June 2005

Real-Time Monitoring of Video Quality in IP Networks

Shu Tao
University of Pennsylvania

John Apostolopoulos
Hewlett-Packard Laboratories

Roch A. Guérin
University of Pennsylvania, guerin@acm.org

Real-Time Monitoring of Video Quality in IP Networks

Abstract
This paper investigates the problem of assessing the quality of video transmitted over IP networks. Our goal is to develop a methodology that is both reasonably accurate and simple enough to support the large-scale deployments that the increasing use of video over IP are likely to demand. For that purpose, we focus on developing an approach that is capable of mapping network statistics, e.g., packet losses, available from simple measurements, to the quality of video sequences reconstructed by receivers. A first step in that direction is a loss-distortion model that accounts for the impact of network losses on video quality, as a function of application-specific parameters such as the video codec and loss recovery technique, coded bit rate, packetization, video characteristics, etc. The model, although accurate, is poorly suited to large-scale, on-line monitoring, because of its dependency on many parameters that are difficult to estimate in real-time. As a result, we introduce a "relative quality" metric that bypasses this problem by measuring video quality against a quality benchmark that the network is expected to provide. The approach offers a lightweight video quality monitoring solution that is suitable for large-scale deployments. We assess its feasibility and accuracy through extensive simulations and experiments.

Keywords
Video, Quality, Monitoring

Real-Time Monitoring of Video Quality in IP Networks

Shu Tao
University of Pennsylvania
Philadelphia, PA 19104
shutao@seas.upenn.edu

John Apostolopoulos
Hewlett-Packard Labs
Palo Alto, CA 94304
japos@hpl.hp.com

Roch Guérin
University of Pennsylvania
Philadelphia, PA 19104
guerin@ee.upenn.edu

ABSTRACT

This paper investigates the problem of assessing the quality of video transmitted over IP networks. Our goal is to develop a methodology that is both reasonably accurate and simple enough to support the large-scale deployments that the increasing use of video over IP are likely to demand. For that purpose, we focus on developing an approach that is capable of mapping network statistics, e.g., packet losses, available from simple measurements, to the quality of video sequences reconstructed by receivers. A first step in that direction is a loss-distortion model that accounts for the impact of network losses on video quality, as a function of application-specific parameters such as the video codec and loss recovery technique, coded bit rate, packetization, video characteristics, etc. The model, although accurate, is poorly suited to large-scale, on-line monitoring, because of its dependency on many parameters that are difficult to estimate in real-time. As a result, we introduce a "relative quality" metric that bypasses this problem by measuring video quality against a quality benchmark that the network is expected to provide. The approach offers a lightweight video quality monitoring solution that is suitable for large-scale deployments. We assess its feasibility and accuracy through extensive simulations and experiments.

Categories and Subject Descriptors:
C.2.3 [Computer-Communication Networks]: Network Operations

General Terms: Measurement, Experimentation

Keywords: Video Quality, Loss, Model

1. INTRODUCTION

With the greater availability of broadband access, delivering video through IP networks (e.g., the emerging IPTV service) has become an increasingly attractive solution to service providers. However, IP networks can subject video to a variety of impairments because of packet loss and delay jitter. Assessing the impact of those impairments, and therefore the ability of IP networks to deliver video of consistent quality, calls for tools capable of continuously monitoring the quality of video transmitted over different network paths. This paper describes a simple approach that is suitable for large-scale, real-time monitoring of video quality over IP networks.

The material presented in this paper is based upon work supported by the National Science Foundation under Grant No. 9906855.

The most straightforward solution for video quality assessment is to compare the reconstructed video sequence at the receiver with the original video sequence at the sender [8]. Obviously, this is unsuitable for on-line usage, as it requires the availability of both the received and the original videos. To develop an on-line quality estimation scheme, we have to rely on approaches that first measure network losses, and then use these loss measures to generate the video quality estimates according to available loss-distortion models.

Video quality is jointly affected by various network-dependent and application-specific factors. For instance, packet losses and delay jitter (which also translates into losses in the playback buffer) are the major network-dependent factors, while video codec and loss recovery technique, coding bit rate, packetization scheme, and content characteristics are the major application-specific factors that affect video quality and its sensitivity to network errors. Most of the prior work on loss-distortion modeling, e.g., [2, 3, 4, 5], focus on only some of the network or application factors. For instance, in [5], distortion in the decoded frame sequence was modeled as a function of the loss rate. In [2], this model was further extended to accommodate the effects of different loss patterns. Similarly, the models developed in [4] aim at directly translating loss sequences into video quality estimates. In contrast to these previous works, we develop a model that characterizes the relation between packet loss and video distortion as a function of specific video codec and loss recovery technique, coding bit rate, packetization, and content characteristics. The proposed loss-distortion method is generally applicable to any motion-compensated video compression scheme, e.g., any MPEG-x or H.26x codec. In our study, we use two different and practically important video codecs – MPEG-2 and H.264/AVC – to conduct simulations and experiments and demonstrate the effectiveness of our approach.

Although our model proved accurate in translating loss statistics into video quality estimates, it still requires evaluating several application parameters. In particular, it relies on an accurate estimate of the distortion caused by losing certain frames or slices (independently decodable portions of a frame). While this information can be obtained by conducting offline simulations [2] or parsing the transmitted video stream [4], such an approach is not scalable, especially when the network is used to distribute a variety of video content to a large number of customers. As a result, we seek to develop a quality estimation technique that does not rely on the knowledge of specific video characteristics. Our second contribution in this paper is, therefore, the proposal of a video quality metric, relative PSNR (rPSNR), which can be evaluated without knowledge of the video characteristics, yet still is capable of capturing video quality variations on a network path. rPSNR is a met-

ric relative to the quality of video transmitted over a benchmark network path that delivers the desired performance guarantees. Using this metric, video quality can be estimated using only network statistics and some basic configuration parameters of the video application. Our study demonstrates that rPSNR is capable of accurately estimating video quality across a broad range of network conditions and variations in content characteristics.

The remainder of this paper is organized as follows. In Section 2, we present the loss-distortion model and discuss how it captures the impact of codec selection, coding bit rate, packetization, and video characteristics on video quality. Section 3 introduces the rPSNR metric and demonstrates its effectiveness in path quality estimation. Finally, Section 4 concludes the paper with a summary of our findings.

2. LOSS-DISTORTION MODELING

In order to estimate video quality, we need to first investigate the relation between packet losses and distortions in the decoded video. In the following analysis, we use notation from [2], and measure video distortion in terms of the Mean Square Error (MSE). Consider a video sequence with frames of size $N_1 \times N_2$ pixels, we use $f[k]$ (of size $N_1 \times N_2$) to denote the 1-D vector obtained by line-scanning frame k, and $\hat{f}[k]$ to denote the corresponding frame restored by the decoder. Thus, the error signal in frame k is $e[k] = \hat{f}[k] - f[k]$, which represents the signal impairment in frame k caused by packet losses. The MSE in frame k is defined as

$$\sigma^2[k] = (e^T[k] \cdot e[k])/(N_1 \cdot N_2). \quad (1)$$

The total distortion for a video sequence is the MSE averaged over all its frames. The value of $\sigma^2[k]$ caused by a loss event is affected by several network and application-dependent factors. For example, the length of a loss burst impacts the number of pixels and the number of subsequent frames affected by the loss event, where the latter also depends on the number of packets per coded frame. Conversely, the error concealment techniques employed by the decoder, together with the prediction strategy applied at the encoder and the characteristics of the video content itself (i.e., the spatial-temporal correlation between different macroblocks), play a role in determining the corresponding distortion in the decoded video.

2.1 Basic model

An important issue in modeling the distortion that a loss event can cause to predictively encoded video, is the extent to which the resulting error propagates across frames. Specifically, since temporal prediction introduces dependencies between adjacent frames, a single packet loss affects not only the frame with data carried in the missing packet, but also other frames with coding dependencies on it. Fortunately, because of the explicit or implicit spatial filtering applied at the decoder (which can be modeled as a low pass filter [5]), the error signal introduced by a lost packet tends to decay over time. If an error in frame k has a resulting MSE of $\sigma^2[k]$, the power of the propagated error in frame $(k+i)$ can be approximated as [2]:

$$\sigma^2[k+i] = \sigma^2[k] \cdot \gamma^i. \quad (2)$$

The attenuation factor γ ($\gamma < 1$) accounts for the effect of spatial filtering, and therefore is dependent on the power spectrum density of the error signal and the spatial filtering applied by the decoder, i.e., varies as a function of the video characteristics and decoder processing.

To limit error propagation, periodic intra coding is often used in video compression. As a result, errors in one frame only propagate until corresponding macroblocks (or entire frame) are refreshed by intra coding. For instance, if $(T-1)$ frames are predictively coded (P-frames[1]) between two consecutive intra-coded frames (I-frames), the total distortion caused by losses in frame k is $D = \sum_{i=0}^{x-1} \sigma^2[k+i]$, where x is the number of frames from where the original loss occurs (frame k) to the next I-frame.

We first model the average distortion caused by losing one slice[2] in a frame. We assume that the initial distortion caused by a lost slice is a constant, σ_S^2, and that the location x of the frame with the lost slice is uniformly distributed in $[0, T-1]$. On average, the total distortion caused by losing a single slice is then

$$D_1 = \sum_{i=0}^{T-1} \sigma_S^2 \cdot \gamma^i (1 - \frac{i}{T}) = \frac{\gamma^{T+1} - (T+1)\gamma + T}{T(1-\gamma)^2} \sigma_S^2 = \alpha \cdot \sigma_S^2, \quad (3)$$

where α is a function of γ and T, and accounts for the total propagation effect of the error signal. The values of α and D_1 can be estimated for individual slice losses by simulating the losses and measuring the MSE in the decoded frames, as demonstrated in [2]. However, since we are mainly interested in the average distortion over the entire video sequence instead of the distortion in individual frames, we will use the average values of both α and D_1 in our modeling.

When losing n ($n > 1$) consecutive packets in a single loss event, $f(n)$ slices will be affected. Here $f(n)$ is the mapping from the number of lost packets to the number of lost slices, which is specific to the implementation of the codec and loss recovery technique. Given $f(n)$, we can then model the resulting distortion as proportional to the distortion caused by an individual slice loss, i.e.,

$$D_n = f(n) D_1. \quad (4)$$

As studied in [2, 6], this additive model may slightly underestimate the distortion in the case of bursty losses. However, it greatly simplifies the final model. More important, as we show later, this simplification is key in enabling us to develop a video quality metric independent of individual video characteristics.

We define m as the number of packets transmitted between two consecutive loss events, or loss distance. Let P_n denote the probability of having n consecutive packets lost in a loss event, and P_m denote the probability of having two consecutive loss events separated by m packets. We assume that each frame is transmitted using L packets, and that n and m are independent. Thus, the expected MSE of the reconstructed video is given by

$$\overline{D} = \frac{\sum_n P_n D_n}{\sum_m P_m \, (m/L)} = \frac{\overline{f(n)}}{\overline{m}} L D_1. \quad (5)$$

Or equivalently,

$$\overline{D} = P_e \overline{f(n)} L D_1, \quad (6)$$

where P_e is the probability that a loss event (of any length) occurs within the video stream. P_e and $\overline{f(n)}$ capture the characteristics of the loss process seen by the video stream ($\overline{f(n)}$ is also a function of packetization and loss recovery at the decoder), while L and D_1 are specific to the codec and video content. For instance, L is typically larger when video is encoded at a higher bit rate, and D_1 is itself dependent on α and σ_S^2, as shown in Eq. (3).

[1]To simplify the analysis, here we do not consider bi-directionally predicted frames (B-frames). However, since B-frames are not used as coding reference for other frames, it is straightforward to extend our model to accommodate losses in B-frames.

[2]Recall that a slice is an independently decodable portion of a frame.

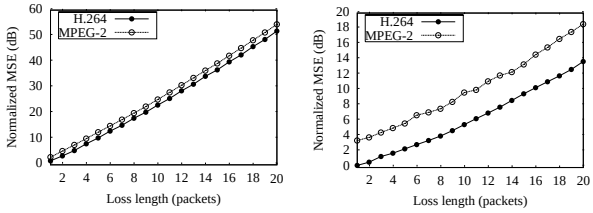

Figure 1: Average distortion caused by a single loss event with different lengths: *Foreman*, QCIF with $s = 1$, $L = 2$ (left); *Foreman*, CIF with $s = 1$, $L = 8$ (right).

2.2 The impact of codec selection

Although most video compression standards support picture segmentation in the form of slices, different codecs typically react differently in the face of slice losses. As a comparison, we study a MPEG-2 codec and a H.264 codec, implemented with different error handling capabilities. In the MPEG-2 codec, packet losses are simply handled as follows: if the decoder detects any number of packet losses in a frame, it discards the entire damaged frame and replaces it with the frame previously decoded. The H.264 codec employs more sophisticated error-concealment techniques: all slices in the received packets are decoded and the slices contained in the lost packets are recovered using the corresponding slices in the previous frame and the motion-compensation information of the other slices in the same frame. Obviously, the above two codecs will result in different loss-distortion models, since the mappings from packet losses to slice losses, $f(n)$, are different. In the MPEG-2 codec, a loss event affects not only the slices contained in the lost packets, but also the other slices in the same frame, while in the H.264 codec, only the slices in the lost packets are affected. As a consequence, the value of $\overline{f(n)}$ of the MPEG-2 codec tends to be larger than that of the H.264 codec, even if they are experiencing the same loss process. Note that the above descriptions of MPEG-2 and H.264 represent specific implementations. Some MPEG-2 based systems incorporate more sophisticated loss concealment schemes similar to those used by H.264 systems. Conversely, some H.264 systems use only simple loss handling schemes, as basic MPEG-2 systems do. However, we believe that the above examples of MPEG-2 with a simple loss recovery scheme and H.264 with a more sophisticated loss recovery scheme, are representative of many systems either deployed or being deployed and of the applications for which they are used.

We assume s slices per packet, L packets per frame, and that in each frame the starting point of a loss event (if it occurs) is uniformly distributed between packet 1 and packet L. For the MPEG-2 codec, $f(n)$ can be estimated as follows: let $r = n \bmod L$, the average number of slices affected by n consecutive packet losses is

$$f(n) = sL\left[\frac{1}{L}\frac{n}{L} + \left(1 - \frac{1}{L}\right)\left(\frac{n}{L} + 1\right)\right]$$

if $r = 0$, and

$$f(n) = sL\left[\frac{L - r + 1}{L}\left\lceil\frac{n}{L}\right\rceil + \frac{r - 1}{L}\left(\left\lceil\frac{n}{L}\right\rceil + 1\right)\right]$$

if $r \geq 1$, which in both cases simplifies to a linear function of n:

$$f(n) = s\,(n + L - 1) \tag{7}$$

For the H.264 codec, the mapping is simply

$$f(n) = sn, \tag{8}$$

Table 1: Loss statistics and MSE for *Foreman*: Bernoulli loss.

Format	s	$\overline{n}$	L	P_e	$\overline{D}$ (MPEG-2)	$\overline{D}$ (H.264)
QCIF	2	1.034	1	0.038	43.19	62.59
QCIF	1	1.041	2	0.038	85.53	54.67
CIF	2	1.037	4	0.039	184.13	42.78
CIF	1	1.038	8	0.038	402.61	39.10

Table 2: Loss statistics and MSE for *Foreman*: bursty loss.

Format	s	$\overline{n}$	L	P_e	$\overline{D}$ (MPEG-2)	$\overline{D}$ (H.264)
QCIF	2	1.724	1	0.023	53.80	78.24
QCIF	1	2.639	2	0.015	75.58	73.87
CIF	2	3.860	4	0.010	99.43	60.88
CIF	1	5.448	8	0.007	109.65	60.51

since each packet loss causes the loss of s slices. Combining Eqs. (6), (7) and (8), the overall distortion of a video sequence can be modeled as

$$\overline{D} = \begin{cases} s\,(\overline{n} + L - 1)\,P_e L D_1 & : \text{MPEG-2 codec} \\ s\overline{n}P_e L D_1 & : \text{H.264/AVC codec} \end{cases} \tag{9}$$

Note that the above model captures the effect on video distortion of (1) the packet loss patterns as expressed by $\overline{n}$ and P_e, (2) the compressed bit rate as expressed by the required number of slices per frame (given by sL), (3) the packetization strategy as expressed by L, and (4) the video codec and loss recovery mechanisms as captured for MPEG-2 and H.264.

To verify this model, we simulated a scenario involving a single loss event within the packet stream (thus, P_e in Eq.(9) is a fixed value), and varied the length of the resulting loss event from 1 to 20 packets over different runs of the simulation. The tested video sequence *Foreman* contains 300 frames and is transmitted in two video formats corresponding to different qualities and different transmission rates. Specifically, we use the standard QCIF and CIF video formats of 144x176 and 288x352 pixels/frame, coded at bit rates of 100 Kbps and 500 Kbps, respectively. Each QCIF frame contains 2 slices, while each CIF frame contains 8 slices, and we transmit each slice in a separate packet. The frame rate is 30 frames per second, for both formats. We encode the video using both the MPEG-2 and the H.264 codecs. As shown in Fig. 1, the average distortion is indeed well modeled by a linear function of $\overline{n}$ (which in this case equals n), as predicted by Eq. (9). The difference between the two codecs is also roughly constant, which again agrees with Eq. (9) given that the values of D_1 for the two codecs are close.

2.3 The impact of packetization

As mentioned earlier, how video data is packetized is another important factor that influences how video quality is affected by packet losses. The effects of packetization are two-fold. First, the number s of slices contained in a packet, together with the number L of packets used for transmitting a frame, affect the mapping from packet losses to slice losses, i.e., $f(n)$, as shown in Section 2.2. Second, video streams with different values of L sample network paths differently, and can, therefore, experience different packet-level loss processes even when transmitted on the same path. For instance, video streams configured with larger L tend to see longer packet loss bursts than those with smaller L. This effect has been analyzed in our earlier work [7].

To study the impact of packetization, we rely on a network emulator to simulate path performance variations using a Markov model with two states [7]. The two states of a path are associated with loss probability b_0 and b_1, respectively. Furthermore, the time that the

path stays in each state is exponentially distributed with mean λ_0 and λ_1. By varying the values of the above parameters, we can simulate changes not only in the packet loss rate, but also in the loss burstiness. We again use the *Foreman* video sequence coded using both MPEG-2 and H.264 codecs, in QCIF and CIF formats which we recall correspond to 2 and 8 slices, respectively. We then generate a range of packetized video streams configured with different transmission parameters, i.e., combination of s and L values that determine how slices are packed into packets, which we transmit simultaneously through the network emulator. We measure the related loss statistics, i.e., P_e and $\overline{n}$, experienced by each video stream, as well as the MSE of the decoded frame sequences.

Table 1 and 2 summarize the results of two simulations. In Table 1, we simulate a path using a Bernoulli model. The average loss rate is configured as 4% (i.e., $b_0 = b_1 = 0.04$). In Table 2, we simulate bursty losses by having $b_0 = 0$ and $b_1 = 0.9$, while still keeping the average loss rate equal to 4% by selecting appropriate values for λ_0 and λ_1. From the simulation results, we observe that packetization, specifically the number of packets per frame (L), has an important impact on video quality. As mentioned earlier, this is because different packetization schemes result in different mappings from packet losses to slice losses:

- The performance of the MPEG-2 codec degrades as L increases. This is because a packet loss affects not only the slices in that packet, but also all the other slices in the same frame. According to Eq. (7), this effect becomes more pronounced as L increases. This can be observed from both Tables 1 and 2.
- For H.264, varying L has only a minor effect on video quality, as shown in Tables 1 and 2. This is because the mapping from packet losses to slice losses is independent of L.

The effect of loss patterns is also different across codecs:

- For H.264, bursty losses typically degrade its error-concealing capability. This is because with isolated losses, it is easier for the decoder to extrapolate the lost slices from the received ones. Moreover, losing consecutive slices causes greater distortion than the distortion caused by losing the same number of slices individually. This is because of the cross correlation between the error signals in different frames when losses are bursty [2].
- For MPEG-2, the latter effect still exists, but is dominated by the multiplicative effect that its frame-based error concealment mechanism has on losses. Specifically, a single packet loss can translate into a much larger number of lost slices, i.e., all the slices of the corresponding frame, so that at equal loss rate it is better to concentrate all the lost packets in the same frame. As a result, unlike H.264, MPEG-2 video quality is better in the presence of bursty losses than Bernoulli losses. The only exception is when $L = 1$, since in this case each frame is encapsulated in a single packet, so that each lost packet corresponds to a lost frame. Hence, the cross-correlation of error signals in different frames becomes again the dominant factor when losses are bursty.

By comparing the data in the above tables and Eq. (9), we can see that our model can indeed characterize the general relations between packet losses and video distortions. However, our ability to estimate video quality is predicated on an accurate estimate of D_1 in Eq. (9). As mentioned earlier, this can be done by monitoring loss events and measuring the resulting value of D_1. This is, however, challenging in practice because the loss process on a path may not be stationary. As a result, the estimated value of D_1 will often vary over time as path conditions change. For instance, curve fitting the data in Table 1 and Table 2 yields very different values for D_1. From the data in Table 1, the QCIF video encoded with the H.264 codec and using $s = 2$ and $L = 1$ has a value of D_1 of 782, while we have $D_1 = 978$ from the corresponding data in Table 2. This indicates that relying on Eq. (9) for gauging video quality without continuously updating our estimates of D_1 may result in relatively poor accuracy of our quality estimates.

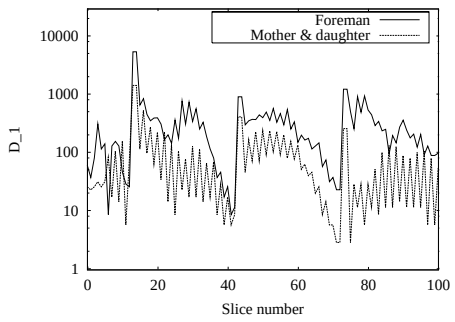

Figure 2: The value of D_1 for slices in QCIF video *Foreman* and *Mother & Daughter*. Each frame is segmented into 2 slices. Both video sequences are encoded with an INTRA period of 15 frames. The results for 50 frames (100 slices) are shown in the plot.

2.4 The impact of video characteristics

In our model, $D_1 = \alpha\sigma_S^2$ is a factor that is a function of both the implementation of the decoder and video characteristics. As shown in [5], the value of σ_S^2 depends on the power spectrum density of the error signal caused by a slice loss, and on the strength of loop filtering in the decoder [5]. Therefore, it is important to understand how video characteristics affect the resulting video quality under given loss conditions.

In general, video with higher motion makes it more difficult to infer the missing data and thereby conceal the losses. Consequently, the distortion caused by a slice loss also tends to be higher for high-motion video. For example, Figure 2 shows the total distortion caused by losing each slice (D_1) in video *Foreman* and *Mother & Daughter*. It is obvious that D_1 is typically higher for *Foreman* than for *Mother & Daughter*, as the former contains higher motions. It can also be observed that D_1 also varies within a video sequence. For instance, the error signal caused by the loss of an I-slice (e.g., slice number 15, 45, and 75) is stronger than that caused by the loss of a P-slice. Furthermore, slices in the same frame may also have different importance in video decoding. For instance, in the *Mother & Daughter* video, losing the first slice in a frame typically causes more distortion than losing the second slice, since there is typically more motion in the top half of each frame.

These observations clearly indicate that video quality estimates depend on the *specific video characteristics*. Hence, on the same path, different videos may display different qualities for the same loss pattern. Moreover, for the same video sequence, even if the path condition remains unchanged, its quality could vary with scene changes. In order to estimate the *absolute* video quality on a path, we need to dynamically estimate the impact of video characteristics (D_1), which is nontrivial [2] and only feasible when one can parse or decode the transmitted video bit streams offline. For some real-time applications, such as video conferencing, this processing is very difficult. Therefore, it is desirable to develop a video quality metric that is independent of video characteristics.

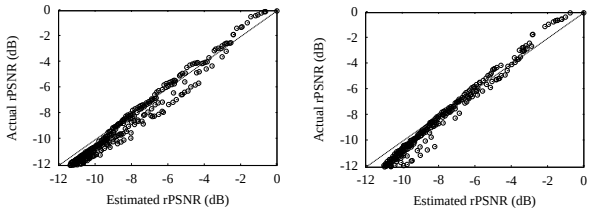

Figure 3: The robustness of the rPSNR metric in video quality estimation: *Foreman*, H.264 coding (left), and *Mother & Daughter*, MPEG-2 coding (right).

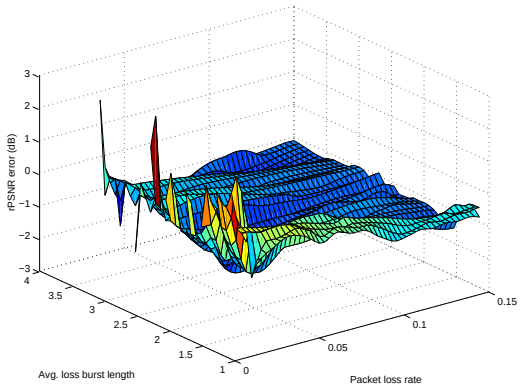

Figure 4: The estimation error for rPSNR as a function of packet loss rate and average loss burst length ($\overline{n}$).

3. ESTIMATING PATH QUALITY

Using the model developed in the previous section, the average distortion ($\overline{D}$) in a video sequence can be estimated. Thus, we can further compute the video quality on a path using the conventional measure of Peak Signal-to-Noise Ratio (PSNR), i.e.,

$$\text{PSNR} = 10\log_{10}\frac{255^2}{\overline{D}}. \tag{10}$$

However, as discussed, estimating the absolute value of PSNR requires not only loss statistics and application configurations, but the knowledge of content characteristics, i.e., D_1 is needed to determine $\overline{D}$. In this section, we introduce a *relative* quality metric, rPSNR, that can be evaluated without estimating D_1, yet still quantifies video quality variations on network paths.

3.1 Relative video quality metric

The first step is to define a reference network path that delivers pre-defined loss performance, i.e., $\overline{n} = n^0$ and $P_e = P_e^0$. This would typically be based on some lower bound of the path quality that a network service provider expects to offer its customers. Consider next that the loss performance on the path is actually $\overline{n} = n'$ and $P_e = P_e'$. The relative path quality, or rPSNR, is then defined as the difference between the actual PSNR and the target PSNR (the PSNR of the transmitted video on the reference path). In other words, rPSNR measures how far we are, quality-wise, from the quality target of the reference path. Let $\overline{D} = D'$ represent the actual video distortion on the current path, and $\overline{D} = D^0$ the video distortion on the reference path. Then, rPSNR is given by

$$\begin{aligned}\text{rPSNR} &= 10\log_{10}\frac{255^2}{D'} - 10\log_{10}\frac{255^2}{D^0} \qquad (11)\\ &= \begin{cases} 10\log_{10}\frac{(n^0+L-1)P_e^0}{(n'+L-1)P_e'} & : \text{MPEG-2} \\ 10\log_{10}\frac{n^0 P_e^0}{n' P_e'} & : \text{H.264/AVC} \end{cases}\end{aligned}$$

From the above equation, we can see that the relative quality can be estimated using only the values of n_0, P_0, L, n', and P_e'. The quantities n_0 and P_0 are predefined, the value of L is easy to determine based on application configurations. Therefore, it only remains to estimate n and P_e' that represent the loss process seen by individual video streams. The most accurate means for obtaining these values is through measurement. For instance, the network provider can install monitoring software at the client to collect the required loss statistics. If it is infeasible to directly monitor individual video streams, probing-based methods can be used to infer the loss performance experienced by different video streams. For example, in [7], we developed an approach that can be used for this purpose. We assume in this paper that accurate estimates of L, n', and P_e' are available for video quality estimation.

We use simulations to demonstrate the robustness of rPSNR as a metric that represents path quality under different loss patterns. We simulate path performance variations using a Gilbert model [1], and simulated a total of 340 distinct loss processes. The average loss rate on the simulated paths ranges from 1% to 15%. The burstiness of the loss process was also varied for each loss rate. We used a Bernoulli process with a loss rate of 1% as our reference path. For each loss model, including the reference model, we simulated the transmission of 3000 frames and measured the PSNR of each decoded frame sequence. This allows us to compute the exact value of rPSNR for each loss model, which we then compare to the value obtained from Eq. (11) using the measured loss statistics. This comparison allows us to assess the robustness and accuracy of the proposed approach in estimating relative video quality.

In Fig. 3, we show the results of the above comparison for QCIF versions of the videos *Foreman* (H.264 coding) and *Mother & Daughter* (MPEG-2 coding). Each encoded frame is encapsulated in two packets ($s = 1, L = 2$), and the frame rate is 30 frames/second. For most of the simulated loss models, the estimated rPSNR is very close to the actual value. We also repeated the simulation using videos in CIF format and with other application configurations, the results consistently showed that our estimates for rPSNR can indeed capture video quality under various loss conditions.

In Fig. 4, we plot the rPSNR estimation error (the estimated rPSNR minus the measured rPSNR) as a function of packet loss rate and average loss burst length. The plot is based on the simulation data for *Foreman* and H.264 coding. As shown in the figure, rPSNR estimation using Eq.(11) is more accurate when the loss rate is relatively low. For instance, the average estimation error is 0.64 dB when the loss rate is 1%, while this value is 0.78 dB when the loss rate is 15%. This is because when packet loss rate increases, video distortions caused by different loss events become less independent, making the modeling behind Eq. (6) less accurate. It can also be observed that the variance of the estimation error is bigger when loss rate is low. In these cases, the simulated loss process consists of very few loss events, which increases the likelihood that the loss events affect slices/frames of different importance to video decoding. The figure also shows that our estimate is less accurate when losses are bursty. This is mainly due to the simplification in the loss-distortion model discussed in Section 2.3. In an actual network, we expect that packet loss rates would be relatively low (typically less than 10%), and the actual loss burstiness at the time scale of an individual video stream should not be excessive. As a result, we expect our model to generate reasonably accurate real-time video quality estimates.

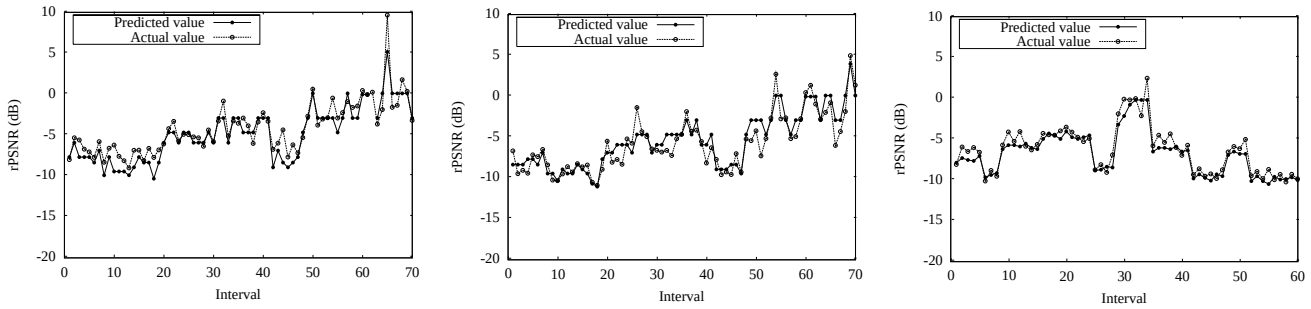

Figure 5: Estimating video quality in rPSNR on the tested path. Reference path condition: 1% Bernoulli loss. (a) MPEG-2 codec, 10 second estimation interval (left); (b) H.264 codec, 10 second estimation interval (middle); (c) H.264 codec, 100 second estimation interval (right).

3.2 Experimental validations

To further validate our method for path quality estimation, we conducted extensive experiments on real network paths over which for each transmitted video stream we recorded the resulting packet trace. The traces were used to reconstruct the video sequence that would have actually been seen by the user. The rPSNR was then computed from Eq. (11) using the loss statistics measured on each path, and compared to the rPSNR obtained directly from the decoded video sequences. Figs. 5(a) and (b) report on this comparison for the QCIF video *Highway*, using the MPEG-2 codec and the H.264 codec, respectively. The path used in the experiments of Fig. 5 was between the University of Minnesota and the University of Pennsylvania, and we focused on a period of time during which the path experienced quality variations. The rPSNR estimation and measurement were performed every 10 secs. As shown in the figure, the proposed method generates reasonably accurate estimates of quality variations of the video transmitted on the path.

From the derivation of our loss-distortion model, it is clear that the rPSNR estimate is more accurate when two conditions are met. First, when the distribution of losses on the path is stationary. Second, when the characteristics of the video content are relatively constant. Clearly, these two conditions are not always satisfied in practice. This is evident from Figs. 5(a)(b) which display instances where the actual rPSNR exhibit significant fluctuations that translate into greater differences with the estimated value. This behavior is partially caused by the relatively short duration of the estimation interval (10 secs or 300 frames). The small number of frames or slices in each interval makes it more likely that different sets are of different importance when it comes to video quality. This would then translate into different rPSNR values depending on which sets are affected by losses during the experiments. Because the rPSNR estimates computed based on our model represent the *average* quality difference between the video transmitted on the actual path and one transmitted on the reference path, the actual rPSNR values for a specific packet loss pattern afflicting a specific set of frames or slices may deviate above or below the average rPSNR estimates. Nonetheless, the predicted rPSNR values track the actual rPSNR values as shown in the Figs. 5. One option for further improving the accuracy is to increase the duration of the estimation interval, although increasing it too much would obviously affect the real-time responsiveness of the video quality estimates. Fig. 5(c) shows for the same experiment as that of Figs. 5(a)(b), the impact on the accuracy of rPSNR estimates of increasing the estimation interval to 100 secs (3000 frames), which we believe represents a reasonable compromise between responsiveness and improved accuracy. The figure shows a clear improvement in accuracy. Similar findings were observed across several other experiments.

4. CONCLUSION

This paper introduced an approach for on-line estimation of the quality of video transmitted over network paths. Our goal was to devise a lightweight solution that would allow the large-scale monitoring of video quality using only simple measurements of network performance. In particular, we wanted to avoid solutions that require detailed knowledge of video characteristics. In that context, our first contribution is the developement of a loss-distortion model that accounts for the impact of various network-dependent and application-specific factors on the quality of decoded video. Our second contribution is in using this model to define a relative video quality metric, rPSNR, that can be evaluated without parsing or decoding the transmitted video bit streams and also without requiring knowledge of D_1 – thereby leading to significant reductions in complexity. The robustness and reasonable accuracy of our rPSNR estimate were then demonstrated through a broad range of simulations and experiments conducted over real networks.

5. REFERENCES

[1] W. Jiang and H. Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *Proc. NOSSDAV*, June 2000.

[2] Y. J. Liang, J. G. Apostolopoulos, and B. Girod. Analysis of packet loss for compressed video: Does burst-length matter? In *ICASSP*, 2003.

[3] A. R. Reibman and V. Vaishampayan. Quality monitoring for compressed video subject to packet loss. In *Proc. of IEEE ICME*, July 2003.

[4] A. R. Reibman, V. Vaishampayan, and Y. Sermadevi. Quality monitoring of video over a packet network. *IEEE Trans. Multimedia*, April 2004.

[5] K. Stuhlmuller, N. Farber, M. Link, and B. Girod. Analysis of video transmission over lossy channels. *IEEE J. Select. Areas Commun.*, June 2000.

[6] S. Tao and R. Guerin. Application-specific path switching: A case study for streaming video. In *Proc. of ACM Multimedia*, October 2004.

[7] S. Tao and R. Guerin. On-line estimation of Internet path performance: An application perspective. In *Proc. of IEEE INFOCOM*, March 2004.

[8] VQEG. Final report on the validation of objective models of video quality assessment. *http://www.vqeg.org/*, August 2003.