Analysis of Rate-Distortion Functions and Congestion Control in Scalable Internet Video Streaming

Min Dai
Department of Electrical Engineering
Texas A&M University
College Station, TX 77843
min@ee.tamu.edu

Dmitri Loguinov
Department of Computer Science
Texas A&M University
College Station, TX 77843
dmitri@cs.tamu.edu

ABSTRACT

Internet streaming applications usually have strict requirements on bandwidth, delay, and packet loss, while the current best-effort Internet does not provide any Quality-of-Service (QoS) guarantees to end flows. To achieve a higher level of QoS for the end user, Fine-granular Scalability (FGS), which has both strong error-resilience and flexibility during streaming over variable-bandwidth channels, has been accepted as a standard coding scheme for the video streaming profile in MPEG-4 [19]. Note that FGS and its extensions (e.g., progressive FGS) can also be used in the emerging video coding standards such as H.26L. This paper investigates rate-distortion (R-D) models of FGS coders and shows how such models can be used in a simple rate control framework for FGS streaming over the Internet. In terms of congestion control, we examine advantages and drawbacks of Kelly's proportional-fairness framework [12] and investigate its practical use both in the best-effort and AQM-enabled Internet. Our simulation results show that the developed R-D models provide fundamental insight into the structure of FGS coders and that constant-quality streaming is possible as long as the number of end flows competing at each bottleneck resource remains fairly stationary.

Categories and Subject Descriptors

I.4 [**Image Processing and Computer Vision**]: Compression (Coding); C.2.1 [**Network Protocols**]: Network Communications

General Terms

Algorithms, Performance, Experimentation

Keywords

MPEG-4 FGS, Scalable Coding, Video Streaming, Congestion Control, R-D Modeling

NOSSDAV'03, June 1–3, 2003, Monterey, California, USA.

1. INTRODUCTION

Video streaming is becoming an increasingly important Internet application. In order to be successful in the best-effort environment, video streaming must possess both congestion control and adaptive video scaling mechanisms. This paper investigates several fundamental properties of scalable video (MPEG-4 FGS) and examines the suitability of recently proposed congestion controls [10], [12], [13] for achieving constant-quality streaming.

Congestion control has been actively studied in the area of networking and video-coding, and two general approaches emerged [26]: the network-centric approach and the end-system approach. The *network-centric* approach requires routers/switches to guarantee end-to-end bandwidth and to prevent large delays and packet loss in the network. The *end-system* approach employs control techniques in the video coding scheme to maximize the video quality without QoS support from the network [3], [8], [15]. Since it is expected that no QoS support will be available in the Internet in the near future, this paper studies video streaming using the *end-system* approach and relies on support from smooth end-to-end congestion control to adjust the sending rate of the server.

Recall that FGS [14], [20] has been chosen as the streaming profile of the ISO/IEC MPEG-4 standard [19], because it provides a flexible and low-overhead foundation for scaling the enhancement layer to match variable network capacity. FGS consists of two layers: the base layer and a single enhancement layer. The base layer is usually coded at significantly lower bitrates than the enhancement layer. Hence, it is often assumed that the end-to-end path has enough capacity to deliver the base layer to the user, which under certain generic assumptions (such as base-layer FEC and priority retransmission of the base layer) guarantees its error-free delivery to the receiver. As a result, the server only needs to control the amount of transmitted bits in the enhancement layer to fill the remaining capacity of the network channel.

Also recall that due to the inherent nature of rate control in the current video standards (e.g. MPEG-4), the encoder often produces video sequences with highly fluctuating visual quality [27], [28], [29]. Therefore, it is only natural to scale the FGS layer during transmission so as to "flatten out" the fluctuating quality of the base layer.

Many current approaches (e.g., [3], [24]) develop rate control algorithms that assume a *constant-rate* channel and do not couple FGS scaling with congestion control. Further-

more, the existing approaches to constant-quality streaming often rely on empirical R-D models to decide how to scale the FGS layer [27], [28], which provide very little new information about the structure of scalable coders or their R-D tradeoffs. What is missing from this picture are deeper understanding of R-D functions of scalable (enhancement) layers and realistic congestion control assumptions about the network channel. If the existing methods were to use the classical AIMD (or various other TCP-friendly schemes), the fluctuating rate of these controllers would void any attempts of the server to produce a flat PSNR curve. The goal of our work is to bridge this gap by studying R-D characteristics of FGS and investigating asymptotically stable (smooth) controllers for video streaming.

The paper is organized as follows. Section 2 provides the necessary background and motivation. Section 3 develops a novel closed-form R-D model of FGS video that generalizes the R-D models in classical information theory [4], [24] to second-order polynomials. Section 4 shows how this model can be applied in video streaming when the server knows its available bandwidth. Section 5 discusses the benefits and limitations of proportional-fairness congestion control. Section 6 couples our analytical R-D model with feedback congestion control and shows simulation results. Section 7 concludes the paper.

2. MOTIVATION AND RELATED WORK

Recall that a fundamental problem both in video coding and real-time scaling of the enhancement layer is the knowledge of the correct R-D information of the video sequence. There are two means of obtaining R-D curves: analytical and empirical. The *analytical* approach builds a closed-form R-D model of the source and/or encoder based on the statistical properties of the source data and/or coding scheme [3], [8]. The *empirical* approach constructs R-D curves by interpolating between several sampled values of rate and distortion [15], [28]. Unfortunately, the empirical approach does not give us much insight into the video coding process and its high computation requirements typically place an unnecessary burden on streaming servers.

On the other hand, present analytical R-D approaches are mostly developed for non-scalable video (base layer) and thus lack accuracy in FGS streaming applications [3], [24]. It should be further noted that classical rate distortion theory and information theory develop simple closed-form R-D formulas for statistical properties (e.g., memoryless Gaussian sources) not typically found in real sequences.

Recall that in information theory, DCT coefficients of each frame i are modeled as a zero-mean random variable X_i. One popular statistical model for DCT data is that of a Gaussian source with mean $\mu = 0$ and variance σ_x^2, which leads to tractable results (that are upper bounds on achievable quality) in information theory [4]:

$$D(R) = \sigma_x^2 2^{-2R}. \tag{1}$$

In terms of quantization step Δ, the classical model (1) can be summarized as [6], [7]:

$$D(\Delta) = \frac{\Delta^2}{\beta} \tag{2}$$

and

$$R(\Delta) = \frac{1}{2}\log_2\left(\frac{\varepsilon^2 \beta \sigma_x^2}{\Delta^2}\right), \tag{3}$$

where β is 12 for uniformly distributed sources (often used for Gaussian and Laplacian sources [1]) and ε^2 is introduced to model the reduction in efficiency due to quantization (ε^2 = 1.4 for Gaussian, 1.2 for Laplacian, and 1.0 for uniform source data). Several variations of the classical R-D model have been proposed in the past. For example, Hang *et al.* [7] extend the model in (2) by introducing three content-dependent parameters and adjusting the value of β depending on the quantization step and the value of these parameters empirically estimated for each frame.

There is a significant research activity in the area of rate control for FGS streaming [22], [27], [28]. All these approaches are developed under a buffer constraint that the number of bits consumed in a group of pictures (GOP) or a certain (fixed) window should be equal to the size of decoder's buffer. Under this constraint, Zhao *et al.* [28] apply a Newton search method to find the optimal distortion based on the empirical R-D curve; however, the method does not consider variation of the available bandwidth during transmission, which makes it more suitable for offline downloading rather than real-time streaming.

In another work based on empirical R-D modeling, Zhao *et al.* [27] adopt a sliding window approach to perform rate adaptation for each frame in both the enhancement and base layers. This approach not only alters the quantization parameters in the encoder for the base layer, but also adaptively allocates bits for the enhancement layer. However, the high computation complexity of this approach makes it less appealing during real-time streaming where the server must adapt to bandwidth variations very quickly. Finally, Wang *et al.* [24] use the classical R-D model (1)-(3) and Lagrange optimization to allocate bits for the FGS enhancement layer.

3. MODELING FGS VIDEO

3.1 Introduction

To show that R-D curves for scalable coders are not modeled by traditional results in information theory, we coded the Foreman sequence (128 kb/s base layer @ 10 fps) and extracted the actual R-D functions of the FGS layer from frames 117 and 219 (shown in Figure 1). Notice that (1) is a straight line in the PSNR-R domain and that it does not match well the actual shape of the curves in Figure 1. Furthermore, as seen in the figure, even a quadratic function is not capable of modeling both low and high bitrates at the same time.

Recall that video coding coupled with congestion control requires accurate R-D information to decide how to control the encoding and transmission bitrate under strict bandwidth constraints. Devising a good R-D model involves two important components: a) precisely modeling source data; and b) selecting a sound operational model. Therefore, we first investigate statistical properties of source data in FGS enhancement layers and then build a closed-form R-D model of scalable FGS coders.

3.2 R-D Modeling Framework

The enhancement layer input to the FGS encoder is the discrete cosine transform (DCT) residue between the origi-

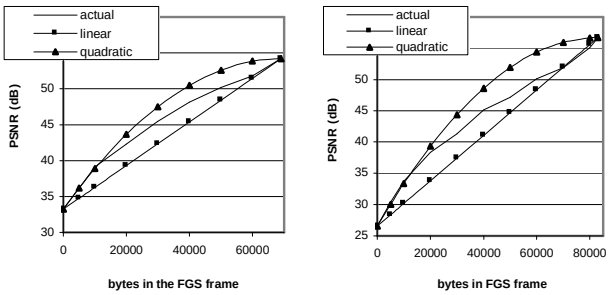

Figure 1: Frame 117 (left) and 219 (right) of Foreman CIF.

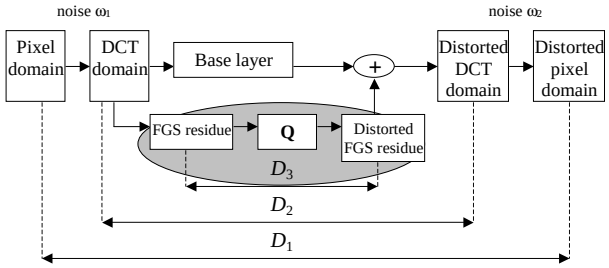

Figure 2: Different levels of distortion and a basic model of FGS.

nal image and the reconstructed image in the base layer [19]. There are three stages during FGS coding, which causes three levels of distortion. Figure 2 provides a conceptual view of the FGS coding process and illustrates how these levels of distortion are generated. Recall that the distortion observed by the end user is the difference between the source and its reconstructed version in the *pixel* (i.e., spatial) domain (shown as D_1 in the figure). Let us further analyze how this distortion is formed.

Initially, the source signal in the spatial domain is transformed into the DCT domain with random DCT round-off errors (which are labeled as noise ω_1 in the figure). In the DCT domain, the coefficients are separated into the base layer and the enhancement layer. After the base layer is combined with the quantized FGS signal at the receiver, it incurs the second level of distortion D_2, which is the classical frequency-domain distortion often modeled in previous work. Note, however, that we have a third level of distortion D_3 that comes from quantization errors in the FGS enhancement layer.

It is easy to notice that distortion D_1 and D_2 are equal in an ideal codec (without round-off errors) since DCT is an orthogonal transform. In real coding schemes, round-off errors ω_1 and ω_2 are very small in comparison with values of typical distortion, which for all practical purposes allows us to write $D_1 \approx D_2$ not only in theory, but also practice.

It is further easy to see that distortion D_2 is equal to D_3. Consider an original DCT coefficient x_i approximated in the base layer with a value of b_i. The corresponding DCT residue in the enhancement layer is $e_i = x_i - b_i$. Assume that e_i is quantized to some value q_i through bitplane coding (i.e., the receiver decodes q_i instead of e_i). Thus, the distorted

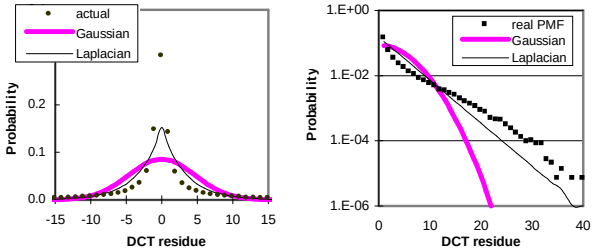

Figure 3: The PMF of DCT residue with Gaussian and Laplacian estimation (left). Logarithmic scale of the PMFs for the positive residue (right).

DCT coefficient is $b_i + q_i$, and distortion D_2 is given by:

$$D_2 = \sum_i (x_i - (b_i + q_i))^2 = \sum_i (e_i - q_i)^2 = D_3. \qquad (4)$$

Therefore, for FGS-coded sequences, the distortion in the FGS enchantment layer *alone* determines the distortion of the combined signal at the receiver. This means that R-D modeling of the enhancement layer is sufficient to describe the visual quality of video streaming. Thus, our modeling below focuses entirely on the FGS layer and develops an R-D framework that is independent of the base layer.

3.3 Source Statistical Properties

In image and video coding, Gaussian and Laplacian (double exponential) distributions are the two most popular statistical models for DCT coefficients [1], [9], [22], [23] and FGS DCT residue [22]. These models are popular often more due to their mathematical tractability rather than because they accurately describe real video source data.

To examine statistical properties of real DCT residue, we conducted an extensive analysis of the probability mass function (PMF) of DCT residue coefficients for different frames and different sequences. A typical example of what we observed is shown in Figure 3. Figure 3 (left) shows that neither Gaussian nor pure Laplacian distribution fits the sharp peak of the real PMF. Notice that a significant fraction of all coefficients are located near the peak, which means that it is very important to accurately model the actual PMF near zero. It may seem at first that the Gaussian and the Laplacian distributions can fit the tail of the real PMF in Figure 3 (left); however, close examination of the tails on the logarithmic scale (shown in Figure 3 (right)) reveals that the Gaussian distribution decays too quickly and the Laplacian distribution cannot describe the bending shape of the real PMF.

Further notice that in Figure 3 (right), the log-scaled PMF of the DCT residue can be partitioned into two straight lines, which indicates that the shape of the PMF can be approximated by a *combination* of two exponential distributions. Thus, to capture the sharp peak and heavy tails of the actual PMF, the natural choice is to use a *mixture-Laplacian* model described below.

Suppose that the DCT residue is generated by a random variable X with probability p and another random variable Y with probability $(1-p)$. Thus, assuming the corresponding density (mass) functions for these two variables are $p_X(k)$ and $p_Y(k)$, the PMF of the DCT residue is given

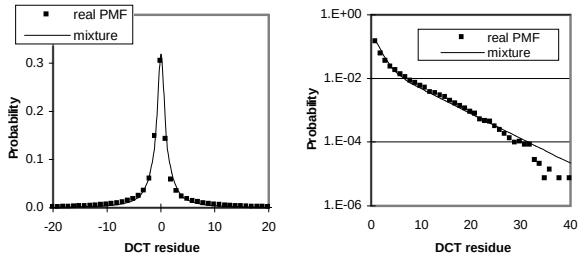

Figure 4: The real PMF and the mixture Laplacian model (left). Tails on logarithmic scale of mixture Laplacian and the real PMF (right).

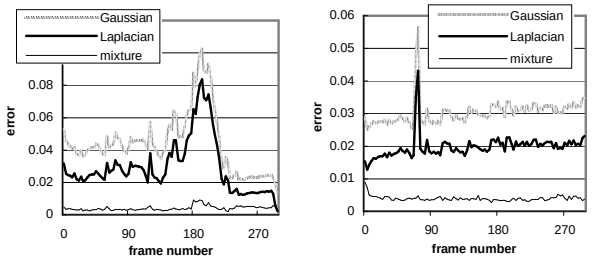

Figure 5: The absolute error of the three DCT models in Foreman CIF (left) and Coastguard CIF (right). Both sequences coded at 10 fps and 128 kb/s in the base layer.

by:

$$\begin{aligned} p(k) &= p \cdot f_X(k) + (1-p) \cdot f_Y(k) \\ &= p \cdot P(X=k) + (1-p) \cdot P(Y=k) \\ &= p\frac{\lambda_X}{2} e^{-\lambda_X |k|} + (1-p)\frac{\lambda_Y}{2} e^{-\lambda_Y |k|}, \end{aligned} \quad (5)$$

where λ_X and λ_Y are the shape parameters of the corresponding Laplacian distributions. We can further note that one Laplacian random variable (say X) concentrates the probability mass near 0 due to its low variance, whereas the other random variable (say Y) spreads out the rest of the mass across larger values due to its high variance. In practice, we use the EM (Expectation-Maximization) algorithm to obtain the estimates of parameters $\{\lambda_X, \lambda_Y, p\}$.

As illustrated in Figure 4, the mixture Laplacian distribution fits the histogram of the DCT residue much better. The discrepancy at the end of the tail in Figure 4 (right) does not affect the source model, since only very few of the samples are contained there (0.04% in this example). It should be pointed out that the mixture Laplacian distribution can also describe statistical properties of other signals with sharp peaks and heavy tails[1], such as base-layer DCT coefficients.

We next examine the discrepancy between these three models (Gaussian, Laplacian and mixture Laplacian) and the real PMF for Foreman CIF and Coastguard CIF in Figure 5. Note that the error is weighed by the amount of DCT coefficients it affects (i.e., discrepancies toward the tail of the distribution weigh less since they affect only a handful of samples).

In summary, experimental results show that one cannot directly apply classical (e.g., Gaussian or Laplacian) statistical models to DCT residue in MPEG-4 FGS. However, we observed that *the mixture-Laplacian distribution follows the majority of the real data with exceptional accuracy.*

3.4 Previous Closed-Form R-D Models

In traditional rate-distortion theory [9], distortion D is derived to be an exponential function of rate R: $D = Ee^{\alpha R}$, where α is a constant and E is a *function* of the power spectrum density (PSD) of the source. Under certain autoregressive assumptions about the source, the PSD model

[1]In statistical modeling of DCT data, "heavy" tails mean distributions decaying slower than Gaussian. This is entirely different from heavy tails found in network modeling where they mean some form of the Pareto (hyperbolic) distribution.

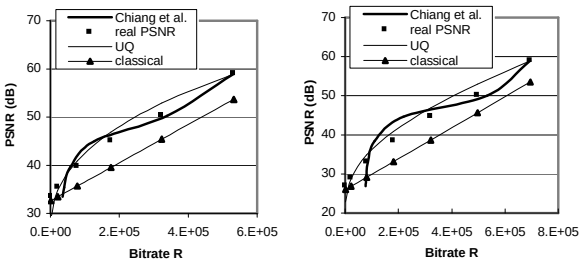

Figure 6: The model of Chiang et al. in (7), the real R-D curve, the classical model in (6) and the UQ model for frame 0 of CIF Foreman (left). The same simulation for frame 252 of CIF Foreman (right).

can be simplified into a closed form as long as the source is Gaussian and the quantization step Δ is small [9], [24]:

$$D = \gamma^2 \varepsilon^2 \sigma_x^2 2^{-2R}, \quad (6)$$

where σ_x^2 denotes signal variance as before and γ^2 is derived from the autocorrelation function of the source [9]. Notice that (6) is a scaled version of (1), which still produces straight R-D curves in the PSNR domain.

Chiang *et al.* [3] use a "quadratic" R-D model based on a Taylor expansion of the classical result in (1). Their model assumes the following shape:

$$R = aD^{-1} + bD^{-2}, \quad (7)$$

where parameters a, b are obtained from multiple empirical samples of the R-D curve. Finally, one additional popular closed-form R-D model is the widely-used uniform quantizer (UQ) shown in (2).

To illustrate the accuracy of these models, we plot the actual R-D curve and the model estimation for frames 0 and 252 of Foreman CIF in Figure 6. Observe that a large mismatch exists between these models and the real R-D curve, not only in the actual points, but also in the underlying *shape* of the curve.

3.5 Square Root R-D Model

3.5.1 Distortion Model

Assume that the quantization step applied to a given frame is Δ, which depends on the bitplane number where the server stopped transmission of the FGS layer. If the

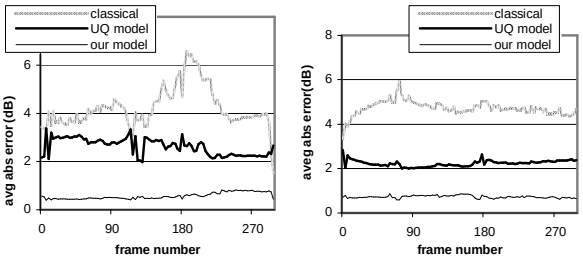

Figure 7: The average absolute error in Foreman CIF (left) and Coastguard CIF (right).

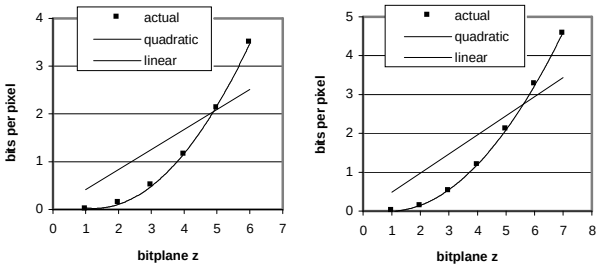

Figure 8: Quadratic model for $R(z)$. Frame 0 (left) and frame 252 (right).

maximum number of bitplanes in a given frame is n and the last transmitted bitplane is z (in the order from the most-significant to the least-significant), then $\Delta = 2^{n-z}$. Then the distortion produced by quantizer Δ is given by [7], [21]:

$$D(\Delta) = 2\sum_{k=0}^{N/\Delta}\sum_{m=k\Delta}^{(k+1)\Delta-1}(m-k\Delta)^2 p(m), \qquad (8)$$

where $p(m)$ is a (symmetric) PMF of the source DCT residue. Substituting a single exponential distribution $p(m) = ae^{bm}$ into (8) and evaluating the discrete sum, we have:

$$D(\Delta) = \frac{2a}{(1-e^{b\Delta})b}\times \left(e^{b(\Delta-1)}\left[\Delta^2 - 2\Delta\left(1+\frac{1}{b}\right)+\frac{2}{b^2}\right]-\frac{2}{b^2}\right), \qquad (9)$$

where a and b are the parameters of the generalized Laplacian distribution $ae^{-b|m|}$. To demonstrate the accuracy of (9) over two sample FGS sequences, Figure 7 plots the average absolute error of the classical model, UQ, and model (9) for Foreman CIF and Coastguard CIF. As the figure shows, (9) is a very accurate estimator of distortion $D(\Delta)$. However, the complex form of the model serves very little useful purpose. Thus, we next focus on simplifying it.

Recall that $PSNR$=$10\log_{10}(255^2/D)$ and take the logarithmic transform of distortion model (9). After omitting less-significant terms and grouping constants, we have:

$$\log D(\Delta) \approx a_1 + a_2\Delta + \log\left(b_1\Delta^2 + b_2\Delta + b_3\right), \qquad (10)$$

which can be further simplified to the following assuming a limited range of Δ found in standard coding schemes:

$$\log D(\Delta) \approx c_1\log^2\Delta + c_2\log\Delta + c_3. \qquad (11)$$

Note that (11) is a quadratic (rather than linear) function of bitplane number z and smoothly generalizes classical information-theoretic results. Thus, we can re-write (11) in terms of bitplane number z and quality PSNR:

$$PSNR(z) \approx d_1 z^2 + d_2 z + d_3. \qquad (12)$$

3.5.2 Rate Model

We conducted numerous experiments to better understand the properties of bitrate $R(z)$ as a function of bitplane z and compared them to the traditional model. Recall the tradition R-D model in (3) and notice that its rate R is a linear function of $\log(\Delta)$ (or bitplane z). This is an important observation since we earlier found out that classical distortion $PSNR(z)$ in (2) was also a linear function of bitplane z. Hence, the traditional R-D theory is a combination of *two linear functions of bitplane z.*

Carefully examining Figure 8 and many others like it, one notices that the shape of the $R(z)$ curve is not linear and is likely to be a polynomial function. Through simulations, we found that second-order polynomials (quadratic functions) were sufficient to model $R(z)$ as schematically illustrated in Figure 8. Skipping a large set of simulation results that show quadratic modeling of $R(z)$, we next combine our findings into a single R-D model and check its accuracy in the entire sequence.

3.5.3 Square-Root Model

What we have learned so far allows us to construct a close-from R-D curve $D(R)$ and generalize results of the linear traditional model to second-order polynomial functions in the bitplane domain (or the z-domain, following the notation in [8]). Consider a polynomial function for $R(z)$ in the z-domain for some constants $e_1 - e_3$:

$$R(z) \approx e_1 z^2 + e_2 z + e_3. \qquad (13)$$

Inverting the polynomial in (13), keeping in mind that $PSNR(z)$ is also a quadratic function of z, dropping insignificant terms, and combining constants:

$$PSNR(R) \approx AR + B\sqrt{R} + C. \qquad (14)$$

Notice that this result is a direct generalization of the classical formula (1), which models the PSNR as a *linear* function of rate R. Our work found substantial evidence that suggests that linear approximations in the z-domain do not produce accurate models (as evidenced by many figures in this paper) and that exploring more complicated models can bring additional insight into understanding R-D properties of complex sources and encoders. Re-writing (14) in the distortion domain, the final closed-form R-D function is a smooth generalization of the classical result in (1):

$$D(R) = 2^{aR+b\sqrt{R}+c}. \qquad (15)$$

In Figure 9, we examine the *maximum* (over all bitplanes) absolute error of our model (14), a simple quadratic model shown in Figure 1, and the usual linear model from information theory. Since models (6) and (7) are both expanded from the classical linear result, they can be combined under the linear "umbrella" of the classical model. Figure 10 shows the *average* absolute error of the same models in the same two sequences. Other extensive experimental results show that (14) significantly outperforms the classical linear model, Chiang's result [3], as well as the quadratic model.

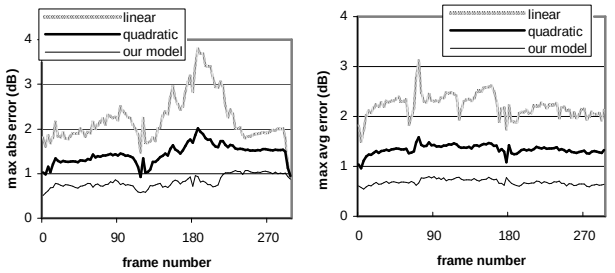

Figure 9: The maximum absolute error in Foreman CIF (left) and Coastguard CIF (right).

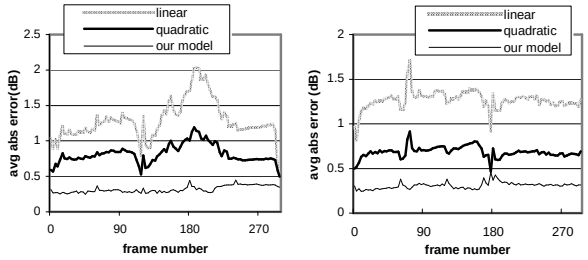

Figure 10: The average absolute error in Foreman CIF (left) and Coastguard CIF (right).

4. APPLICATION OF THE MODEL

As we mentioned in Section 2, rate control is one popular application of R-D models. The main question here is how to scale the FGS layer to both match the available bandwidth R_T (total bits allowed for the entire sequence) and achieve certain *constant* quality D after decoding. We illustrate the solution to this problem using Figure 11 (left) and a simple sequence consisting of two frames. First, the server inverts the result in (14) or (15) and obtains two $R(D)$ curves (one for each frame). Second, it generates the combined rate curve $R_1(D)+R_2(D)$, which shows the amount of *total* bits required to achieve constant D in both frames. Knowing R_T, the combined curve needs to be inverted one more time to obtain the value of D_T that provides the required total bitrate R_T. The size of individual frames is given by $R_1(D_T)$ and $R_2(D_T)$ as the final step.

In general, adding the R-D curves of each frame, we get a combined function $F(D)$, which is constrained by R_T:

$$F(D) = \sum_{i=1}^{N} R_i(D) = R_T, \tag{16}$$

where $R_i(D)$ is the R-D function of frame i and N is the number of frames in the *remainder* of the sequence. Partial summation in (16) is important since congestion control often changes its rate in the middle of actual streaming and (16) needs to be recomputed every time such change is encountered. Finding the root of (16) involves inverting $F(D)$ and evaluating

$$D_T = F^{-1}(R_T). \tag{17}$$

Even though the new R-D framework does not lead to a closed-form solution for F^{-1}, each of the individual curves can be generated with high accuracy using only a 3-point

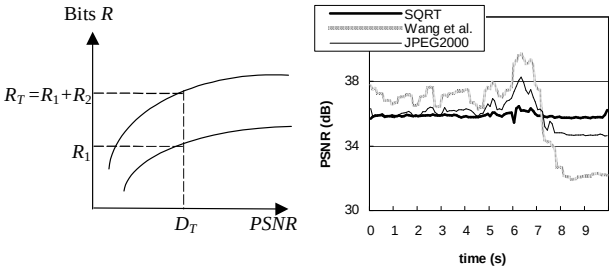

Figure 11: Given a target rate R_T, the location of constant quality D_T (left). Comparison between our model, the result of [24], and rate control in JPEG2000 (right).

interpolation and the resulting function $F(D)$ can be computed (and then inverted) very efficiently.

In Figure 11 (right), we illustrate this simple rate control algorithm (labeled as "SQRT") assuming that the channel capacity is fixed (variable channel rates are studied in the next section). The figure shows simulation results using Foreman CIF with 128 kb/s for the base layer and 768 kb/s for the enhancement layer in comparison with two other rate-control methods – those proposed in the JPEG2000 image coding standard and in Wang *et al.* [24]. Experimental results show that the new R-D framework can be successfully used to both dramatically reduce undesirable quality fluctuation during streaming and to relieve the server from expensive interpolation. The variance in PSNR between adjacent frames in the SQRT curve shown in Figure 11 (right) is only 0.04 dB.

This is the point where most FGS streaming papers stop. We find that the neither the exact method of scaling the enhancement layer (this section), nor the underlying R-D model (the previous section) are very important if the application relies on any of the wide variety of AIMD-style congestion control methods. Hence, we feel that with goals of constant-quality streaming, it becomes much more important to continue the research into the area of smooth congestion control, which is a pre-requisite to actual implementation of any of these methods. Unfortunately, the current Internet does not provide an environment where smooth (asymptotically stable) sending rates can be easily achieved; nevertheless, there are promising classes of congestion controllers for the future Internet than may fulfill these requirements. One such class is studied next.

5. CONGESTION CONTROL

5.1 Overview

There are many challenges facing Internet streaming applications, all of which stem from the lack of quality-of-service (QoS) guarantees in the transport layer. One of the primary impediments to high-quality delivery of real-time video to the end user is the *variable* channel bandwidth. Notice that even though end-to-end paths often experience relatively stationary conditions (in terms of the number of competing flows, average long-term packet loss, etc.), current congestion control methods built on top of a variety of TCP-friendly schemes cannot asymptotically converge (from a control theory point of view) to a single stationary rate or

provide a smooth "virtual" channel to the video application.

After AIMD (Additive Increase, Multiplicative Decrease) was found unsuitable for video applications due to large rate fluctuations, a major effort has been dedicated to developing smoother congestion control methods for multimedia streaming (e.g., TFRC [5] and binomial algorithms [1]). Nevertheless, these newly-developed methods are not asymptotically stable, nor do they have any stationary points in the feasible operating range of a typical application. Note that unless a video application can employ a stable congestion controller, any attempts to provide constant-quality streaming will be moot.

In this section, we study continuous-feedback congestion controllers proposed by Kelly *et al.* [12] and investigate whether their performance provides the necessary foundation for achieving the goals of this paper.

5.2 Continuous-Feedback Controllers

Recall that TCP and classical binary-feedback methods (such as AIMD and binomial algorithms) rely on packet loss in order to increase or decrease their rates. Since the decision about changing the current rate is binary, we can summarize their control functions as following:

$$\frac{dr}{dt} = (1 - sgn(p))F(r) - sgn(p)G(r), \tag{18}$$

where $r(t)$ is the rate, $p(t)$ is packet loss, $F(r)$ is the increase function, and $G(r)$ is the decrease function. Notice that with a reasonable choice of functions F and G, the right side of (18) does not have roots, which means that the equation does not have stationary points. Since (18) cannot be stabilized, it must oscillate or diverge. It is easy to show that under certain mild conditions on $F(r)$ and $G(r)$ [1], [16], (18) oscillates around the equilibrium (equal-share) rate. The amount of oscillations depends on the choice of $F(r)$ and $G(r)$ and typically leads to a trade-off between the size of oscillations and the rate of response to congestion signals. Thus, controls that produce small oscillations are usually susceptible to more packet loss due to their reluctance to back off during congestion.

What is interesting about binary-feedback methods is that they typically do not possess any methods that can force the oscillations to asymptotically decay to zero, even under *stationary* cross-traffic conditions. Therefore, we seek alternative methods that provide this functionality and are provably stable under both immediate and *delayed* feedback. One such alternative is given by Kelly's congestion control framework called *proportional fairness* [12]:

$$\frac{dr}{dt} = r(\alpha U'(r) - \beta \sum_{l \in P} p_l), \tag{19}$$

where $U(r) = \log(r)$ is the utility function of the end user and p_l is the price that the flow pays for using resource (router) l along the end-to-end path P. Kelly's controls have received significant attention in the theoretical networking community [10], [12], [13], [17]; however, their application in real networks or streaming applications has been limited.

Notice several differences (borderline simplifications) of the original framework (19), which are necessary to make this controller practical. First, it is common to use packet loss as the continuous feedback (instead of the price) simply because the current Internet is still best-effort and prices are a meaningless metric for individual routers. Second, instead of summing up the packet loss experienced by *all* routers of an end-to-end path, it sometimes makes more sense to use the *maximum* packet loss among these routers in order to match the rate of the application to the bandwidth of the *slowest* link in the path:

$$p(t) = \max_{l \in P} p_l. \tag{20}$$

Another option is to use the common *end-to-end* notion of packet loss where the flow measures combined loss experienced by its packet over the entire path. Since packet loss at individual routers is not additive (i.e., if loss is 70% in router A and 50% in router B, the combined loss is *not* 120%), the resulting end-to-end measurement is given by:

$$p(t) = 1 - \prod_{l \in P} (1 - p_l). \tag{21}$$

In general, the question of whether max-min fairness at each resource offers undisputed benefits over proportional fairness [12] or other types of fairness (such as minimum potential delay [13], [18]) is a topic of ongoing debate, which we do not address in this paper.

Expanding (19) using a *single* feedback $p(t)$ of the most-congested resource or the standard end-to-end feedback, we have a more application-friendly version of the controller:

$$\frac{dr(t)}{dt} = \alpha - \beta p(t) r(t). \tag{22}$$

Notice that when the application decides to rely on some form of AQM (Active Queue Management) inside the routers to feed back the value of $p(t)$, this framework aligns well with other next-generation congestion controllers such as XCP [11]. To show that the overhead needed to generate the feedback is very reasonable (often even less than required by RED or ECN), consider the simplest shape of p_l:

$$p_l(t) = \frac{\left(\sum_i r_i(t) - C_l\right)^+}{\sum_i r_i(t)}, \tag{23}$$

where r_i is the sending rate of the i-th flow passing through resource l, C_l is the speed of the resource (i.e., its outgoing bandwidth), and $(\cdot)^+$ represents $\max(\cdot, 0)$. Each router needs to maintain one variable with the total number of bytes placed in the outgoing buffer during the last T time units. At the end of each interval, this counter is divided by T to obtain an estimate of $\sum_i r_i(t)$, which is then used to calculate p_l using (23). The new value of p_l is inserted into each passing packet as long as the corresponding p_{l-1} contained in the packet is *lower* than the value computed by this router. Notice that the router does not need to count the number of flows or estimate the individual rates r_i. This means that the feedback is based on the *aggregate* flow rate $R(t) = \sum_i r_i(t)$ rather than on individual flow rates. This in general increases the scalability of these AQM functions inside each router.

Kelly controls have been shown to be stable under arbitrary delays both in continuous and *discrete* cases [10], [17]. On the other hand, XCP and other recent methods (e.g., [25]) have only been analyzed in the Laplacian domain assuming continuous derivatives (i.e., arbitrarily small steps during control actions) and zero feedback delays. Their stability in the presence of delayed feedback or discrete control equations is unknown.

The final subtle difference between other next-generation controllers and (23) is that Kelly controls do not *necessarily*

require AQM support. Assuming that only one resource is heavily congested, feedback $p(t)$ in (20) can be estimated using end-to-end measurements. Alternatively, the applications may decide to explicitly use (21) in their control equation. In either event, accurate end-to-end estimation of packet loss is still a difficult problem (as for example is demonstrated in [5]). The difficulty appears to be surmountable since under stationary cross-traffic conditions, recursive Kalman filters typically can provide an asymptotically accurate estimate of $p(t)$. We leave these details for future work and in the meantime, study how a network of AQM-equipped resources enables video streaming applications to deliver constant-quality presentations to the end user.

5.3 Basic Properties

Given the expression of $p_l(t)$ in (23), each flow in (22) has a single stationary point given by:

$$r^* = \frac{\alpha}{\beta p^*} = \frac{C_l}{n} + \frac{\alpha}{\beta}, \tag{24}$$

where p^* is the packet loss in the stationary state, C_l is the speed of the most congested resource for flow r, and n is the number of flows sharing that resource. Notice that the stationary point does *not* depend of the RTT of the flow, which means that flows with different round-trip delays share the resource equally. Furthermore, it is easy to demonstrate that the control equation (22) converges to the stationary point and remains asymptotically stable under arbitrary feedback delays [10], [17]. Thus, the main controller (22) is both fair and stable under a wide range of realistic conditions.

Notice several limitations of this framework. First, the stationary point has a strictly non-zero packet loss p^*:

$$p^* = \frac{n\alpha}{C_l\beta + n\alpha} > 0. \tag{25}$$

This is not a major problem since (23) can be adapted to move the stationary point below C_l (see below). However, the presence of n in the numerator of (25) prevents Kelly controls from staying below capacity C_l as the number of flows grows arbitrarily large. Another way of showing this is to analyze the total load $R(t) = \sum_i r_i(t)$ on a given bottleneck resource in the stationary point. From (24), we have:

$$R^* = nr^* = C_l + \frac{n\alpha}{\beta}, \tag{26}$$

where R^* is the total rate at the router in the equilibrium point. Thus, the amount of overshoot $\frac{n\alpha}{\beta}$ in the stationary state (and hence packet loss p^*) grows linearly with n (assuming α is positive). Therefore, even under AQM feedback in (23), packet loss cannot remain fixed as the number of flows n grows. Linear increase in packet loss is not as severe as in other methods (such as n^2 in AIMD or n^3 in IIAD [16]), but it does represent undesirable performance when the number of flows becomes very large.

5.4 Exponential Convergence to Efficiency

The next property of AQM-based feedback is the ability of Kelly controls to converge exponentially to the equilibrium point. Since the router explicitly computes (23), there is no inherent limitation on keeping the feedback positive. Hence, relaxing this requirement, (23) becomes:

$$p_l(t) = \frac{\sum_i r_i(t) - C_l}{\sum_i r_i(t)}, \tag{27}$$

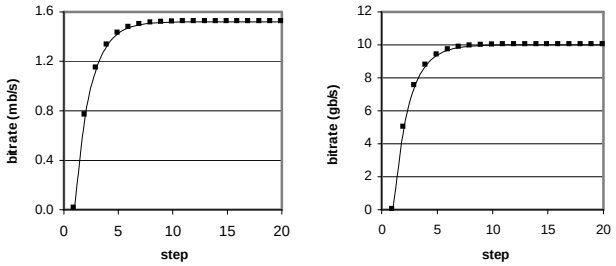

Figure 12: Exponential convergence of rates for C = 1.5 mb/s (left) and C = 10 gb/s (right).

as long as there are is some traffic passing through the router (i.e., $\sum_i r_i > 0$). Note that exponential convergence to the stationary point holds for the combined rate $R(t)$ and *not* the individual rates $r_i(t)$. Thus, this control converges to *efficiency* exponentially, while convergence to *fairness* is (potentially) slower. Exponential convergence to efficiency can be shown as follows. Summing up (22) for all n flows and using (27):

$$\frac{dR(t)}{dt} = n\alpha - \beta R(t)p(t) = n\alpha - \beta(R(t) - C). \tag{28}$$

Notice that (28) admits a closed-form solution:

$$R(t) = (C + \frac{n\alpha}{\beta})\left[1 - e^{-\beta t}\right] + R(0)e^{-\beta t}, \tag{29}$$

where $R(0)$ is the initial combined rate of all flows. Parameter β *solely* determines the rate of convergence (this was not the case with AIMD, where α was responsible for convergence to efficiency). Using $\beta = 0.5$ and $\alpha = 10$ kb/s, it takes 8 steps for a single-flow to fill a 1.5 mb/s T1 bottleneck and it takes only 16 steps for the same flow to fill a 10 gb/s link. This is illustrated in Figure 12. Note that both flows reach within 5% of C in just 6 steps.

Therefore, we can conclude that under AQM feedback, Kelly controls are highly suitable for high-speed networks of the future Internet and, along with XCP [11], may provide a new avenue of high-speed congestion control.

5.5 Zero Packet Loss

One obvious way to control the increase in packet loss as the number of flows becomes large is to explicitly estimate n inside the router and adapt (27) to become:

$$p_l(t) = \frac{\sum_i r_i(t) - \left(C_l - \frac{n\alpha}{\beta}\right)}{\sum_i r_i(t)}. \tag{30}$$

Of course, feedback (30) no longer represents packet loss, but this should not make any difference for the end flows. Under these assumptions, re-write (26) again assuming that link l is the most congested router:

$$R^* = \frac{n\alpha}{\beta\frac{(R^* - C_l + n\alpha/\beta)}{R^*}} = \frac{nR^*\alpha}{\beta R^* - \beta C_l + n\alpha}. \tag{31}$$

and solve for R^*:

$$R^* = C_l. \tag{32}$$

Therefore, the knowledge of the number of flows allows distributed and asymptotically stable controls of the Kelly

framework to achieve both constant and zero packet loss as shown in (32). Furthermore, link utilization in this case stays at 100%.

5.6 Discussion

There is a wide range of possible uses of Kelly controls in the context of the Internet. We examined several simple methods, which can be broadly partitioned into two categories – end-to-end and AQM. The end-to-end methods have many limitations: a) packet loss $p(t)$ must be estimated at the receiver; b) the stationary point p^* is strictly positive (i.e., bottleneck buffers are constantly full); c) packet loss grows linearly with the number of flows n; and d) the convergence to efficiency is *linear*.

The AQM methods can also be divided into two categories – those that estimate the number of flows n and those that do not. In both cases, we gain exponential convergence to efficiency, while in the former case, we also avoid the packet-loss increase problem.

This analysis reflects the general philosophy of distributed congestion control – the more flows know about the state of the network, the better control can be accomplished. Among a wide range of methods, controllers with *distributed* control functions are generally more desirable. Thus, XCP and various ATM ABR (Available Bitrate) Explicit Rate [2] methods that monitor queue size and implement router-based controllers cannot be fully classified as "distributed."

On the other hand, Kelly and pure end-to-end methods (such as TCP) are inherently *end-flow* controls. The only difference between the two is that Kelly controls gradually become smoother and nicer as *additional* information becomes available in router feedback, but neither of them absolutely requires such feedback to operate. It is to be seen whether ATM-like congestion control inside the routers will overpower end-flow congestion control. In the meantime, we use Kelly controls as the model of one of the many possible controllers in future high-speed networks since they possess many appealing characteristics for real-time applications.

6. SIMULATIONS

In this section, we examine the PSNR quality curves when the target rate R_T is not know a-priori, but is rather supplied by real-time congestion control. We obtained the traces of $r(t)$ from ns2 simulations and then applied them to the video scaling algorithm offline. We should point out that one limitation of this approach is that we did not take into account the effect of lost packets during the simulation on the quality of the stream. This is reasonable in streaming scenarios where the application protects its packets by FEC or some form of retransmission. Since in Kelly controls, the amount of packet loss p^* in the steady state is fixed and known to the end flow once it reaches the equilibrium, it becomes easy to send enough FEC to cover the exact amount of lost data. In general, we do not claim that this is an exhaustive congestion control simulation since much more thorough examination of these controls over real networks is required before making any far-reaching conclusions.

First examine the typical PSNR curve produced by AIMD (1,0.5) and AQM Kelly controls in Figure 13 (left). The simulations are run over a single bottleneck resource of capacity $C = 1$ mb/s, the round-trip delay is 100 ms, and there is only one flow at the link at any given time. As the figure shows, both controls at first follow the PSNR of the base layer since

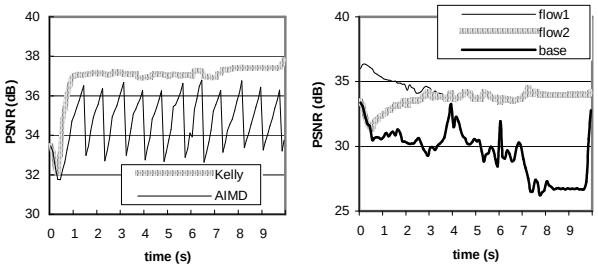

Figure 13: Comparison of AIMD and Kelly controls over a 1 mb/s bottleneck link (left). Kelly controls with two flows starting in unfair states (right).

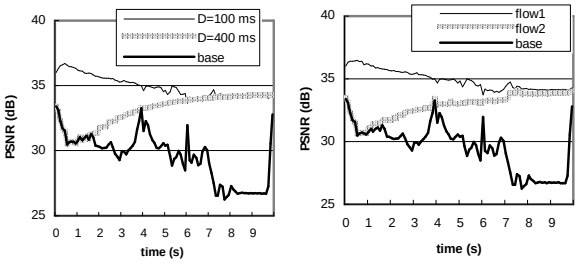

Figure 14: Comparison of PSNR for two flows with different (but fixed) round-trip delays D (left). Two flows with random round-trip delays (right).

there is not enough discovered bandwidth to send any FGS data. Once this stage is passed, both controls achieve high PNSR; however, the difference is that AIMD backs off by half upon every packet loss, while Kelly controls eventually stabilize at a fixed rate. Rate fluctuation in AIMD results in periodic jumps (sometimes as high as 4 dB) throughout the entire sequence.

Figure 13 (right) shows another scenario where two Kelly flows are sharing the same bottleneck link C under identical 100-ms round-trip delays. Flow_1 in the figure is started with $r_1(0) = C$ and flow_2 is started with its base-layer bandwidth. The two flows converge to a fair allocation at approximately $t = 3$ seconds and then follow the same flat quality curve.

The next issue to examine is whether different round-trip delays D have any effect on fairness. Figure 14 (left) shows two flows started in the same unfair states as in Figure 13 (right), but this time the delay of flow_2 is four times larger than the delay of flow_1 (400 and 100 ms, respectively). Since the second flow responds to congestion feedback slower, the convergence to fairness is slower and the two flows are stabilized at point $t = 7$ seconds. Notice that if flow_1 had larger RTTs, the convergence would have been quicker, because flow_2 would have been able to claim its bandwidth faster due to smaller RTT.

The effect of *random* feedback delays on the PSNR quality is shown in Figure 14 (right), where each flow experiences a random feedback delay uniformly distributed between 100 and 400 ms (the initial rates are the same as before). The convergence is somewhat slower than in the previous examples, but at time $t = 8$ seconds, both flows reach a fair allocation of bandwidth at the bottleneck link.

Finally examine the case of $n = 10$ flows over a faster

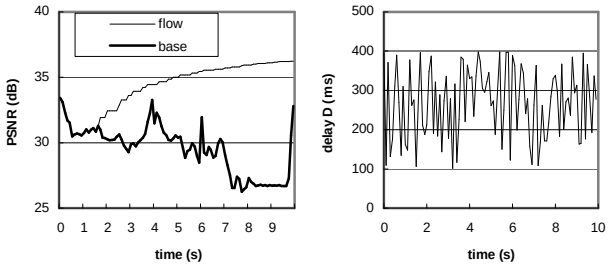

Figure 15: A single-flow PSNR when $n = 10$ flows share a 10 mb/s bottleneck link (left). Random delay D for the flow (right).

bottleneck $C = 10$ mb/s. In this case, one flow initially occupies the whole bandwidth and then 9 other flows enter the path. All delays are random between 100 and 400 ms. Figure 15 (left) shows the trajectory of one (randomly selected) flow. As the figure shows, at first only the base layer is transmitted, but starting at $t = 2$ seconds, the FGS layer "kicks in" and the flow smoothly converges to 37 dB without any oscillations. The time to stabilize at 37 dB is approximately 9.5 seconds, which appears to be reasonable under many streaming conditions. The variation in delay D is shown on the right side of the same figure.

In summary, Kelly controls converge to equilibrium without oscillation and then stay there as long as the number of flows at the bottleneck remains fixed. When new flows join or leave, the transition between fair (equilibrium) points is monotonic in most situations. This provides a nice foundation for video-on-demand and other entertainment-oriented video services where each flow is long-lived and can take full advantage of this smooth congestion control framework.

7. CONCLUSION

This paper analyzed the question of representing the empirical R-D curves with the smallest number of interpolation points and found that only *three* points were sufficient to describe rate-quality tradeoffs of scalable FGS coders. We successfully applied this analysis to create a simple rate adjustment algorithm that can work well with a variety of feedback-based congestion controllers. We further studied one class of smooth controls based on Kelly's proportional fairness and found them to work very well in an AQM environment of the future Internet. Our future work involves analysis of *end-to-end* Kelly controls, reduction of the effect of packet loss on the video quality, and various decoder-based buffer-management techniques.

8. REFERENCES

[1] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms," *IEEE INFOCOM*, April 2001.

[2] L. Benmohamed and S.M. Meerkov, "Feedback Control of Congestion in Packet Switching Networks: The Case of a Single Congested Node," *IEEE/ACM Transactions on Networking*, vol. 1, no. 6, December 1993.

[3] T. Chiang and Y.Q. Zhang, "A new Rate Control Scheme Using Quadratic Distortion Mode," *IEEE Trans. CSVT*, vol. 7, Feb. 1997.

[4] T.M. Cover and J.A. Thomas, "Elements of Information Theory," Wiley, New York, NY, 1991.

[5] S. Floyd, M. Handley, and J. Padhye, "Equation-Based Congestion Control for Unicast Applications," *ACM SIGCOMM*, September 2000.

[6] R.M. Gray and D.L. Neuhoff, "Quantization," *IEEE Trans. on Information Theory,* vol. 44, Oct. 1998.

[7] H.-M. Hang and J.-J. Chen, "Source model for transform video coder and its application. I. Fundamental theory," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 7, April 1997.

[8] Z. He and S. K.Mitra, "A Unified Rate-Distortion Analysis Framework for Transform Coding," *IEEE Trans. CSVT*, vol. 11, Dec. 2001.

[9] N. Jayant and P.Noll, *Digital Coding of Waveforms*, Englewood Cliffs, NJ: Prentice Hall,1984.

[10] R. Johari and D. Tan, "End-to-End Congestion Control for the Internet: Delays and Stability," *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, December 2001.

[11] D. Katabi, M. Handley, and C. Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks," *ACM SIGCOMM*, 2002.

[12] F. P. Kelly, A. Maulloo, and D. Tan, "Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability," *Journal of the Operational Research Society*, 49, 1998.

[13] S. Kunniyur and R. Srikant, "End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN marks," *IEEE INFOCOM*, March 2000.

[14] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 Video Standard," *IEEE Trans. Circuits and Systems for Video Technology*, March 2001.

[15] J. Lin, A. Ortega, "Bit-rate control using piecewise approximation rate-distortion characteristics," *IEEE Trans. CSVT*, vol. 8, Aug. 1998.

[16] D. Loguinov and H. Radha, "Increase-Decrease Congestion Control for Real-time Streaming: Scalability," *IEEE INFOCOM*, June 2002.

[17] L. Massoulié, "Stability of Distributed Congestion Control with Heterogeneous Feedback Delays," *IEEE Transactions on Automatic Control*, vol. 47, no. 6, June 2002.

[18] L. Massoulié and J. Roberts, "Bandwidth Sharing: Objectives and Algorithms," *IEEE INFOCOM*, March 1999.

[19] MPEG, "ISO/IEC 14496-5/PDAM3 (FGS Reference Software)," MPEG 2001/N3096, Jan.2001.

[20] H. Radha, M.V. Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. Multimedia*, vol. 3, Mar. 2001.

[21] N.M. Rajpoot, "Simulation of the Rate-Distortion Behaviour of a Memoryless Laplacian Source," *Middle Eastern Symposium on Simulation and Modelling*, September 2002.

[22] S.R. Smoot and L.A. Rowe, "Laplacian Model for AC DCT Terms in Image and Video Coding," *Ninth Image and Multidimensional Signal Processing workshop*, March 1996.

[23] S.R. Smoot and L.A. Rowe, "Study of DCT Coefficient Distributions," *SPIE Symposium on Electr. Imaging*, Feb. 1996.

[24] Q. Wang, Z. Xiong, F. Wu, and S. Li, "Optimal Rate Allocation for Progressive Fine Granularity Scalable Video Coding," *IEEE Signal Processing Letters*, vol. 9, Feb. 2002.

[25] J.T. Wen and M. Arcak, "A Unifying Passivity Framework for Network Flow Control," *IEEE INFOCOM*, 2003.

[26] D. Wu, Y. T. Hou, and Y.-Q. Zhang, "Transporting Real-time Video over the Internet: Challenges and Approaches," *Proceedings of the IEEE*, vol. 88, Dec. 2000.

[27] L. Zhao, J. W. Kim, and C.-C. Kuo, "MPEG-4 FGS Video Streaming with Constant-Quality Rate Control and Differentiated Forwarding," *VCIP*, 2002.

[28] X.J. Zhao, Y.W. He, S.Q. Yang, and Y.Z. Zhong, "Rate Allocation of Equal Image Quality for MPEG-4 FGS Video Streaming," *Packet Video Workshop*, April 2002.

[29] X.M. Zhang, A.Vetro, Y.Q. Shi, and H. Sun, "Constant Quality Constrained Rate Allocation for FGS Video Coded Bitstreams," *VCIP*, 2002.