Quality monitoring of video over a packet network

Amy R. Reibman, AT&T Labs – Research, amy@research.att.com
Vinay Vaishampayan, AT&T Labs – Research, vinay@research.att.com
Yegnaswamy Sermadevi, Cornell University, swamy@ece.cornell.edu

***Abstract*— We consider monitoring the quality of compressed video transmitted over a packet network from the perspective of a network service provider. Our focus is on no-reference methods, which do not access the original signal, and on evaluating the impact of packet losses on quality. We present three methods to estimate Mean Squared Error (MSE) due to packet losses directly from the video bitstream. NoParse uses only network-level measurements (like packet loss rate), QuickParse extracts the spatio-temporal extent of the impact of the loss, and FullParse extracts sequence-specific information including spatio-temporal activity and the effects of error propagation. Our simulation results with MPEG-2 video subjected to Transport Packet losses illustrate the performance possible using the three methods.**

I. INTRODUCTION

Accurate measures of video quality are invaluable whether one is designing, deploying, or maintaining a system which transports video across a network. For example, a video quality metric can help to write system requirements, or to decide which vendor's encoder should be used. It can help tune a video encoding algorithm, so bits are allocated where they can improve quality the most. In system design, an accurate video quality metric can help determine which system techniques to use in a given situation and when to switch from one technique to another as system parameters change. Further, a video quality metric is useful to verify the continued correct operation of an existing system.

We consider monitoring the quality of compressed video transmitted over a packet network from the perspective of a network service provider. As such, we are primarily concerned with monitoring bitstreams produced by an embedded base of video encoders that have been already deployed. Our motivation is to provide a tool that can quickly and efficiently monitor the quality of video that is being transported across a network. We focus on measuring the impact of packet loss on the perceived video quality. Since the original video is not available for comparison, we focus on quality monitoring tools that operate on the bitstream.

In this paper, we use the mean squared error (MSE) as a rough measure of video quality. The methodology we apply is general for any motion-compensated video compression algorithm, although the specific application that we consider in this paper is the transport of MPEG-2 video when Transport Packets may be lost.

In Section II, we discuss different measurements that can be used to determine video quality, and discuss their relevance for a network monitoring application. In Section III, we describe a framework for understanding the impact of packet losses on the decoded video. Section IV expands this framework, presenting three different methods to estimate the Mean Squared Error (MSE) from the received video bitstream. The three methods differ in the amount of information extracted from the bitstream and thus differ in their accuracy and complexity. Our goal is to have a method that can predict the quality of individual videos by taking into account sequence-specific factors including spatio-temporal activity and the effects of error propagation, with low enough complexity that it can be easily applied to many different video streams being sent across the network. Section V compares the performance of the three methods, and Section VI concludes.

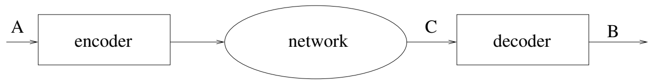

Fig. 1. Measurements for evaluating video quality

II. MEASURING THE QUALITY OF NETWORKED VIDEO

In a networked environment, a human's perception of video depends on both network-specific and network-independent factors. Network-specific factors include delay, delay variation, bit-rate, packet loss rate. The latter two are most important. Delay variation (or jitter) maps directly into lost packets, since if a packet is delayed too long the corresponding video will have been already displayed.

Perceived video quality depends on numerous network-independent factors, including the user and the environmental viewing conditions. Sequence-specific factors (length, content, amount of motion, amount of texture, and spatial and temporal resolution) play a distinct role in how visible the artifacts from network impairments are. A large error will be produced in regions with medium motion and a few edges, while a lost packet in a mostly still scene may produce no sizeable error.

Figure 1 illustrates measurements that could be used to characterize video quality. Measurement A corresponds to the uncompressed video available at the input to the encoder. Measurement B corresponds to the decompressed video available at the output of the decoder. Measurement C corresponds to the transmitted bitstream itself. Measurement C could be taken either at the input to the decoder or inside the network. The closer to the decoder that measurement C is taken, the more accurately it can characterize the video that will be displayed to the viewer. However, there may be system constraints or application requirements that force the measurement to be made in the network.

Video quality metrics can be categorized based on the measurements they use. Full-Reference (FR) methods require measurements at both A and B, while Reduced-Reference

(RR) methods require measurements at A and either B or C. Both FR and RR methods have received a lot of attention in the literature. (See for example [1], [2], [3] and the references therein.) FR methods use the entire set of pixels at both A and B, while RR methods extract key information from the encoder and decoder pixels and base their quality judgment on the extracted information. Both FR and RR require reliable transport of information to the video quality monitor from the transmitting and receiving ends. The rate of this transfer is very high for the FR methods, although they may have the greatest accuracy because they use the most information. However, the cost of maintaining a reliable alternate channel to a central facility may be prohibitive even for RR methods. Further, even RR methods are not possible if for any reason the original video is unavailable.

No-Reference (NR) methods do not require the original video, because they predict video quality using only measurements at either B or C. However, those methods that use measurements at B require a complete decoder for each video stream. If the goal is to measure the video quality of many streams in a network simultaneously, this can become prohibitive.

In what follows, we consider NR video quality measures that make use of measurements only at C, from the video bitstream itself. This approach has two advantages. First, it does not require a complete decoding for each video bitstream to be monitored and would require a lower bandwidth to transport data to a central monitoring facility. Therefore, this approach may be appropriate for measuring video quality within a network. Second, the location of a packet loss can be known explicitly from measurements at C.

The disadvantage of not using measurements at B is that the quality monitor must rely on assumptions regarding how the actual decoder behaves when presented with information that is known by external means to be corrupt, including what form of error concealment is used or whether the decoder discards corrupt information. If some knowledge is available regarding how the decoder deals with errors, this information can be incorporated into the error estimate.

In this paper, we estimate video quality using only the bitstream, at measurement point C. Further, we are interested in measuring only the impact of packet losses. As a result, we estimate the degradation between the network input (not labeled) and the network output (point C). Prior methods that consider measuring video quality using bitstream measurements [4], [5] have only considered the degradation between the encoder input (point A) and the encoder output (not labeled). Further research would be necessary to extend our systems to incorporate the ideas in [4], [5].

Specifically, [4] presents a method to evaluate video quality due to quantization noise only, based on measurements at point C for MPEG-2 video. They estimate a picture activity measure on a macroblock basis using the quantizer-scale value and the number of bits used to encode the DCT coefficients. Using the picture activity measure, the slope relating the PSNR and the logarithm of the quantizer-scale is obtained. Then, using the logarithm of the quantizer-scale and the obtained slope, the PSNR incurred by compression only is estimated.

Turaga et. al. [5] extend this work by estimating a separate statistical distribution for each DCT coefficient, using quantized data and assuming the distribution is constant across the frame. They use the estimated distribution to approximate the quantization error. They also do not consider packet losses, so the only error for I-frames is from quantization.

In this paper, we focus on measuring the quality of video when it is subjected to packet losses. We consider the case where we have access to the video bitstream and packet headers have been removed and any re-assembly done. Our quality monitors can be used in any network environment (IP, ATM, etc.). The basic methodology of the quality monitor can be applied to any motion-compensated compressed video bitstream, including MPEG-1, MPEG-2, and H.263.

We use the term MSE throughout this paper, although the output of the system could be more general and could include bounds on the maximum or minimum error. This may be particularly useful to characterize different expected errors depending on the concealment strategy of the decoder.

III. Impact of packet losses

In this section, we consider the impact of packet losses on the errors in the decoded video. We begin with some definitions. Specifically, let $\hat{f}(n,i)$ be the pixel value at the encoder at location i and time n reconstructed after encoding, and $\tilde{f}(n,i)$ the corresponding reconstructed pixel value at the decoder after possible losses. If there have been no losses, then $\hat{f}(n,i) = \tilde{f}(n,i)$. We consider the random variable $e(n,i) = \hat{f}(n,i) - \tilde{f}(n,i)$, which is the reconstruction error of pixel i in frame n at the decoder due to packet losses.

Let $\epsilon_p^2(n,i) = E[e(n,i)^2]$ be the MSE of the reconstruction error of the pixel i in frame n at the decoder due to packet losses. The subscript p denotes pixel granularity. We will also be interested in ϵ_{mb}^2, ϵ_{sl}^2, and ϵ_{seq}^2, which are the mean squared error at a macroblock, slice, and sequence level, respectively. To denote the estimates of MSE obtained by the quality monitors, we will add a "hat" to the relevant quantities.

In this section, we develop recursive equations to compute the error signal $e(n,i)$ exactly using three sets of information. The first is the set of all macroblocks that are missing from the bitstream, denoted $\mathcal{M}$. The second is the initial error for each pixel in the lost macroblocks, $e_0(n,i,t)$, which is introduced because of lost motion and difference information. This will be defined precisely below. The third is ψ, the set of macroblock type and motion information for every macroblock not in $\mathcal{M}$. We will refer to ψ as the prediction process, because it contains all necessary information to describe how the decoder should predict the current macroblock from previously decoded macroblocks. ψ governs how the initial error $e_0(n,i,t)$ from the missing macroblocks propagates spatially and temporally into other macroblocks.

In general, information from both the encoder and the decoder is necessary to compute the error signal for a particular set of packet losses. The encoder does not know exactly which packets have been lost ($\mathcal{M}$), and does not know the concealment strategy used by the decoder. However, the encoder does know ψ and the source (bitstream) information regarding the initial error $e_0(n,i,t)$. The decoder knows $\mathcal{M}$ and the concealment strategy, and also knows ψ. However, it does not know the contents of the lost packet and thus cannot know $e_0(n,i,t)$ accurately.

Recently, a number of methods have been presented which compute *at the encoder* the estimated distortion at the decoder when the video is sent across error-prone channels [6], [7], [8], [9]. The goal of these models is to optimally choose encoding parameters (for example, the intra-refresh rate of ψ) to obtain the best average video quality across the range of possible packet losses. These methods can make use of all information available to the encoder, but must rely on estimates of $\mathcal{M}$.

In this paper, we address the situation of using the bitstream to be received *by the decoder* to compute the MSE caused by lost packets. In section IV we present three methods to extract information from the received bitstream, so as to estimate the key parameters $\mathcal{M}$, $e_0(n,i,t)$, and ψ and thus estimate ϵ^2_{seq} for the actual packet loss in the received bitstream. The more information extracted from the bitstream, the more accurate the estimation process becomes.

In what follows, we show that the error signal $e(n,i)$ can be decomposed into two components: one due to the propagation of previous errors and the other the innovation due to new errors introduced by new losses. Next, we use these exact expressions to approximate the MSE due to packet loss, based only on information in the received bitstream. The FullParse and QuickParse methods are developed using these equations.

A. Exact recursive computation of error signal

We consider the reconstruction error at the decoder in a given frame due to any current or previously lost packets, at a pixel granularity. The impact of error propagation and multiple packet losses is naturally accounted for in this framework.

To describe the method, we assume full-pixel accuracy for the motion compensation and assume for notational convenience that there is one B-frame between intervening reference frames. However, our implementation does not rely on these assumptions. Therefore pixel i in a P-block in P-frame n is predicted from pixel j in frame $n-2$, and pixel i in a B-block in frame n is predicted from pixels j_{-1} and j_1 within frames $n-1$ and $n+1$, respectively.

We consider the error signal first for the case when existing errors are propagated, by considering those macroblocks not in $\mathcal{M}$ (ie, the received macroblocks). We use the information in ψ to consider the error signal propagation for each of the three macroblock types.

- For a received I-macroblock, the decoder can reconstruct the pixels exactly. Hence,

$$e(n,i) = \hat{f}(n,i) - \tilde{f}(n,i) = 0. \tag{1}$$

- For a received P-macroblock, the decoder adds the received difference signal $\hat{d}(n,i)$ to the motion-compensated prediction signal:

$$\tilde{f}(n,i) = \tilde{f}(n-2,j) + \hat{d}(n,i).$$

The error at the decoder can be written as

$$\begin{aligned} e(n,i) &= \hat{f}(n,i) - \tilde{f}(n,i) \\ &= (\hat{f}(n-2,j) + \hat{d}(n,i)) \\ &\quad - (\tilde{f}(n-2,j) + \hat{d}(n,i)) \\ &= \hat{f}(n-2,j) - \tilde{f}(n-2,j) \\ &= e(n-2,j). \end{aligned} \tag{2}$$

No new errors are introduced, but past errors propagate.

- When a B-macroblock is received, the decoder computes:

$$\tilde{f}(n,i) = (\tilde{f}(n-1,j_{-1}) + \tilde{f}(n+1,j_1))/2 + \hat{d}(n,i).$$

The error at the decoder can be expressed as

$$e(n,i) = (e(n-1,j_{-1}) + e(n+1,j_1))/2. \tag{3}$$

Again, no new errors are introduced, but errors from the reference frames propagate.

If the macroblock is in $\mathcal{M}$ (ie, it is not received), the decoder uses error concealment. For a lost macroblock in a reference frame (ie, I- or P-frame), we assume the decoder uses motion-compensated error concealment from the previous reference frame. For a lost macroblock in a B-frame, we assume the decoder uses the closest reference frame for concealment, in either the past or future, although this is easily generalized. Let $n-t$ be the index of the reference frame used for concealment. The pixel from frame $n-t$ used to estimate the current pixel i in frame n is denoted $\tilde{f}(n-t,k)$. If simple replacement from the previous frame is done, then $k=i$.

The decoder error can be written as

$$\begin{aligned} e(n,i) &= \hat{f}(n,i) - \tilde{f}(n-t,k) \\ &= (\hat{f}(n,i) - \hat{f}(n-t,k)) \\ &\quad + (\hat{f}(n-t,k) - \tilde{f}(n-t,k)). \end{aligned}$$

If we define

$$e_0(n,i,t) = \hat{f}(n,i) - \hat{f}(n-t,k), \tag{4}$$

then

$$e(n,i) = e_0(n,i,t) + e(n-t,k). \tag{5}$$

The second term in (5) is the propagation of errors from previous packet losses. The first term is an innovation term, characterizing the new error introduced by the lost packet.

The impact of multiple losses is captured by (5) even when consecutive frames are lost, because of the explicit dependence on t, the distance to the reference frame used for concealment.

The definition of $e_0(n,i,t)$ depends on the encoder's reconstructions and on the decoder's concealment strategy. The encoder can compute it given knowledge of the decoder's concealment strategy, but it can never be exactly computed at the decoder in the case of packet loss.

B. Mean squared error

While it is possible to recursively compute the error signal exactly, knowing $\mathcal{M}$, $e_0(n,i,t)$, and ψ, we must make two approximations to recursively compute MSE. Finding the MSE of I- and P-macroblocks using (1) and (2) is straightforward. To find the MSE for a B-macroblock, we must consider two cases: when only one of the terms in (3) is non-zero, and when both terms are non-zero. If reference frame $n - r$ has zero error at pixel j_{-r}, then the squared error is

$$\epsilon_p^2(n,i) = \epsilon_p^2(n+r, j_r)/4.$$

However, if both reference frames have an error, then we must consider the cross-term. We assume the cross-term contributes the same impact as the sum of each of the terms individually. In this case,

$$\epsilon_p^2(n,i) = (\epsilon_p^2(n-1, j_{-1}) + \epsilon_p^2(n+1, j_1))/2.$$

This was found to match the data well, and because this is for a B-block, any inaccuracy in this approximation will not propagate into other frames.

To find the MSE in the case of a lost macroblock, we must consider the correlation between the two terms in (5). The first term is the new error introduced by the lost packet, and the second term is the propagation of errors from previous packet losses. In general, these two terms are correlated [10]. However, they are nearly uncorrelated unless the previous loss affected the same pixel in the immediate reference frame. In the situation we consider for our simulation conditions, that of MPEG-2 video with losses of Transport Packets, this rarely happens. Therefore, we assume here they are uncorrelated, and express the expected squared error of (5) as

$$\epsilon_p^2(n,i) = \epsilon_{p0}^2(n,i,t) + \epsilon_p^2(n-t, k).$$

The first term can be estimated statistically from data, as described below. The second term is the effect of past error propagation.

So far, we have considered the MSE at a pixel granularity, ϵ_p^2. The coarser granularities of interest, ϵ_{mb}^2, ϵ_{sl}^2, and ϵ_{seq}^2, are simply the average of ϵ_p^2 over the range of interest.

IV. Video quality from bitstreams

We present three methods to estimate MSE from the video bitstream, which vary in their ability to predict sequence-specific quality and in their complexity and resulting ability to scale to measuring multiple streams in a network environment.

We choose these three methods because they lie at natural boundaries for how deeply to parse the bitstream. System constraints in a particular scenario may dictate how much bitstream processing is possible and therefore which method can be implemented. In the first method, which we call FullParse (FP), the only restriction is that a complete decoding is not possible; no IDCT or motion compensation of the decoded pixels takes place. In the second method, called QuickParse (QP), we use only high-level parsing. We search for start codes and decode the header immediately following them. However, we do not parse motion vectors, DCT coefficients, etc. In the third method, NoParse (NP), we do not examine the contents of the video bitstream. Instead we rely only on the number of packet losses and the average bit-rate.

Because of the depth at which they can parse the bitstream, the three methods differ in the accuracy with which they can estimate the three variables $\mathcal{M}$, $\epsilon_{p0}^2(n,i,t)$, and ψ. Thus, the three methods differ in the granularity with which it makes sense to estimate the error. FullParse keeps track of the error at the pixel granularity (ϵ_p^2) and estimates the initial error with macroblock granularity (ϵ_{mb}^2), QuickParse keeps track of and estimates the error at a slice granularity (ϵ_{sl}^2) and NoParse can only estimate the error at the sequence level (ϵ_{seq}^2).

A. *FullParse*

The first method we consider, FullParse, extracts as much information as possible from the video bitstream, *without* a complete decoding process (ie, no pixel reconstructions are ever formed). This monitor extracts sequence-specific information that affects video quality, including temporal and spatial resolution, slice length, and macroblock-specific information including motion, quantization parameters, macroblock type, and summary coefficient information.

We begin by describing the basic FullParse algorithm. This uses the iterative procedure of Section III-B above. FullParse also estimates seven spatio-temporal parameters from the received bitstream, which are used to help estimate the initial error $\hat{\epsilon}_{p0}^2(n,i,t)$ due to packet loss. These two estimation procedures are described in more detail after the basic algorithm.

1) The basic FullParse algorithm to recursively estimate MSE due to lost packets: FullParse extracts the lost macroblocks $\mathcal{M}$ and the prediction process ψ completely from the received bitstream with packet losses. FullParse also extracts other macroblock-specific information from the received bitstream, described in more detail below.

For each macroblock in the decoded video:

1) Determine if this macroblock is in $\mathcal{M}$ or not, and determine the prediction process ψ from the received video bitstream.
2) Extract any available macroblock-specific information from the received bitstream (as described in detail below), and estimate the seven spatio-temporal parameters required for equation (10) (as described in detail below).
3) If the macroblock is not in $\mathcal{M}$, compute propagation of any past errors using the macroblock type and motion information in ψ:

$$\hat{\epsilon}_p^2(n,i) = \begin{cases} 0 & \text{if (ia)} \\ \hat{\epsilon}_p^2(n-2, j) & \text{if (iia)} \\ \hat{\epsilon}_p^2(n-r, j_r)/4 & \text{if (iiia)} \\ \frac{1}{2}(\hat{\epsilon}_p^2(n-1, j_1) + \hat{\epsilon}_p^2(n+1, j_{-1})) & \text{if (iva).} \end{cases}$$

   Here, condition (ia) corresponds to an I-block, condition (iia) corresponds to a P-block, condition (iiia) corresponds to a B-block when only one reference frame has an error, and condition (iva) corresponds to a B-block when both reference frames have errors.
4) If the macroblock is in $\mathcal{M}$, estimate the initial error due to this macroblock being missing using (10) below, and compute the MSE for this macroblock:

$$\hat{\epsilon}_p^2(n,i) = \hat{\epsilon}_{p0}^2(n,i,t) + \hat{\epsilon}_p^2(n-t, k). \quad (6)$$

2) A model for the initial MSE of a missing macroblock: The initial error for each pixel when there is a loss, $\epsilon_{p0}^2(n,i,t)$, is difficult to extract from the bitstream. Its value depends heavily on the source content. The strategy used by the decoder for concealment also affects $\epsilon_{p0}^2(n,i,t)$. When the impact of losses is estimated at the encoder [6], [7], [8], [9], [10], the contribution of the source to this value can be readily incorporated. However, because we need to estimate $\epsilon^2_{p0}(n,i,t)$ using the received bitstream, an essential aspect of our work is to find accurate estimates of this parameter.

FullParse uses a statistical model to estimate $\epsilon^2_{p0}(n,i,t)$ for the missing macroblocks. In presenting the model, for notational simplicity we assume the decoder uses no motion-compensated concealment, or $k = i$. The more general case follows easily. The decoder uses pixel i in the closest reference frame (indicated by index $n-t$) to conceal the missing pixel i in frame n. If we define $\mu(n,i) = E\left[\hat{f}(n,i)\right]$, then we can write $\epsilon^2_{p0}(n,i,t)$ due to this macroblock being lost as

$$\begin{aligned}
&\epsilon^2_{p0}(n,i,t) \\
&\quad= E\Big[(\hat{f}(n,i) - \mu(n,i) + \mu(n,i) \\
&\qquad\quad - \hat{f}(n-t,i) + \mu(n-t,i) - \mu(n-t,i))^2\Big] \\
&\quad= E\left[(\hat{f}(n,i) - \mu(n,i))^2\right] \\
&\qquad +E\left[(\hat{f}(n-t,i) - \mu(n-t,i))^2\right] \\
&\qquad -2E\left[(\hat{f}(n,i) - \mu(n,i))(\hat{f}(n-t,i) - \mu(n-t,i))\right] \\
&\qquad +(\mu(n,i) - \mu(n-t,i))^2. \qquad (7)
\end{aligned}$$

If we let $\sigma^2_n = E\left[(\hat{f}(n,i) - \mu(n,i))^2\right]$, and assume that $\sigma^2_{n-t} \approx \sigma^2_n$, then we can write this as

$$\epsilon^2_{p0}(n,i,t) = 2\sigma^2_n(1 - \beta_{n,n-t}) + (\mu(n,i) - \mu(n-t,i))^2.$$

Here, we have defined the correlation between pixel i in the current frame and the same pixel in the reference frame to be

$$\beta_{n,n-t} = \frac{E\left[(\hat{f}(n,i) - \mu(n,i))(\hat{f}(n-t,i) - \mu(n-t,i))\right]}{\sigma^2_n}.$$

To compute the value of $\beta_{n,n-t}$ from information gathered from the bitstream, we assume the video is governed by a Gauss-Markov model, which is separable temporally, horizontally, and vertically. Let $\alpha_{n,n-t}$ be the temporal correlation between frames n and $n-t$ that remains *after* motion compensation. That is,

$$\alpha_{n,n-t} = \frac{E\left[(\hat{f}(n,i) - \mu(n,i))(\hat{f}(n-t,j) - \mu(n-t,j))\right]}{\sigma^2_n}.$$

Also, let the horizontal and vertical spatial correlations between adjacent pixels in the same frame be ρ_h and ρ_v respectively. If the motion vector is $j - i = u$ with components (u_h, u_v) then we can write the correlation between pixels j and i in the same frame as

$$\rho_{i,j} = \rho_h^{|u_h|}\rho_v^{|u_v|}.$$

Note that with these definitions, the correlation between pixels $f(n,i)$ and $f(n-t,m)$ will be greatest if $m = j$, and will decrease as $|j-m|$ increases.

With these definitions, then $\beta_{n,n-t} = \alpha_{n,n-t}\rho_{i,j}$, and the new mean squared error due to a lost packet at pixel i when the decoder uses zero-motion concealment becomes:

$$\begin{aligned}
&\epsilon^2_{p0}(n,i,t) \qquad (8) \\
&\quad= 2\sigma^2_n(1 - \alpha_{n,n-t}\,\rho_h^{|u_h|}\rho_v^{|u_v|}) + (\mu(n,i) - \mu(n-t,i))^2.
\end{aligned}$$

Now let us examine $\alpha_{n,n-t}$, the temporal correlation after motion compensation. If we consider the process of motion compensation at the encoder for a P-macroblock, we can express the energy of the difference signal that is sent to the decoder for a P-macroblock using the same process as (7).

$$\begin{aligned}
\sigma^2_d &= E\left[(\hat{d}(n,i))^2\right] = E\left[(\hat{f}(n,i) - \hat{f}(n-t,j))^2\right] \\
&= 2\sigma^2_n(1 - \alpha_{n,n-t}) + (\mu(n,i) - \mu(n-t,j))^2. \qquad (9)
\end{aligned}$$

Solving for $\alpha_{n,n-t}$ and substituting into (9), we have the initial expected squared error due to this macroblock being lost when the decoder uses zero-motion concealment:

$$\begin{aligned}
\epsilon^2_{p0}(n,i,t) &= \left[2\sigma^2_n + (\mu(n,i) - \mu(n-t,i))^2\right] \qquad (10) \\
&- (\rho_h^{|u_h|}\rho_v^{|u_v|})\left[2\sigma^2_n + (\mu(n,i) - \mu(n-t,j))^2 - \sigma^2_d\right].
\end{aligned}$$

We use (10) to estimate $\hat{\epsilon}^2_{p0}(n,i,t)$ in (6). All the parameters in (10) can be estimated from parameters extracted from the bitstream with variable length decoding and inverse quantization. Motion compensation, and the Inverse Discrete Cosine Transform (DCT) are not necessary. This estimation procedure is explained next.

3) Estimating spatio-temporal video parameters : We estimate the seven statistical parameters in (10) from the received bitstream on a macroblock basis: $\mu(n,i)$, σ^2_n, σ^2_d, ρ_v, ρ_h, u_v, and u_h. We assume these values are the same for all pixels in the macroblock.

In the current implementation, we keep an estimate of each of the seven parameters for each macroblock, received or not. For missing macroblocks, these parameters are estimated from their neighbors. For received macroblocks in I-frames, $\mu(n,i)$, σ^2_n, ρ_v, and ρ_h can be estimated from bitstream information, while σ^2_d, u_h, and u_v cannot. For received macroblocks in P- and B-frames, σ^2_d, u_h, and u_v can be easily estimated from the bitstream information, while σ^2_n, ρ_v, ρ_h, and $\mu(n,i)$ cannot. With the exception of $\mu(n,i)$, whenever the parameter cannot be estimated from the bitstream, we use the most recent estimate from that spatial location in the neighboring frames.

For a received I-macroblock, the pixel mean $\mu(n,i)$ can be recovered from the bitstream directly, using the DC component of the DCT. The spatial variance σ^2_n can also be extracted directly from the DCT coefficients. For received P- and B-macroblocks, the only way to exactly compute the macroblock mean is with a complete decode. Therefore, we obtain an estimate of $\mu(n,i)$ for these blocks by adding their DC component to a motion-compensated estimate of the mean of the appropriate reference frame.

For a received P- or B-macroblock, the parameter σ^2_d (which is a measure of the temporal energy of the macroblock) can be extracted directly using the received DCT coefficients. The motion components u_v, and u_h are extracted directly from the bitstream for P- and B-macroblocks.

The vertical and horizontal spatial correlations ρ_v, ρ_h, are estimated from the received I-block DCT coefficients. The horizontal spatial correlation is obtained from the DCT coefficients on the same horizontal row as the DC coefficient (denoted $D_{0,k}$) using a two-step process. First, we compute $\rho_{h,128}$, the horizontal spatial correlation assuming the pixels have mean 128:

$$\rho_{h,128} = \left[\sum_{k=0}^{8} C_{0,k} D_{0,k}^2 C_{k,1}\right] / \left[\sum_{k=0}^{8} C_{0,k} D_{0,k}^2 C_{k,0}\right]$$

where $C_{0,k}$ is the $(0,k)$-th element of the matrix comprising the DCT transform. Second, we compute the correlation ρ_h by accounting for the fact that the pixels actually have a mean of $\mu(n,i)$:

$$\rho_h = \rho_{h,128} - \frac{(\mu(n,i)-128)^2}{\sigma_n^2}(1-\rho_{h,128})$$

The vertical spatial correlation ρ_v can be estimated similarly.

B. QuickParse

The second method we present, QuickParse, extracts only high-level information from the video bitstream. Therefore, it can process a many bitstreams more quickly than the FullParse method. However, because it cannot extract macroblock-specific information, it is cannot be as accurate as FullParse.

QuickParse extracts only picture, slice, and Group of Block (GOB) start codes and the information in the corresponding headers, including the slice/GOB location and the slice quantizer. The start codes themselves are typically byte-aligned synchronization codes, so they are quickly and easily identified within the bitstream. Using this information, QuickParse can determine exactly which macroblocks are lost, $\mathcal{M}$. However, because it cannot access macroblock-specific information, it is unable to determine ψ exactly. In general, ψ can be decomposed into two components: the temporal duration and the spatial spread of an error. QuickParse can use the frame types that it extracts from the picture headers to obtain some information regarding the temporal duration of a loss. However, it is not able to extract information regarding the spatial spread of each loss.

The amount of spatial spread is governed by the motion vectors, which QuickParse cannot access. The spatial filtering incurred by motion compensation was modeled in [6] by a geometric attenuation factor that depends on the number of frames since the loss occurred. Because we would like an implementation that does not need to keep track of the time since each loss, we use a fixed attenuation factor for all reference frames, with no attenuation for B-frames. We choose $\gamma_\psi = 0.85$ to account for both I-blocks within P- or B-frames and the spatial spread. For the sequences we considered, this value appeared to match the data well.

The basic algorithms for QuickParse and FullParse are similar. However, QuickParse computes MSE for each slice, whereas FullParse computes MSE for each pixel. QuickParse can only use frame-type information to infer macroblock-type information, whereas FullParse can use actual macroblock-type information.

The basic QuickParse algorithm follows. For each slice in the decoded video:

1) Determine $\mathcal{M}$ and the frame type from the received video bitstream.
2) If the slice is not in $\mathcal{M}$, compute propagation of any past errors using the frame type and approximate error attenuation using γ_ψ:

$$\hat{\epsilon}_{sl}^2(n,s) = \begin{cases} 0 & \text{if (ib)} \\ \gamma_\psi\ \hat{\epsilon}_{sl}^2(n-2,s) & \text{if (iib)} \\ \hat{\epsilon}_{sl}^2(n-r,s)/4 & \text{if (iiib)} \\ \frac{1}{2}(\hat{\epsilon}_{sl}^2(n-1,s)+\hat{\epsilon}_{sl}^2(n+1,s)) & \text{if (ivb)} \end{cases}$$

Here, condition (ib) corresponds to an I-frame, condition (iib) corresponds to a P-frame, condition (iiib) corresponds to a B-frame when only one reference frame has an error, and condition (ivb) corresponds to a B-frame when both reference frames have errors.
3) If the slice is in $\mathcal{M}$, estimate the initial error due to this slice being missing and compute the MSE for this slice:

$$\hat{\epsilon}_{sl}^2(n,s) = \hat{\epsilon}_{sl0}^2(n,s,t) + \gamma_\psi\ \hat{\epsilon}_{sl}^2(n-t,s).$$

The index s corresponds to the slice number.

QuickParse has limited ability to extract information regarding the magnitude of the initial squared error, $\epsilon_{sl0}^2(n,s,t)$. Currently, we use training data to estimate this value, using only the frame type and the distance, t, to the closest reference frame. We train across multiple sequences, so this technique is not sequence-specific and produces the same initial MSE for all lost macroblocks in the same type frame with the same distance to a reference frame.

C. NoParse

The final method we consider, NoParse, cannot access any information regarding $\mathcal{M}$, ψ, or $\epsilon_{p0}^2(n,i,t)$ for each loss. NoParse estimates the sequence MSE based only on measurements of the packet loss rate (PLR) inside the network. A similar method was studied in [11], where analytic expressions were developed to show that the root mean square energy loss due to ATM cell losses in an MPEG-2 bitstream has a square-root relationship with PLR, or equivalently, that the sequence MSE is linear with PLR. The constant multiplier depends on video-specific factors [11] and decoder concealment [6].

There are three difficulties associated with this simple model. First, a single linear fit across multiple sequences will not be accurate, even when the sequences use identical encoding parameters, because of different amounts of source activity. Second, the linear fit will cease to be accurate when multiple losses affect adjacent frames [10]. Third, even for isolated losses and small PLR, the variance of the actual MSE for a single sequence can be quite large. Thus, simply measuring the PLR observed in the network may not accurately characterize the resulting quality for a specific sequence.

Despite these difficulties, there are some instances in which no measurements except bit-rate and PLR are available. Therefore, this method may still be a valuable tool for network monitoring of transported video. To estimate PLR in the network, we simply count the number of observed losses in the video bitstream.

V. Performance

A. Description of sequences

We use sixteen 10-second MPEG-2 video sequences, divided into four groups of four sequences each. All video has 720 pixels, 480 lines, and 60 fields per second. Each sequence

TABLE I

SLOPES OF LINEAR FIT FOR SIXTEEN SEQUENCES.

Group	1	2	3	4
D	6600	12000	4170	13400
E	7050	18700	10900	4000
F	8650	20200	20800	19100
G	6070	9540	2550	5210

TABLE II

WITHIN-SEQUENCE CORRELATIONS BETWEEN ϵ^2_{seq} AND $\hat{\epsilon}^2_{seq}$.

	Bound	FP	QP	NP
mean in test set	0.992	0.983	0.963	0.921
mean in train set	0.997	0.985	0.973	0.932

contains 299 frames. The sixteen sequences have nearly equal bit-rate, equivalent Group-of-Picture structures, but different motion characteristics. We use groups F and G as our training set for QuickParse and NoParse and use groups D and E as our test set. Recall that FullParse does not use training data.

A packet loss corresponds to the loss of an MPEG-2 Transport Packet (188 bytes), and the decoder uses zero-motion concealment. For each average PLR in the set $\{5 * 10^{-5}, 10^{-4}, 2 * 10^{-4}, 5 * 10^{-4}, 10^{-3}, 2 * 10^{-3}, 3 * 10^{-3}, 4 * 10^{-3}, 5 * 10^{-3}\}$, we inject 25 random loss patterns into each of the sixteen sequences, for a total of 3600 samples of ϵ^2_{seq}. We believe these average packet loss rates span the space of interest for MPEG-2 video with Transport Packet losses.

To characterize the sensitivity of loss of each of the sixteen sequences, Table I indicates for each sequence the slope relating PLR to sequence MSE as discussed in Section IV-C. Clearly, there is a wide range of activity in the sequences, with sequence F3 having the largest sensitivity to loss, and sequence G3 having the least. The sequences in the test set (groups D and E) have sensitivities somewhere in between. Applying a linear fit to all eight training sequences simultaneously, we have the equation relating sequence MSE to PLR for the NoParse quality metric: $\hat{\epsilon}^2_{seqNP} = 11500 * PLR$.

B. Performance comparison

Table II shows the mean of the within-sequence correlation coefficient between the estimated MSE and the actual MSE for each sequence and each method, averaged over the test set and the training set. Also shown is the performance of a bound[1], obtained when FullParse uses the *actual* initial error instead of the estimated initial error from (10). We see that, within a sequence, each method produces an estimated MSE that is highly correlated with the actual MSE, although FullParse performs best and NoParse is worst.

Table III shows the cross-sequence correlation coefficient between the estimated MSE and the actual MSE when 8 sequences are in each set. The performance of the test set is better than that of the training set because the training set has greater variability. The bound and FullParse are both able to estimate MSE well across multiple sequences, while the correlation for QuickParse and NoParse are much lower.

[1]The bound still does not compute MSE exactly, because it relies on the approximations in Section III-B.

TABLE III

CROSS-SEQUENCE CORRELATIONS BETWEEN ϵ^2_{seq} AND $\hat{\epsilon}^2_{seq}$.

	Bound	FP	QP	NP
across test set	0.991	0.969	0.810	0.774
across train set	0.998	0.948	0.789	0.714

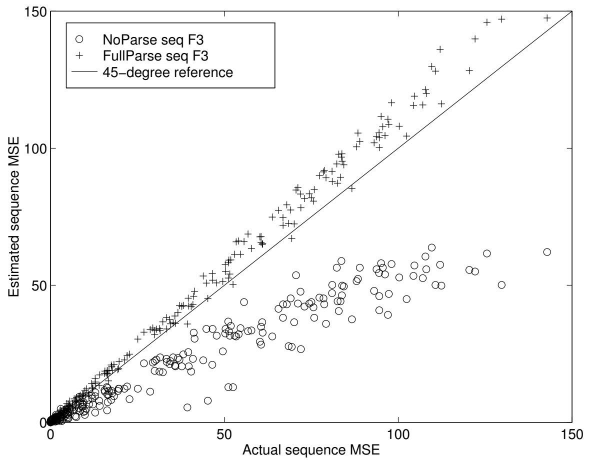

Fig. 2. Scatter plot for NoParse and FullParse, sequence F3.

However, the correlation coefficients do not clearly indicate why this performance trend occurs. Figures 2 and 3 show scatter plots of the estimated vs. actual MSE for sequence F3. Figure 2 shows results for both NoParse and FullParse, while Figure 3 shows results for both QuickParse and the bound. From these figures, we see a clear linear relationship between ϵ^2_{seq} and $\hat{\epsilon}^2_{seq}$ for each method, although the slope of the resulting fit may be far from one (which is the slope of an unbiased estimator). For this sequence, QuickParse has a much narrower spread than NoParse, although each has a slope around 1/2. The slope of the scatter for FullParse is 1.12, and the bound has a very tight scatter around a slope of one.

These ideas can be made more rigorous by using linear regression to fit the estimated MSE $\hat{\epsilon}^2_{seq}$ using the actual MSE ϵ^2_{seq}. Of interest is the regression coefficient b, (see (1.2.9) in

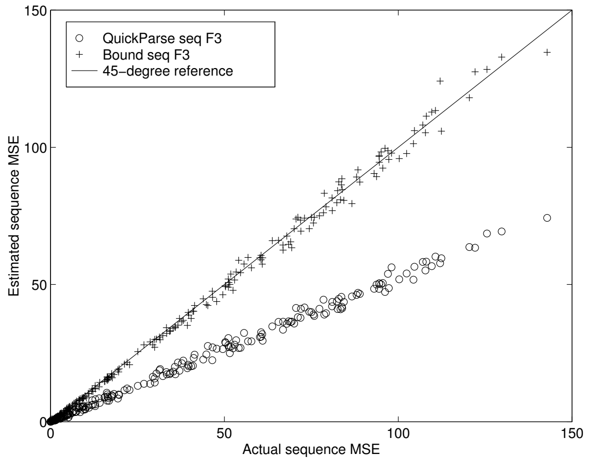

Fig. 3. Scatter plot for QuickParse and the bound, sequence F3.

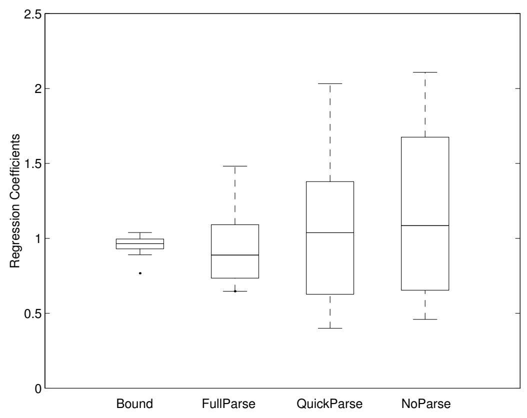

Fig. 4. Box-and-whisker plot of regression coefficients for 16 sequences.

[12]) which is the fitted slopes of the scatter plots. The within-sequence correlations summarized in Table II characterize the accuracy of the regression [12], while Figure 4 shows a box-and-whisker plot of these regression coefficients for each method and all 16 sequences, where the box contains the data within the middle two quartiles and the whiskers show the extent of the rest of the data.

The high within-sequence correlations in Table II indicate that a linear fit is accurate for all methods within each sequence, even though Figure 4 shows that the fitted slopes for QuickParse and NoParse are often far from one. The disparate slopes for different sequences lead to low cross-sequence correlations in Table III for QuickParse and NoParse. FullParse and the bound have fitted slopes much closer to one, which explains their larger cross-sequence correlations.

Results illustrating the ability of each of the methods to accurately produce an alarm when the estimated sequence MSE $\hat{\epsilon}^2_{seq}$ exceeds a threshold, T can be found in [13].

VI. Conclusions

We considered the problem of monitoring video quality in a lossy packet network using received bitstream measurements alone. Three methods for estimating the MSE were presented that span the range of trade-offs between complexity and accuracy. FullParse, the most accurate method, extracts detailed local information regarding the impact of the packet loss. QuickParse is able to identify the spatial extent and temporal duration of each loss, although its current implementation does not obtain an accurate estimate of the initial error of each loss. NoParse, the least accurate of our methods, treats all losses identically and only estimates the MSE on the same time scale as the PLR. The bound provides a benchmark that shows the FullParse method could be improved in future work.

References

[1] S. Hemami and M. Masry, "Perceived quality metrics for low bit rate compressed video", *Int. Conf. on Image Proc. (ICIP)*, pp. 721–724, Sept. 2002.

[2] H. R. Wu, T. Chen, S. Winkler, and Z. Yu, "Vision-model-based impairment metric to evaluate blocking artifacts in digital video", *Proceedings of the IEEE*, vol. 90, no. 1, pp. 154 –169, Jan. 2002.

[3] S. Wolf and M. Pinson, "In-service performance metrics for MPEG-2 video systems", IAB, Montreux, Switzerland, Nov 12-13, 1998.

[4] M. Knee, "The picture appraisal rating (PAR) – a single-ended picture quality measure for MPEG-2", *Int. Broadcast Convention*, 2000. Also available at http://www.snellwilcox.com/internet/reference/pdfs/parpaper.pdf

[5] D. Turaga et. al. "No reference PSNR estimation for compressed pictures", in *Int. Conf. on Image Proc. (ICIP)*, pp. III.61–III.64, Sept. 2002.

[6] K. Stühlmuller et. al. "Analysis of video transmission over lossy channels", *IEEE J. on Selected Areas in Comm.*, vol. 18, no. 6, pp. 1012-1032, June 2000.

[7] Z. He, J. Cai, and C. W. Chen, "Analytic end-to-end rate distortion modeling and control for packet video over wireless network", *International Workshop on Packet Video*, Pittsburgh, PA, April 2002.

[8] R. Zhang, S. L. Regunathan, and K. Rose, "Video coding with optimal Inter/Intra-mode switching for packet loss resilience", *IEEE J. on Selected Areas in Comm.*, vol. 18, no. 6, pp. 966-976, June 2000.

[9] D. Wu, Y. T. Hou, B. Li, W. Zhu, Y.-Q. Zhang, and H. J. Chao, "An end-to-end approach for optimal mode selection in Internet video communication: Theory and application", *IEEE J. on Selected Areas in Comm.*, vol. 18, no. 6, pp. 977-995, June 2000.

[10] Y. J. Liang, J. G. Apostolopoulos, B. Girod, "Analysis of packet loss for compressed video: does burst-length matter?", *ICASSP '03*, Hong Kong, April 2003.

[11] C. W. Snyder, U. K. Sarkar, and D. Sarkar, "Effects of cell loss on MPEG video: analytic modeling and empirical validation", *Int. Conf. on Multimedia and Expo*, Lausanne, Switzerland, pp. 457–460, August 2002.

[12] N. R. Draper and H. Smith, *Applied Regression Analysis*, John Wiley and Sons, New York, 1981.

[13] A. R. Reibman, Y. Sermadevi, and V. Vaishampayan, "Quality monitoring of video over the Internet", Asilomar Conference on Signals, Systems, and Computers, Nov. 3–6, 2002.