheses of the separation theorem do not apply. Even disregarding practical issues of complexity of the encoders/decoders, it is well known that the separation principle does *not* hold under delay constraints, under channel uncertainty, and for multiple access channels, all of which are present in a packet network. In our approach instead, we propose to design a data compression algorithm *jointly* with the transmission protocol that will be used to move the resulting compressed stream, as illustrated in Fig. 4.

To accomplish our goal, there are essentially two issues that we need to consider 1) we need to understand properties of the channel, to be able to design efficient methods for encoding and transmitting information across it and 2) we need to design an interface between the network and the coder, which can estimate and track channel state fluctuations, as well as present this information in a useful manner to the encoder.

C. Network/Coder Interface

The design of interfaces between networks and applications is a problem that has received significant attention. However, most of the work so far has focused on transmission and traffic regulation over asynchronous transfer mode (ATM) networks [8], [26], of which perhaps the simplest example is the Leaky Bucket controller [24]. Video over ATM was of interest because of two reasons: 1) ATM was thought of as one of the candidate transport technologies for future broadband integrated services data networks (B-ISDN) *all the way to the end user* and 2) ATM networks are able to provide Quality-of-Service (QoS) guarantees

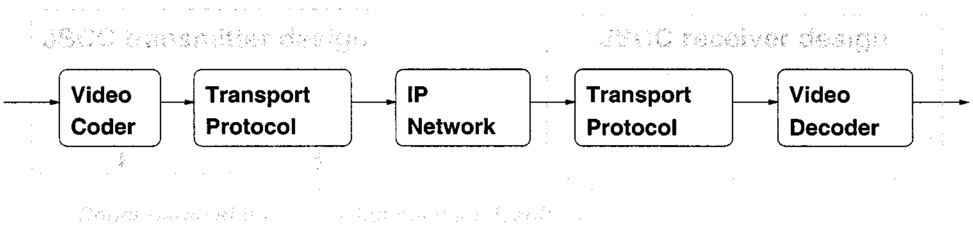

Fig. 4. Joint source/channel coding (JSCC) view of the problem. In this approach, the interactions between the coder and the protocol are explicitly taken into account.

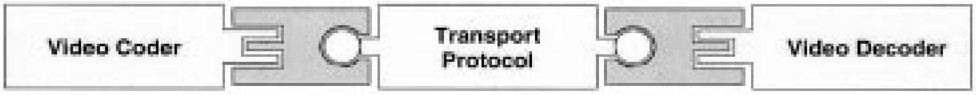

Fig. 5. Network/coder interface. In our approach, we design jointly the three most important components of our communications system: the source coder, the transport protocol, and the interface between these two. We claim that a clean interface among these components far outweighs the importance of fully optimizing each of the individual components independently of the others.

for applications. Work has been done on the problem of streaming video over ATM networks [22], [28], and more specifically and related to what we need, on the design of controllers which, sitting at the network/coder interface, set parameters of the encoder so as to maximize the quality of the video signal delivered to the receiver, subject to the constraint of complying with the traffic flow contract enforced by the network [13].

Current IP networks are inherently different from ATM networks however, in that they take a *best-effort* approach to packet transmission and routing, and hence in that no QoS guarantees whatsoever are provided.[1] As a result, there is no contract to be negotiated between the source and the network, and hence there are no policing mechanisms applied by the network at its interface with the source. The source is expected to inject packets in a "responsible" manner (e.g., using the flow control algorithm of TCP [14]), because this will result in good performance not only for itself but also for other sources. But there is no mechanism to prevent a source from injecting as many packets as it wants. This poses a complication in the design of controllers such as that of [13], but in the context of IP networks: if the network does not provide any *explicit* guidelines as to what the source should do (e.g., in the form of a contract, as in the case of ATM), then what is a good control strategy? How should the controller configure the encoder to ensure smooth and high-quality video playback, when there is uncertainty on key parameters of the communications channel, such as available bandwidth, propagation delay, and packet loss rate?

The answer to these questions can be found in the theory of control of stochastic dynamical systems, which deals precisely with problems of making decisions under uncertainty [16]. However, to be able to apply stochastic control principles, we need to impose a statistical model on the channel, based on which we can derive probability measures on the uncertain parameters.

D. Main Contributions and Organization of the Paper

A fundamental point we try to make in this paper is that there is an inherent mismatch between properties of state-of-the-art video coders and properties of existing transport protocols, and that these design tasks need to be addressed *jointly*, as illustrated in Fig. 5.

For this reason, we feel the main contribution presented in this work is the middleware module we put between the network and the coder: *we design a controller which adjusts key parameters of the error-resilient coder as a function of channel states*. Furthermore, in the process we present a solution to the problem of modeling the channel and of gathering channel state information: we use a hidden Markov model (HMM) to give a statistical description of the behavior of the channel, based on which we can design our controller applying standard stochastic control principles. Other contributions are the design of a TCP-friendly transport protocol (a simple modification of TCP/IP to make it suitable for the transport of delay-constrained data, and to make its internal state visible to upper system layers), and the design of an error-resilient video coder, along the lines of those designed in [32].

The rest of this paper is organized as follows. In Section II, we present modifications to the TCP protocol, as well as the design of an error-resilient video coder. In Section III, we present the design of the network/coder interface. In Section IV, we present performance results, and in Section V, we present conclusions and discuss issues for future work.

II. Transport and Coding

A. Joint Design of Coders and Transport Protocols

In this section, our goal is to define one transport protocol and one algorithm for compressing video signals with a clear and well defined interface.

- The transport protocol is defined such that 1) a hard upper bound can be given on the propagation delay of any one

[1] Note that there is an IETF effort which attempts to provide QoS guarantees in the context of IP networks (e.g., a number of RFCs on differentiated services approaches). However, the vast majority of high-speed public IP networks still provide *only* best effort service. This is true even for experimental networks, like vBNS. But even if such services eventually become massively available, pricing considerations and/or multiple users falling within a same class of service make the best-effort case a most important one to consider.

packet in the network and 2) a statistical description of the channel can be given, made available to applications so that these can be configured accordingly.

- The way in which 1) is ensured above is by eliminating retransmissions. Although the flow control algorithm ensures that packet losses are kept low, a nonnegligible number of packets will still be lost. As a result, the coder must be able to reconstruct high-quality video signals even when a small but arbitrary subset of its data is missing.

The idea that properties of the transported signal must affect the design of transport protocols was first defined in Clark and Tennenhouse's Application Level Framing (ALF) protocol architecture [5], whereas the idea that coders for this application need to be designed based on JSCC principles was first proposed by Garrett and Vetterli [11]. McCanne in his thesis developed the idea that ALF and JSCC are in fact parallel concepts [20]. Our ideas first appeared in [31], of which this paper is an extended version. Later (and independently), Lee *et al.* combined (in [17]) the error-resilience techniques of [21] with the transport protocol of [41]. That work differs from ours however in that, although the protocol and the coder used are based on similar principles (a TCP-like protocol and an error-resilient coder), they never consider the problem of designing a good interface between these two, which is the main topic of our work. Prior to our work the joint design of coders and transport protocols had also been considered by Garrett [10], Ortega [22] and McCanne [20]. In all cases however, the solutions obtained are based on *layered* (or multiresolution) coders: as a result, systems designed using these coders are completely different from systems designed using coders like ours, in which all data packets are equally important in terms of decoded signal quality.

B. RT-TCP: Real-Time TCP

1) Mechanics of RT-TCP: There are three basic modifications that we introduce to the standard TCP.

Elimination of Retransmissions: When the timer for a transmitted packet expires, instead of retransmitting this assumed lost packet (as specified by standard TCP), a new packet is sent.

Redefined Acknowledgment: In TCP, an acknowledged data segment implies all previous segments are available at the receiver [36]. Since without retransmissions some packets will necessarily be lost (because of a continuous probing of available channel capacity), the meaning of ack messages needs to be redefined: an acknowledgment explicitly states the packets received.

Extra Headers: Timestamps, sequence numbers, and other information needed for "booking" purposes are added to the headers.

The main rationale behind these modifications is that they represent the least set of changes needed to have TCP be efficient at carrying traffic with real-time delay constraints. The current Internet works under the assumption that all end-systems react to congestion by adjusting accordingly their packet injection rates. Doing so has a number of benefits, such as keeping network utilization high, or preventing congestion collapse. Another benefit is that of sharing in a fair manner the instantaneous available bandwidth among multiple competing flows, implemented using possibly different protocols. But since most current Internet traffic is based on TCP, it is crucial that any new protocol remains fair to TCP. Our modifications are perhaps the simplest way of accomplishing exactly that. Note however that prior to this work such an approach had not led to feasible system designs: without retransmissions—or else some other mechanism that can ensure error-free transmission—classical *layered* video coders take a severe performance hit in the presence of even a small amount of randomly placed packet losses (e.g., MPEG).

Note also that being able to use TCPs flow control as is (may be with nonessential changes) has a definite advantage: as far as the network is concerned, the traffic generated by such a source is no different from the traffic generated by bulk TCP transfers. We are motivated to take this "evolutionary" approach to the transport of delay-constrained data (as opposed to "revolutionary" approaches, in which entirely new protocols are built from scratch), by the simple practical observation that TCPs flow control has delivered outstanding network performance over many years, not just in simulations but in actual transmissions performed by a very large number of Internet users. Because of this, we feel it is a good idea to keep flow control as it is now and build applications on top, instead of replacing it with entirely new controllers.

2) A Hidden Markov Model for RT-TCP Channels: In order for upper layers in the communication architecture to use information about channel states, the first requirement is that of a model that can be used to "describe" the channel to these higher layers. The development of such models for RT-TCP connections is not a simple task: feedback in the flow control algorithm introduces memory and time-dependencies in the service rate of the channel, rendering inappropriate the use of simple parameterizations based on classical models (e.g., exponential service rate).

Inspired by work on models for flat fading channels [40], we propose here a Hidden Markov Model (HMM) to capture the statistical properties of RT-TCP channels.

- An RT-TCP connection is regarded as a source that generates $\mathbf{1}$ and $-\mathbf{1}$ symbols at random times: $\mathbf{1}$'s represent acknowledged packets, and $-\mathbf{1}$'s unacknowledged packets.
- Time is quantized into intervals of length T such that, over any one interval $[t, t+T)$, at most one symbol can be generated.[2]
- A discrete-time process $(X_n)_{n\geq 0}$ is defined by $X_n = 0$, if no symbols were generated during $[nT, (n+1)T)$, else $X_n = 1, -1$, depending on the symbol that was generated. This process is modeled using an HMM.

Once we decide to restrict our attention to the class of HMMs, we still need to deal with the problems of selection of a model order and of parameters for the chosen model.

- RT-TCPs flow control [14] has a distinctive characteristic: it produces short transmission bursts, followed by short periods of silence. During transmission bursts, RT-TCP fires packets into the network continuously until the number of packets "in flight" reaches the size of the congestion window. During periods of silence, RT-TCP waits until the number of packets in flight is reduced below the size of the current congestion window, while

[2] For example, T could be taken as the amount of time it takes to execute a write on a socket at the transmitter.

at the same time this window size changes (by means of packet acks and/or timeouts). Because of this bursty nature, we propose to use a two-state HMM to model our channel: such models have been proven extremely useful to describe channels that undergo burst noise [12].

- The standard approach to parameter estimation for HMMs consists of using the Baum–Welch algorithm [39]. Issues specific to the application of this algorithm to RT-TCP data are discussed in Section IV.

Preliminary results on our work in progress on these estimation problems will be presented in Section IV. However, an extensive coverage of these topics is intentionally left out of this paper: our goal here is to solve a control problem at the interface between the source at the network, *assuming a known channel*. Channel estimation and identification is a whole subject in itself, and will be dealt with elsewhere.

C. Error-Resilient Video Coding

Next, we consider the problem of efficiently compressing a video signal into any number of packets. We develop a simple coder with the salient property that the quality of the decoded signal depends only on the number of packets available at the receiver, but not on exactly which packets were received. This section builds heavily on work presented in [32], hence, many details are omitted.

1) Temporal Dependencies: The key feature that distinguishes video coding from still image coding is the presence of temporal dependencies among video frames which, if exploited correctly, lead to compression efficiency higher than that attainable by an efficient still image coder applied on each video frame separately. In order to take advantage of these temporal dependencies, the most common techniques are 1) motion compensation, and 2) subband filtering along the time axis as well as in the two spatial directions (known as three–dimensional (3-D) subbands). From a *pure* compression efficiency viewpoint however, there is a general consensus in the research community that 3-D subband coders do not perform as well as coders based on motion compensation.[3]

We claim that in the presence of a dirty channel, that consensus on the superiority of motion compensation techniques over 3-D subband techniques ought to be revised. Motion compensation creates long-range dependencies among portions of the encoded bit streams: as a result, even a small amount of noise in the channel may lead to a significant performance degradation, due to error propagation. On the other hand, no such long-range dependencies exist among 3-D subband coefficients, and hence this representation *might* potentially perform better in noisy environments. Furthermore, 3-D subbands provide a natural framework in which to perform unequal error protection (UEP): different subbands can be protected with different amounts of redundancy. Such UEP schemes become cumbersome when using motion compensation in the video coder. Because of these reasons, we do not feel we are incurring in any obvious suboptimalities when we choose to design a robust coder based on 3-D subbands.[4]

2) Error-Resilient Subband Coding: Our error resilient video coder works as follows. It takes as input a color sequence in YUV format, of size 352 rows $\times$ 240 columns $\times$ 30 frames/s, and blocks it into groups of pictures (GOPs) of 16 frames. Each color band is processed separately. A four-level Haar decomposition is then applied along the temporal and both spatial directions.[5] Each spatio-temporal subband is then encoded using techniques presented in [32], in the context of robust coding of image subbands.

III. Design of the Network/Coder Interface

In this section, we present the most important contribution of this paper: the design of a middleware module that, sitting at the interface between the encoder and the network, sets the parameters of the encoder as a function of channel states. In order to choose these parameters, we need to formulate and solve first a problem of optimal control, and then a problem of optimal allocation of resources.

- The first one is the rate-control problem. Given a sequence of N GOPs to transmit, we have to decide how many fixed-length packets to allocate to each GOP. However, there are conflicting goals that need to be met. On one hand, we would like each GOP to get as large a number of packets as possible, because this leads to high-quality signals decoded at the receiver. On the other hand, we would like each GOP to get as small a number of packets as possible, because then the transmission of each GOP will not take a long time, thus preventing future GOPs from arriving in time. These two conflicting goals need to be balanced somehow.
- Once we decide how many packets to allocate to a GOP, and based on a current estimate of the statistics of packet losses, we need to choose parameters of the encoder to be applied on each subband, leading to the best possible quality of the signal reconstructed at the receiver under the given constraints [32].

Control-theoretic concepts have been used before in the context of communication networks. Keshav presents a control-theoretic treatment of the problem of flow control [15]. Li and Nahrstedt apply concepts from classical control theory in the design of a middleware platform for QoS adaptations in a distributed environment [18]. Specifically on video communication problems, Ronda-Prieto considers in his thesis problems of modeling and control of video coders [25]. However, in that thesis it is assumed that the channel is clean (no packet losses) and deterministic (exact knowledge of how long any transmis-

[3] We want to emphasize that this observation relates *only* to pure compression efficiency. Three-dimensional subband coders are able to provide other functionalities that are much more difficult to provide using a coder based on motion compensation (e.g., scalability in bit rate, image size, and frame rate). In the design of a *complete system*, these are most important issues to take into account.

[4] We should point out that although this paragraph reflects our own ideas, the potential benefits of using 3-D subbands to send video over error-prone channels appear to have been first recognized in prior work by Srinivasan and Chellappa [35].

[5] However simple the Haar filters may appear to be, in our experiments we found them to be the ones providing *by far* the best visual quality. Other filters we tried (e.g., the Daubechies 10-18 wavelet), led to PSNR improvements of about 0.3–0.4 dB over the Haar filters. But by virtue of their length, they induce ringing artifacts. And while these may not be objectionable with still images, in video signals they are very visible (in the form of a background flicker effect), and extremely annoying.

sion will take). Under those assumptions, all that needs to be controlled is the occupancy levels at the transmitter's output buffer, and the only uncertainty is about the future buffer requirements of a signal that is not assumed known in advance. Our problem is completely different however, since we have a channel that does drop packets, and whose transmission times are also random.

A. *Dynamical System Model*

Let $\mathcal{H}$ denote an HMM for the channel, with typical state and sequence $(s_k)_{k\geq 0}$, and observation sequence $(b_k)_{k\geq 0}$ (intuitively, the b_k's are the sequence of 0's, 1's, and -1's to which we made reference in our informal description of Section II-B). Let T_d denote the number of discrete-time units required to display a GOP.[6] Consider also two functions, τ and λ, defined by

$$\tau(u, \mathcal{H}) = \min\left\{m\colon \sum_{k=0}^{m} |b_k| = u\right\}$$

$$\lambda(u, \mathcal{H}) = \sum_{k=0}^{\tau(u, \mathcal{H})} 1_{\{b_k=-1\}}.$$

τ counts the number of clock ticks it takes to transmit u packets over the channel $\mathcal{H}$, and λ counts the number of packets that are lost when sending u packets over the channel $\mathcal{H}$.[7]

As a function of this HMM, we construct a fully-observed dynamical system described by the following equations:

$$t_{n+1} = t_n + \tau(u_n, \mathcal{H}); \tag{1}$$

$$r_{n+1} = r_n + T_d; \tag{2}$$

$$\pi_{n+1} = \Pr(s_{t_{n+1}} | \pi_n, b_{t_n}, \cdots, b_{t_{n+1}}) \tag{3}$$

$$\ell_n = \Pr(\lambda(u_n, \mathcal{H}) | u_n, \pi_n) \tag{4}$$

with initial conditions $r_0 = t_0 = 0$, and $\pi_0 \sim$ the initial distribution of the HMM. These equations should be interpreted as follows. $(u_n)_{n\geq 0}$ is a sequence of control actions, which represent the number of packets allocated to the nth GOP. t_n is the total number of clock ticks from the beginning of transmission until the transmission of the nth GOP starts, and r_n is the number of clock ticks after which the display of the $n-1$th GOP starts. π_n is a probability measure on the hidden state, conditioned on the observations available until time n.[8] And ℓ_n is a probability measure on the number of packets that will be lost during the transmission of the nth GOP.[9]

B. *Rate Control*

1) *Problem Formulation:* Formally, the problem of rate control is formulated as one of optimal stochastic control of the dynamical system defined by (1)–(3), and for that purpose, we need to specify criteria for the selection of a control sequence $(u_n)_{n\geq 0}$. Our goal is to maintain the receiver clock r_n slightly ahead of the transmitter clock t_n, by a fixed amount $d^* > 0$ (on average). For the system in state $[t, r, \pi]$, and for a control action u, we define the cost

$$c([t, r, \pi], u) = (r + T_d - t - \mathrm{E}_\pi\{\tau(u, \mathcal{H})\} - d^*)^2.$$

$r - t$ is the difference between clocks before the application of control; after control, we expect that difference to become $r + T_d - (t + \mathrm{E}_\pi\{\tau(u, \mathcal{H})\})$. And we penalize deviations of this difference from a fixed value d^*, using a quadratic cost.

Finally, a Markov policy $\mathbf{g}_n\colon [t_n, r_n, \pi_n] \to u_n$ is obtained as a solution of

$$\min_{\mathbf{g}} \mathrm{E}^{\mathbf{g}}\left(\frac{1}{N}\sum_{n=0}^{N} c([t_n, r_n, \pi_n], u_n)\right).$$

Why is this a meaningful formulation of our "real-life" problem? To understand this, we study next properties of optimal policies.

2) *Properties of an Optimal Controller:* First of all, note that if r_n increases steadily faster than t_n, then unbounded large buffers are required at the receiver to hold data until its scheduled decoding time. In that case, one could also attain better quality of the signal reconstructed at the receiver by increasing the number of packets allocated to each GOP. Conversely, if r_n increases slowly relative to t_n, eventually the receiver will starve. Hence, we see that the mean value $\mu^{\mathbf{g}} = \mathrm{E}^{\mathbf{g}}(r_n - t_n)$ of the (closed loop) clock differences should remain constant.

Let us fix a constant value $d^* > 0$ by which r_n should remain ahead of t_n on average, and consider the "meaning" of the cost for N large. Since the HMM is stationary and $\mathbf{g}$ is Markovian, the differences $d_n = r_n - t_n$ are stationary too (for large n), and hence $\mathtt{Var}(d_n)$ is independent of n (for large n). By the Strong Law of Large Numbers we have that

$$\begin{aligned}
&\lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N} c([t_n, r_n, \pi_n], u_n) \\
&\quad = \lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N} [d_n - d^*]^2 + [T_d - \mathrm{E}_\pi\{\tau(u_n, \mathcal{H})\}]^2 \\
&\qquad + 2[d_n - d^*][T_d - \mathrm{E}_\pi\{\tau(u_n, \mathcal{H})\}] \\
&\quad = \mathtt{Var}(d_n) + (d^* - \mu^{\mathbf{g}})^2 \\
&\qquad + (T_d - \mathrm{E}_\pi\{\tau(u_n, \mathcal{H})\})\,[\ldots]\ \text{(a.e.)}.
\end{aligned}$$

There are two properties of an optimal $\mathbf{g}$ that are of interest to us. First, it is clear that an optimal controller will make $\mu^{\mathbf{g}} = d^*$—else the cost is increased by $(d^* - \mu^{\mathbf{g}})^2$–, thus making the parameter d^* be what we intuitively want it to be: the average amount by which r_n stays ahead of t_n. Second, if in steady state we have $\mathrm{E}_\pi\{\tau(u_n, \mathcal{H})\} \neq T_d$, then clearly $\mu^{\mathbf{g}}$ cannot be a constant, and hence the first condition cannot be satisfied. So we see that for an optimal $\mathbf{g}$, the last two terms in the limit drop out. Furthermore, since for N large the cost becomes $\mathtt{Var}(d_n)$, asymptotically the optimal policy yields a *minimum variance* controller. This is desirable because it allows us to pick small values of d^*: large values are undesirable because they result in

[6] $T_d = (1/30)G_s/T$, where G_s is the number of frames in a GOP, and T is the time-quantization step.

[7] $1_{\{P\}}$ is an indicator function: 1 when the condition P is true, 0 otherwise.

[8] π_n is an *information state* [16, Ch. 6.5]: it captures everything that is known about the unobserved state.

[9] Note that in setting up our system model we made the assumption that ack messages are not lost, and therefore that missing acks indicate lost packets. This assumption is consistent with those of [14].

large buffers at the receiver; however, if d^* is "too close" to zero (say, less than a standard deviation), then the probability of the receiver starving is high.

3) Design of a Controller: Our problem is one of optimal control of a stochastic dynamical system. If our observed system were indeed a Markov chain, a natural controller to consider would be the dynamic programming (DP) controller, which is constrained by the DP equations

$$V_n([t, r, \pi]) = \min_{u \in \{0, 1, \cdots\}} c([t, r, \pi], u) + \mathrm{E}\{V_{n+1}(h([t, r, \pi], u))\}$$

(for $0 \leq n < N$), terminal cost $V_N([t, r, \pi]) = 0$, and where $h([t, r, \pi], u)$ denotes the state transition function defined by equations (1)–(3). The value of an optimal control u_n is then defined as the u attaining the minimum in the cost-to-go V_n [16].

We have two problems here though. First, the system defined by equations (1)–(3) is *not* a Markov chain, since the definition of τ involves sums of observations of an HMM, i.e., of dependent variables.[10] But assume that we were willing to consider the DP controller as a reasonable low-complexity approximation to the optimal controller. Still, even though it may be conceptually straightforward to implement the DP controller (just write a computer program to solve the DP equations), in practice this task may prove to be rather involved. Instead, in this work we design a controller by *enforcing* the optimality condition that $\mathrm{E}_\pi\{\tau(u, \mathcal{H})\} = T_d$: we define a stationary controller by setting $\mathbf{g}[t, r, \pi] = u$, where u is chosen such that $\mathrm{E}_\pi\{\tau(u, \mathcal{H})\}$ is as close as possible to T_d.[11] Our controller is defined as follows.

1) Define a control action

$$u' = \arg\max\{u: \mathrm{E}_\pi\{\tau(u, \mathcal{H})\} \leq T_d\}.$$

2) For a random variable S uniformly distributed over $[\mathrm{E}_\pi\{\tau(u', \mathcal{H})\}, \mathrm{E}_\pi\{\tau(u'+1, \mathcal{H})\}]$, define

$$u^* = \texttt{if } (S \geq T_d) \texttt{ then } u' \texttt{ else } u' + 1.$$

Note that by definition of u', $\mathrm{E}_\pi\{\tau(u'+1, \mathcal{H})\} > T_d$. By time-sharing between u' and $u'+1$ we make sure that, on average, $\mathrm{E}_\pi\{\tau(u^*, \mathcal{H})\} = T_d$, which is what we want.

3) The control action applied is $\alpha \cdot u^*$, where $\alpha > 0$ is a real number chosen as a function of the state, as illustrated in Fig. 6.

C. Redundancy Allocation

Space constraints prevent us from presenting a detailed description of how this problem is formulated and solved: here we only mention that this problem is dealt with as one of constrained allocation of resources in the sense of [34]. For full details on this topic, as well as on implementation details of the proposed controller, the reader is referred to [30, Ch. 4].

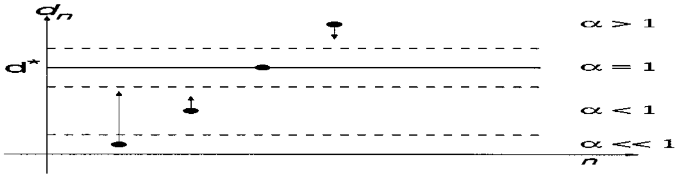

Fig. 6. Definition of the weights α in our proposed controller. The intuition is that when the applied control action u is smaller than u^*, on average d_n will increase, whereas when $u > u^*$ d_n will decrease. Hence, our goal is to give "soft pushes" to d_n to bring it close to d^*; and if d_n gets "too close" to zero (the starvation region), then we want to "push harder."

IV. Experimental Results

In this section, we present experimental results testing different aspects of our proposed system.

A. Experimental Setup

The experimental setup is as follows. We installed a transmitter at `lima.cs.uiuc.edu` (Urbana, IL), and we installed receivers at 1) `dukat.ifp.uiuc.edu` in the Beckman Institute at UIUC, to obtain data for a high speed local area network; 2) `grip.cis.upenn.edu` in Philadelphia, PA, and `octavian.ics.uci.edu` in Irvine, CA, to obtain data for connections within the continental United States across different time zones (1 and 2 h difference); and `lcavsun10.epfl.ch` in Lausanne, Switzerland, to obtain data for an intercontinental connection. Our transmitter picks a receiver at random, transmits data for 5 min, and for each transmitted packet it records both its time of departure as well as, if received, the time of arrival of its acknowledgment. Furthermore, for each trace we record transmission parameters of the connection: size of socket buffers, times of transmission start and end, IP addresses of the transmitter and receiver hosts.

B. Dependence of Control on Channel Parameters

In a first experiment, we study properties of our controller for different transmission scenarios: in Fig. 7 we plot the evolution of the clock differences d_n and control actions u_n, for transmission traces measured at different times of the day.

Note a few things.

- During the night (lower network loads), the controller injects more bandwidth than during the day, and the fluctuations of the differences d_n around the sought value d^* are smaller.
- While transmitting GOPs 80–100 during the day, d_n got "too close" to the starvation region: the controller responded by drastically reducing the injected bandwidth, pushing d_n up.
- Except when reacting to disturbances, in steady-state the controller fluctuates smoothly around the available bandwidth: for this specific connection, ≈6.43 Mbits/s during the day, ≈8.85 Mbits/s during the night.
- A buffer of size 2 GOPs is enough in this case to absorb all clock variations.

[10] In [31] we incorrectly stated that this system has the Markov property.

[11] Note that if we are willing to consider approximations to the optimal controller, this is an intuitively natural choice: we want to inject a number of packets that we believe will take about T_d clock ticks to be transmitted.

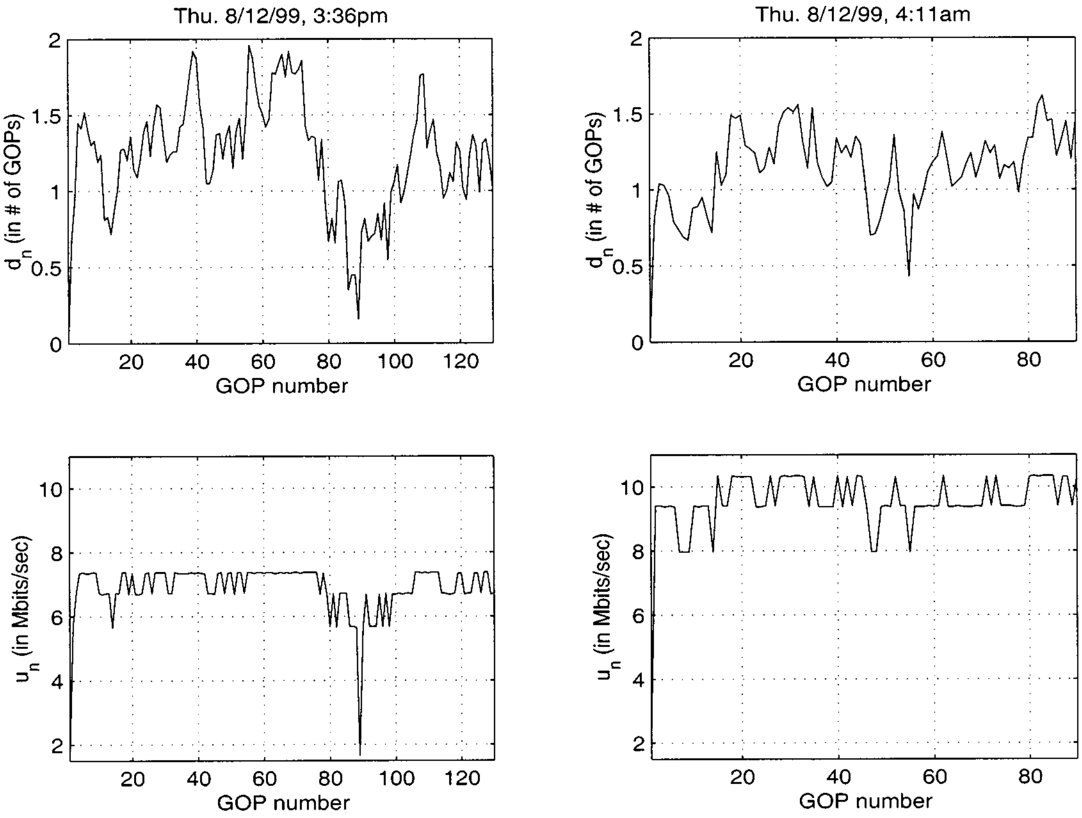

Fig. 7. Evolution of d_n as a function of the control signals. Top plots: d_n; bottom plots: u_n; left plots: day transmission; right plots: night transmission. In this case, $d^* = 1$ and a GOP of size 32 frames was used (so these plots correspond to 1.5–2 min long transmissions).

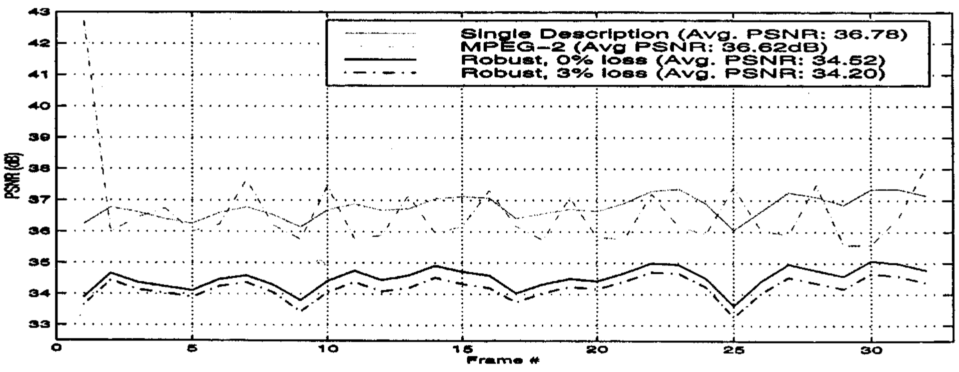

Fig. 8. Frame-by-frame PSNR for 32 frames (1 GOP) of the "football" sequence. References: a non error-resilient version of the coder described in Section II, obtained by replacing the multiple-description quantizer by a regular uniform scalar quantizer, and MPEG-2.

C. Quality of Service

In another experiment, we consider the quality of the signals delivered to the receiver: in Fig. 8 we plot the peak signal-to-noise ratio (PSNR) obtained in a sample transmission, and in Fig. 9 we show an original frame and a sample reconstruction.

Note that there is a gap of about 2 dB between the performance of the single description coder and the performance of the error-resilient coder: this is the price we pay for eliminating retransmissions. We want to point out however three important facts.

- This 2 dB is a worst-case example. We have tried different video sequences at different bit rates, and 2 dB was the largest drop in PSNR we found.
- If PSNR measurements are questionable but still acceptable for still images, for video sequences they are virtually uncorrelated with perceived image quality: we report PSNR numbers because we have no other means of presenting objective quality measurements. Especially at these high bit rates, the reconstruction computed by the single description coder over an ARQ channel and the reconstruction computed by the error-resilient coder with a 3% packet loss rate are visually indistinguishable.

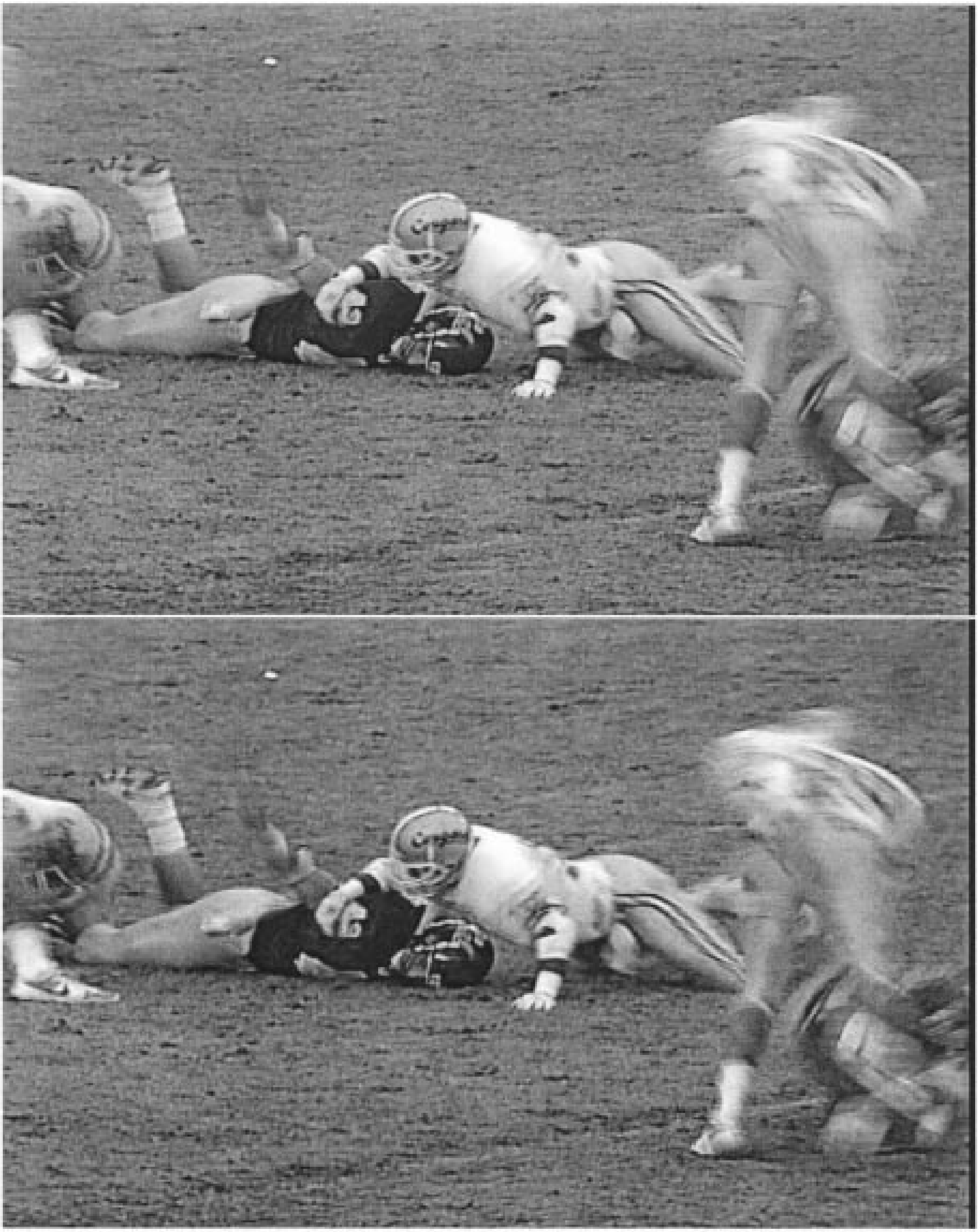

Fig. 9. Quality of the video signal reconstructed at the receiver. Top: original frame #5; bottom: reconstruction with ten packets lost out of 300 transmitted (packet size: 4 Kbytes, injected bandwidth: 9.38 Mbits/s, loss rate: 3%).

- The video coder used as a reference in Fig. 8 (together with MPEG-2) was chosen because it is the "closest" non error-resilient coder to the one we used. This is important to ensure that the PSNR drop shown is not due to differences in the coders, but instead due to elimination of retransmissions. And its performance is seen to be comparable to the standard MPEG-2.

D. Accuracy of the Channel Model

1) Model Order: In a final experiment, we use one of the traces to estimate parameters for two different models: a two-state chain, and a five -state birth-and-death chain. We also compute a new two-state chain, obtained by collapsing states of the five -state chain. The resulting parameters are shown in Fig. 10.

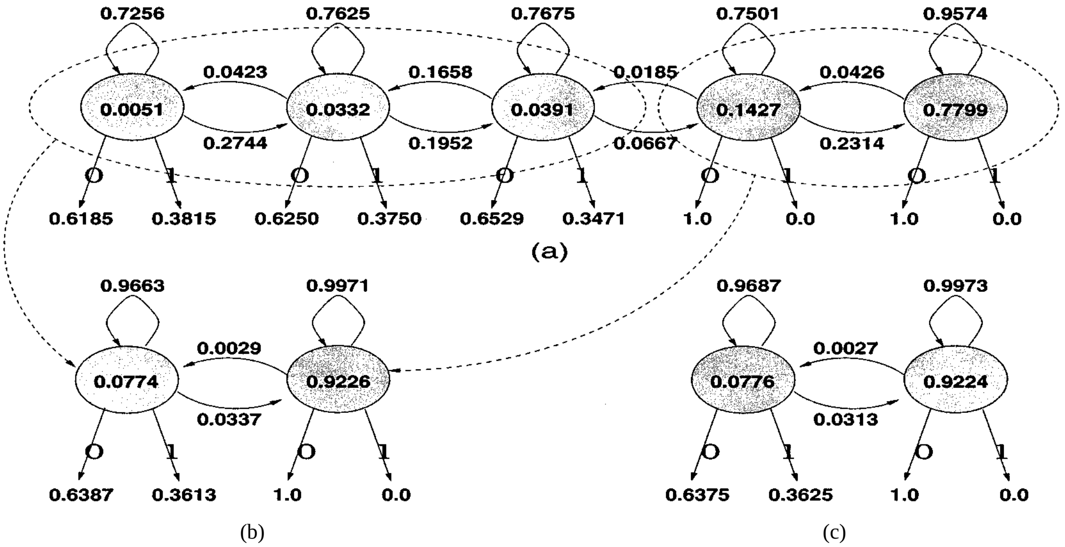

Fig. 10. Parameters for two different HMMs, estimated from the same transmission trace (`lima.cs.uiuc.edu`→`grip.cis.upenn.edu`, Thursday Aug. 12, 1999, 4:11am CST): (a) a five -state birth and death chain, (b) a "synthetic" two-state chain, obtained by collapsing states in the five-state chain of (a), and (c) a two-state chain fitted directly to the trace. The numbers inside states denote stationary probabilities.

The usefulness of our proposed *two-state* HMM is verified by the fact that the distribution of observations induced by both chains is virtually identical.[12] The parameters obtained when fitting a two-state chain are almost identical to those obtained by first fitting a five-state chain and then reducing this one to two states: if more than two states were indeed necessary, this state-reduction process applied to the five-state chain would result in a loss of information, which apparently is not occurring in this case. This is intuitively very pleasing. We argued earlier that our main motivation for considering a two-state HMM was the observation that RT-TCP produces bursts of "silence," followed by bursts of "activity": the two-states discovered in the data by the Baum–Welch algorithm correspond exactly to the silence and activity states, as follows from the probabilities of observations in each state.

2) Initial Conditions for the Baum–Welch Algorithm: A most important issue to deal with when estimating parameters of an HMM using the Baum–Welch algorithm is the choice of initial condition. This algorithm is an iterative procedure which, given a realization of the HMM to be identified, produces a sequence of estimates for the parameters of this HMM such that the probability of the observed sequence is nondecreasing. However, this method is not guaranteed to converge to the global maximum likelihood estimate, and therefore the choice of a good initial condition is critical to the quality of the parameters obtained. Our approach to handling this matter can be described in simple terms by saying that we "manufacture" an initial model in which the average injection and loss rates coincide with that of the measured trace.

Note that an expression for the average bandwidth B_{inj} of a connection, as well as for the average packet loss rate B_{loss} is given by

$$B_{\text{inj}} = \pi_A \cdot (\pi(1|A) + \pi(-1|A)) + \pi_S \cdot (\pi(1|S) + \pi(-1|S))$$

$$B_{\text{loss}} = \frac{\pi_A \cdot \pi(-1|A) + \pi_S \cdot \pi(-1|S)}{\pi_A \cdot (\pi(1|A) + \pi(-1|A)) + \pi_S \cdot (\pi(1|S) + \pi(-1|S))}$$

where A and S are the two states in the chain (for active and silent), and both quantities are expressed in units of packets per time-quantum. To choose appropriate initial conditions, we take the following steps.

1) We choose an arbitrary model: uniform initial probabilities, symmetric transition probabilities with $\pi(A \rightarrow A) = 0.99$, observation probabilities $\pi(0|A) = 0.5$, $\pi(0|S) = 0.9$, $\pi(1|\cdot) = \pi(-1|\cdot)$.
2) We apply the Baum–Welch algorithm to the measured trace, using this arbitrary initial condition.
3) We estimate B_{inj} and B_{loss} from the measured trace.
4) We modify the *observation* probabilities obtained in (2), so that B_{inj} and B_{loss} as computed from the model is equal to the estimate computed in (3).
5) We run the Baum–Welch algorithm again, with the initial condition of (4).

Note that we could continue iterating steps 4) and 5), until a fixed point of the parameter estimates is reached. At this time we have not tried to prove convergence of this procedure to one such fixed point, although we have found empirically that in many cases it does, and furthermore, that it does after a couple of iterations only.

The intuition behind our initialization procedure is that an HMM whose injection and loss rates agree with those of the measured traces should not be "too far away" from the real one, and hence we hope that the local maximum of the likelihood function found by the Baum–Welch algorithm will indeed correspond to the actual channel parameters.

3) Limitations: Our proposed physical interpretation of the states of a chain (in terms of active/silent periods) not only

[12] In Fig. 10 a model with only two observations was used: 0 or 1, counting the number of packets injected, but disregarding acknowledgment. This was done so as to make it easier to read the figure, which is already fairly cluttered. Although not reported here, the same experiment was done with the model based on three observations as defined in Section II, with identical results.

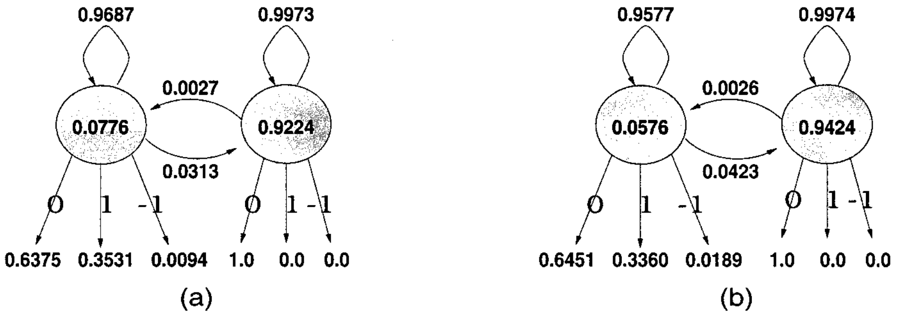

Fig. 11. Parameters for a two-state HMM for the same connection (`lima.cs.uiuc.edu`→`grip.cis.upenn.edu`) and route (as determined by the `traceroute` program), at different times: (a) Thursday Aug. 12, 1999, 04:11am CST, and (b) Thursday Aug. 12, 1999, 3:36pm CST. For packets with a payload of size 32 Kbits, and time steps of length $0.000\,01$ s, for (a) $B_{\text{inj}} \approx 8.85$ Mbits/s and $B_{\text{loss}} \approx 2.6\%$, whereas for (b) $B_{\text{inj}} \approx 6.43$ Mbits/s and $B_{\text{loss}} \approx 5.33\%$.

agrees well with intuition, but it also allows us to discover an inherent limitation of the proposed two-state model. Since the model is stationary, if it were entirely accurate, B_{inj} and B_{loss} should remain constant over time. However, as one would intuitively expect, this is not the case: available bandwidths and packet loss rates depend on machine and network loads (which change over time), as illustrated in Fig. 11.

Such nonstationarities are not captured by our two-state stationary model, which is seen to provide a good approximation in a local sense *only*: the parameters of the model may need to be re-estimated over time. All these modeling and estimation issues are being currently investigated.

V. Conclusions

The most important conclusion drawn from these results is that certain high-speed segments of the public Internet do provide an infrastructure that supports the transport of video signals of quality significantly higher than current systems do, even without any form of network QoS guarantees. We proposed a system whose most salient feature is a network/coder interface which plays a role analogous to that of a Leaky Bucket: it specifies traffic-shape constraints which ensure that, if the source complies with the traffic specification, then the network will most likely be able to deliver all the injected data in time to ensure a high-quality smooth video playback, while at the same time remaining fair to other video or data connections. This interface, in combination with minor modifications to existing transport protocols, and a video coder that can reconstruct high-quality video signals in the presence of moderate packet losses, result in a complete system capable of delivering video signals at the quality of current TV-broadcasts.

Note that our proposed mechanism to support QoS does not require modifications to current network infrastructure: our controller derives all its information just from looking at received packet acknowledgment.[13] This is in contrast with proposals to support QoS based on reservations (such as the RSVP protocol [42]), or based on obtaining fair bandwidth allocations (such as per-flow queueing mechanisms [7], or per-flow dropping mechanisms [19]), which specifically require maintaining per-flow state information at different points in the network. Note also that if the network did provide QoS support then our controller could still benefit, by means of having access to extra information on which to condition when computing the recursive updates of (3).

The importance of having been able to attain such high-quality reconstructions using a system that explicitly avoids retransmitting lost data cannot be overemphasized. Channel coding by retransmission of erased data is known to achieve the theoretical capacity of the binary erasure channel [6], and video coding for clean channels is a problem that has already received a fair amount of attention from the research community. Therefore, if we *pretend* that the separation theorem holds for packet networks, and disregarding delay constraints, the only theoretical suboptimality involved in separation-based approaches is due to video coders not attaining the rate/distortion bound of the source; in practical terms, there is little hope for improvement within this framework. But our proposed system can deliver signals of quality comparable to that of these "optimal" systems, while at the same time satisfying tight real-time delay constraints.

Acknowledgment

The authors would like to thank W. Turin for educating them on the usefulness of HMMs as a tool to approximate complex stochastic phenomena and V. Bharghavan for pointing out an error in an early version of this work, and for bringing to their attention reference [29]. They would also like to thank M. Vetterli, J. Smith and N. Venkatasubramanian, for access to machines at EPFL, University of Pennsylvania, and the University of California, Irvine, respectively, that they used to collect transmission traces and D. Deatrich, for thorough information on EPFLs network. They also thank the anonymous reviewers, for useful feedback that helped them improve the quality of this manuscript.

References

[1] J. Boyce, "Packet loss resilient transmission of MPEG video over the Internet," *Signal Process.: Image Commun.*, vol. 15, no. 1–2, pp. 7–24, 1999.

[2] A. R. Calderbank, "The art of signaling: Fifty years of coding theory," *IEEE Trans. Inform. Theory*, vol. 44, no. 6, pp. 2561–2595, 1998.

[3] Z. Chen, "Coding and transmission of digital video on the Internet," Ph.D. dissertation, Univ. Illinois, Urbana-Champaign, 1997.

[13] There are advantages to this (the most obvious one being backward compatibility), and there are disadvantages too, such as vulnerability to ill-behaved flows, which is the motivation for some of the work we mention below. This statement is not meant as a claim of superiority of one approach over another, only as a remark of a feature of our proposed system.

[4] Z. Chen, S.-M. Tan, R. Campbell, and Y. Li, "Real time video and audio in the world wide web," in *Proc. 4th World Wide Web Conf.*, 1995, http://www.vosaic.com/.

[5] D. D. Clark and D. L. Tennenhouse, "Architectural considerations for a new generation of protocols," in *Proc. ACM SIGCOMM*, 1990.

[6] T. M. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[7] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. ACM SIGCOMM*, 1989.

[8] L. Dittman, S. Jacobsen, and K. Moth, "Flow enforcement algorithms for ATM networks," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 343–350, 1991.

[9] D. Dwyer, S. Ha, J.-R. Li, and V. Bharghavan, "An adaptive transport protocol for multimedia communication," in *Proc. Int. Conf. Multimedia Computing and Systems*, Austin, TX, 1998.

[10] M. W. Garrett, "Contributions toward real-time services on packet switched networks," Ph.D. dissertation, Columbia Univ, New York, 1993.

[11] M. W. Garrett and M. Vetterli, "Joint source/channel coding of statistically multiplexed real-time services over packet networks," *IEEE/ACM Trans. Networking*, vol. 1, no. 1, pp. 71–80, 1993.

[12] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, 1960.

[13] C. Hsu, A. Ortega, and A. Reibman, "Joint selection of source and channel rate for VBR video transmission under ATM policing constraints," *IEEE J. Select. Areas Commun.*, vol. 16, no. 6, pp. 1016–1028, 1997.

[14] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM*, 1988.

[15] S. Keshav, "A control-theoretic approach to flow control," in *Proc. ACM SIGCOMM*, 1991.

[16] P. R. Kumar and P. Varaiya, *Stochastic Systems: Estimation, Identification and Adaptive Control*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[17] K.-W. Lee, R. Puri, T.-E. Kim, K. Ramchandran, and V. Bharghavan, "An integrated source coding and congestion control framework for video streaming in the Internet," in *Proc. IEEE INFOCOM*, 2000.

[18] B. Li and K. Nahrstedt, "A control-based middleware framework for quality of service adaptations," *IEEE J. Select. Areas Commun.*, vol. 17, no. 9, pp. 1632–1650, 1999.

[19] D. Lin and R. Morris, "Dynamics of random early detection," in *Proc. ACM SIGCOMM*, 1997.

[20] S. R. McCanne, "Scalable compression and transmission of Internet multicast video," Ph.D. dissertation, Univ. California, Berkeley, 1996.

[21] A. E. Mohr, E. A. Riskin, and R. E. Ladner, "Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 819–828, 2000.

[22] A. Ortega, "Optimization techniques for adaptive quantization of images and video under delay constraints," Ph.D. dissertation, Columbia Univ., New York, 1994.

[23] M. Parris, K. Jeffay, and F. D. Smith, "Lightweight active router-queue management for multimedia networking," in *Proc. SPIE Multimedia Computing and Networking*, 1999.

[24] C. Partridge, *Gigabit Networking*. Reading, MA: Addison-Wesley, 1994.

[25] J. I. Ronda Prieto, "Modelado Estadístico y control de codificadores de video," Ph.D. dissertation, Univ. Politécnica de Madrid, Spain, 1994.

[26] E. Rathgeb, "Modeling and performance comparison of policing mechanisms for ATM networks," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 325–334, 1991.

[27] Real Networks, Inc., "Real Video Player,", http://www.real.com/.

[28] A. Reibman and B. Haskell, "Constraints on variable bit rate video for ATM networks," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 2, no. 4, pp. 361–372, 1992.

[29] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," in *Proc. IEEE INFOCOM*, 1999.

[30] S. D. Servetto, "Compression and reliable transmission of digital image and video signals," Ph.D. dissertation, Univ. Illinois, Urbana-Champaign, 1999.

[31] S. D. Servetto and K. Nahrstedt, "Video streaming over the public Internet: Multiple description codes and adaptive transport protocols," in *Proc. IEEE Int. Conf. Image Proc. (ICIP)*, Kobe, Japan, 1999.

[32] S. D. Servetto, K. Ramchandran, V. A. Vaishampayan, and K. Nahrstedt, "Multiple description wavelet based image coding," in *IEEE Trans. Image Processing*, vol. 9, May 2000, pp. 813–826.

[33] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423 623–656, 1948.

[34] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1445–1453, Sept. 1988.

[35] M. Srinivasan and R. Chellappa, "Adaptive source/channel subband video coding for wireless channels," *IEEE J. Select. Areas Commun.*, vol. 16, no. 9, pp. 1830–1839, 1998.

[36] W. Stevens, *TCP/IP Illustrated*. Reading, MA: Addison-Wesley, 1994, vol. 1, The Protocols.

[37] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: A scalable architecture to approximate fair bandwidth allocations in high speed networks," in *Proc. ACM SIGCOMM*, 1998.

[38] W. Tan and A. Zakhor, "Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Trans. Multimedia*, vol. 1, pp. 172–186, June 1999.

[39] W. Turin, *Digital Transmission Systems: Performance Analysis and Modeling*. New York: McGraw-Hill, 1998.

[40] W. Turin and R. van Nobelen, "Hidden Markov modeling of flat fading channels," *IEEE J. Select. Areas Commun.*, vol. 16, no. 9, pp. 1809–1817, 1998.

[41] N. Venkitaraman, T.-E. Kim, K.-W. Lee, S. Lu, and V. Bharghavan, "Design and evaluation of a suite of congestion control algorithms for the future Internet," in *Proc. ACM SIGMETRICS*, 1999.

[42] L. Zhang, B. Braden, D. Estrin, S. Herzog, and S. Jamin, "RSVP: A new resource reservation protocol," *IEEE Network*, 1993.

Sergio D. Servetto (S'94–M'99) was born in Argentina in 1968. He received the Licenciatura en Informatica degree from the Universidad Nacional de La Plata (UNLP), Argentina, in 1992, and the M.Sc. degree in electrical engineering and the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign (UIUC), in 1996 and 1999, respectively.

From 1994 to 1999, he was a Graduate Research Assistant at the Image Formation and Processing Group, Beckman Institute, UIUC, and at the Multimedia Operating Systems and Networking Group, Department of Computer Science, UIUC. He held part-time teaching appointments during 1992–1993 at UNLP and during 1996–1997 at UIUC. He has worked for IBM (Buenos Aires, Argentina), Bell Labs (Murray Hill, NJ), and for AT&T Labs (Florham Park, NJ). Since October 1999, he is the First Assistant to Professor Martin Vetterli, Laboratoire de Communications Audiovisuelles, Ecole Polytechnique Federale de Lausanne, Switzerland. His research interests are in the general areas of networks, information theory, and signal processing applications.

Dr. Servetto was the recipient of the 1998 Ray Ozzie Fellowship, given to "outstanding graduate students in Computer Science," and of the 1999 David J. Kuck Outstanding Thesis Award, for the best doctoral dissertation of the year, both from the Department of Computer Science, UIUC. He is currently writing a book, tentatively entitled *Real Time Services over Best-Effort IP Networks*, to be published by Kluwer in 2001.

Klara Nahrstedt (S'93–M'95) received the B.A. in mathematics and the M.Sc. degree in numerical analysis from from Humboldt University, Berlin, Germany, in 1984 and 1985, respectively. In 1995, she received the Ph.D. degree from the Department of Computer and Information Science, University of Pennsylvania, Philadelphia.

She was a Research Scientist in the Institute for Informatik in Berlin from 1985 to 1990. Currently, she is an Assistant Professor, Computer Science Department, University of Illinois at Urbana-Champaign. Her research interests are directed toward reconfigurable multimedia services, multimedia protocols, multimedia security, middleware systems, quality of service (QoS) provision, QoS routing, and QoS-aware resource management in distributed multimedia systems. She is the coauthor of the widely used multimedia book *Multimedia: Computing, Communications and Applications* (Englewood Cliffs, NH: Prentice-Hall).

Dr. Nahrstedt was the recipient of the Early NSF Career Award, the Junior Xerox Award, Lucent Award, and the IEEE Communication Society Leonard Abraham Award for Research Achievements.