EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

Real-time video quality monitoring

Tao Liu*, Niranjan Narvekar, Beibei Wang, Ran Ding, Dekun Zou, Glenn Cash, Sitaram Bhagavathy and Jeffrey Bloom

Abstract

The ITU-T Recommendation G.1070 is a standardized opinion model for video telephony applications that uses video bitrate, frame rate, and packet-loss rate to measure the video quality. However, this model was original designed as an offline quality planning tool. It cannot be directly used for quality monitoring since the above three input parameters are not readily available within a network or at the decoder. And there is a great room for the performance improvement of this quality metric. In this article, we present a real-time video quality monitoring solution based on this Recommendation. We first propose a scheme to efficiently estimate the three parameters from video bitstreams, so that it can be used as a real-time video quality monitoring tool. Furthermore, an enhanced algorithm based on the G.1070 model that provides more accurate quality prediction is proposed. Finally, to use this metric in real-world applications, we present an example emerging application of real-time quality measurement to the management of transmitted videos, especially those delivered to mobile devices.

Keywords: G.1070, video quality monitoring, bitrate estimation, frame rate estimation, packet-loss rate estimation

1 Introduction

With the increase in the volume of video content processed and transmitted over communication networks, the variety of video applications and services has also been steadily growing. These include more mature services such as broadcast television, pay-per-view, and video on demand, as well as newer models for delivery of video over the internet to computers and over telephone systems to mobile devices such as smart phones. Niche markets for very high quality video for telepresence are emerging as are more moderate quality channels for video conferencing. Hence, an accurate, and in many cases real-time, assessment of the video quality is becoming increasingly important.

The most commonly used methods for assessing visual quality are designed to predict subjective quality ratings on a set of training data [1]. Many of these methods rely on access to an original undistorted version of the video under test. There has been significant progress in the development of such tools. However, they are not directly useful for many of the new video applications and services in which the quality of a target video must be assessed without access to a reference. For these cases, no-reference (NR) models are more appropriate. Development of NR visual quality metrics is a challenging research problem partially due to the fact that the artifacts introduced by different transmission components can have dramatically different visual impacts and the perceived quality can largely depend on the underlying video content. Therefore, a "divide-and-conquer" approach is often adopted. Different models are designed to detect and measure specific artifacts or impairments [2]. Among various forms of artifacts, the most commonly studied are spatial coding artifacts, e.g. blurriness [3-5] and blockiness [6-9], temporally induced artifacts [10-12], and packet-loss-related artifacts [13-18]. In addition to the models developed for specific distortions, there are investigations into generic quality measurement which can predict the quality of video affected by multiple distortions [19]. Recently, there are numerous efforts on developing QoS-based video quality metrics, which can be easily deployed in network environment. International Telecommunication Unit (ITU) and Video Quality Expert Group (VQEG) proposed the concepts of non-intrusive parametric and bitstream quality modeling, P. NAMS and P.NBAMS [20]. Based on the investigation of the relationship between video quality and bitrate and quantization parameter (QP) [21], Yang et al. proposed a quality metric by considering various bitstream domain features, such as bit rate,

* Correspondence: tao.liu@dialogic.com
Dialogic Inc., 12 Christopher Way, Suite 104, Eatontown, NJ 07724, USA

QP, packet loss and error propagation, temporal effects, picture type, etc. [22]. Among others, the multimedia quality model which is standardized by ITU-T in its Recommendation G.1070 in 2007 [23] is a widely used NR quality measure.

In ITU-T Recommendation G.1070, a framework for assessing multimedia quality is proposed. It consists of three models: a video quality estimation model, a speech quality estimation model, and a multimedia quality integration model. The video quality estimation model (which we will loosely refer to as the G.1070 model in this article) uses the bit rate (bits per second) and frame rate (frame per second) of the compressed video, along with the expected packet-loss rate (PLR) of the channel, to predict the perceived video quality subject to compression artifacts and transmission error artifacts. Details of the G.1070 models, including equations, can be found in [23]. Since its standardization, the G.1070 model has been widely used, studied, extended, and enhanced. Yamagishi and Hayashi [24] proposed to use G.1070 in the context of IPTV quality. Since the G.1070 model is codec dependent, Belmudez and Moller [25] extended the model, originally trained for H.264 and MPEG4 video, to MPEG-2 content. Joskowicz and Ardao [26] enhanced G.1070 with both resolution- and content-adaptive parameters.

In this article, we showcase how this technology can be used in a real-world video quality monitoring application. To accomplish this, there are several technical challenges to overcome. First of all, G.1070 was originally designed for network planning purposes, and it cannot be readily used within a network or at a video player for the purpose of real-time video quality monitoring. This is because the three inputs to the G.1070 model, i.e. bitrate, frame rate, and PLR of the encoded video bitstream, are not immediately available, and hence they need to be estimated from the bitstream. However, the estimation of these parameters is not straightforward. In this article, we propose efficient estimation methods that allow G.1070 to be extended from a planning tool to a real-time video quality monitoring tool. Specifically, we describe methods for real-time estimation of these three quality-related parameters in a typical video streaming environment.

Second, although the G.1070 model is generally suitable for estimating the quality of video conferencing content, where head-and-shoulder videos dominate, it is observed that its ability to account for the impact of content characteristics on video quality is limited. This is because the video compression performance is largely content dependent. For example, a video scene with a complex background and a high level of motion, and another scene with relatively less activity or texture, may have dramatically different perceived qualities even if they are encoded at the same bitrate and frame rate. To address this issue, we propose an enhancement to the G.1070 model wherein the encoding bitrate is normalized by a video complexity factor to compensate for the impact of content complexity on video encoding. The resulting normalized bitrate better reflects the perceptual quality of the video.

Based on the above contributions, this article also proposes a design for a realtime video quality monitoring system that can be used to solve real-world quality management problems. The ability to remotely monitor in real-time the quality of transmitted content (particularly to mobile devices) enables the right decisions to be made at the transmission end (e.g. by increasing the encoding bitrate or frame rate) in order to improve the quality of the subsequently transmitted content.

This article is organized as follows. In Section 2, the G.1070 video quality model is first introduced as a video quality planning tool, and then a scheme is proposed to extend it for video quality monitoring by estimating the three parameters, i.e. bitrate, frame rate, and PLR, from video bitstreams. In Section 3, we further propose an improved version of the G.1070 model to more accurately predict the quality of videos with different content characteristics. Experimental results demonstrating the proposed improvements are shown in Section 4. Using the proposed video quality monitoring tools, we present an emerging video application to measure and manage the quality of videos delivered to mobile phones in Section 5. Finally, Section 6 concludes this article.

2 Extension of G.1070 to video quality monitoring

In this section, G.1070 is first introduced as a planning tool. Then, we propose the estimation methods for bitrate, frame rate, and PLR, which allow G.1070 to be extended from a planning tool to a real-time video quality monitoring tool [27]. Specifically, we describe methods for real-time estimation of bitrate, frame rate, and PLR of an encoded video bitstream in a typical video streaming environment. Some of the practical issues therein are discussed. Based on simulation results, we also analyze the performance of the proposed parameter estimation methods.

2.1 Introduction of G.1070 as a planning tool

The ITU-T Recommendation G.1070 is an opinion model for video telephony applications. It proposes a quality measuring algorithm for QoE/QoS planning. The framework of the G.1070 model consists of three functions: video quality estimation, speech quality estimation, and multimedia quality integration. The focus of this article is on the video quality estimation model, which estimates perceived video quality (V_q) as a

function of bitrate, frame rate, and PLR, according to the following equations:

$$V_q = 1 + I_{\text{coding}} \exp\left(-\frac{P_{Pl_v}}{D_{PplV}}\right) \quad (1)$$

$$I_{\text{coding}} = I_{Ofr} \exp\left(-\frac{(\ln(Fr_V) - \ln(O_{fr}))^2}{2D_{FrV}^2}\right) \quad (2)$$

$$O_{fr} = v_1 + v_2 Br_V, \quad 1 \le O_{fr} \le 30 \quad (3)$$

$$I_{Ofr} = v_3 - \frac{v_3}{1 + \frac{Br_V}{v_4}^{v_5}}, \quad 0 \le I_{Ofr} \le 4 \quad (4)$$

$$D_{FrV} = v_6 + v_7 Br_V, \quad 0 \le D_{FrV} \quad (5)$$

$$D_{PplV} = v_{10} + v_{11} \exp\left(-\frac{Fr_V}{v_8}\right) + v_{12} \exp\left(-\frac{Br_V}{v_9}\right), \quad 0 \le D_{PplV} \quad (6)$$

where V_q is the video quality score, in the range from 1 to 5 (5 represents the highest quality). Br_v, Fr_v, and P_{Pl_v} represent bit rate, frame rate, and PLR, respectively. I_{coding} represents the quality of video compression, which is followed by the quality degradation caused by packet losses, a function of PLR and packet-loss robustness, D_{Pplv}. The model assumes that there is an optimal quality that can be achieved, I_{Ofr}, with given bitrate. The associated frame rate to optimal quality is denoted as O_{fr}. D_{FrV} is the robustness to quality change due to frame rate change.

v_1, v_2, . . ., and v_{12} are the 12 constants to be determined. These parameters are codec/implementation and resolution dependent. Although in the G.1070 Recommendation parameter sets are provided for H.264 and MPEG-4 videos at a few resolutions, the values of these parameters for other codecs and resolutions need to be determined. Refer to the Recommendation for more detailed interpretation of this model.

The intended application of G.1070 is QoE/QoS *planning*: different quality scores could be predicted by inputting different ranges of the three video parameters. Based on this, QoE/QoS planners can choose proper sets of video parameters to deliver a satisfactory service. G.1070 has the advantage of being simple and lightweight, in addition to being a NR quality model. These features make it ideal to be extended as a video quality monitoring tool. However, in a monitoring application, bit rate, frame rate, and PLR are usually not available to the network provider and end user. These input parameters to G.1070 need to be estimated from the received video bitstreams.

2.2 G.1070 extension to quality monitoring

In order to use G.1070 in a real-time video quality monitoring application, the essence and difficulty lies in effectively and robustly estimating the relevant parameters from encoded video data in network packets. Toward this goal, we propose a sliding window-based parameter estimation process, followed by a quality estimation using the G.1070 model, as shown in Figure 1. The input to the parameter estimation process is an encoded bitstream, packetized using any of the standard packetization formats, such as RTP, MPEG2-TS, etc. Note that in event of packet loss, it is assumed no retransmission is permitted. The parameter estimation process consists of three modules, i.e. feature extractor, feature integrator, and parameter estimator, and the function of this process is to estimate bit rate, frame rate, and PLR from the received bitstream in real-time. These parameters are then used by the G.1070 video quality estimation function [23]. The components of the proposed parameter estimation process are described below.

2.2.1 Feature extractor

The function of the feature extractor is to extract the desired features or data from video bistreams encapsulated in each network packet. Table 1 summarizes the outputs of this module.

2.2.2 Feature integrator

In order to estimate the bit rate, frame rate, and PLR, the feature integrator accumulates statistics collected by the feature extractor over a N-frame sliding window. Table 2 summarizes the outputs of this module.

The estimates of *timeIncrement, bitsReceivedCount*, and *packetsPerPicture* are prone to error due to packet loss. Therefore, extra care is taken while calculating these estimates including compensation for errors. The *bitsReceivedCount* is the basis for the calculation of bit rate, which may be underestimated due to possible packet loss. Thus, it is necessary to perform some compensation during the calculation of bit rate, which will be explained later. However, as will be explained below, the estimation of *timeIncrement* and *packetsPerPicture* are performed such that they are robust to packet loss.

The estimation of the *timeIncrement* between the frames in display order is complicated by the fact that almost all state-of-the-art encoding standards use a highly predictive structure. Because of this, the coding order is not the same as the display order and hence the received timestamps are not monotonically increasing. Also, packet losses can lead to frame losses which can cause missing timestamps. In order to overcome these issues, the *timeIncrement* estimator buffers timestamps over N frames and sorts them in ascending order. The *timeIncrement* is then estimated as the minimum difference between consecutive timestamps in the buffer. The

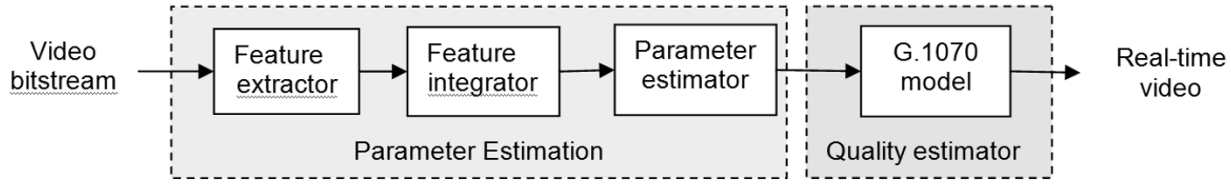


Figure 1 A system for video quality monitoring using the estimated quality parameters.

sorting makes sure that the timestamps are monotonically increasing and calculating the minimum timestamp difference makes the estimation more robust to frame loss. The effectiveness of this method is clear from experimental results on frame rate estimation in the presence of packet loss (Section 4.1.2), since *timeIncrement* is used to estimate the frame rate.

A *packetsPerPicture* estimate is calculated for each picture. For those frames that are affected by packet loss, the corresponding packetsPerPicture estimates are discarded since these may be erroneous.

2.2.3 Parameter estimator

At this point, the feature integrator module has collected all the necessary information for calculating the input parameters of the G.1070 video quality estimation model. The calculation of the input parameters is performed in the three sub-components of the parameter estimator as shown in Figure 2.

The packet-loss rate (*PLR*) estimator takes the *packetReceivedCount* and the *packetLossCount* as inputs and calculates the *P LR* as follows:

$$PLR = \frac{packetsLostCount}{packetsLostCount + packetsReceivedCount} \quad (7)$$

The frame rate (*FR*) estimator takes the *timeIncrement* and *timescale* as inputs and calculates the *FR* as follows:

$$FR = \frac{timeScale}{timeIncrement} \quad (8)$$

The bit rate (*BR*) is estimated from the *bitsReceivedCount*, the *packetsPerPic-ture*, the estimated PLR, and the estimated *FR*. In order to make the calculation of *BR* robust to packet loss, this calculation varies based on the estimated number of packets per picture. When each frame is transmitted in a single packet, i.e. *packetsPerPicture* = 1, no correction factor is needed and the *BR* is calculated as follows:

$$BR = FR \times \frac{bitsReceivedCount}{N}, \quad \text{packetsPerPicture} = 1 \quad (9)$$

However, if a frame is broken into multiple packets, i.e. *packetsPerPicture* > 1, it is likely that only partial frame information can be received when packet loss happens. Therefore, to compensate this impact on the calculation of bitrate, a normalization factor of the percentage of packets received is applied, as shown below:

$$BR = FR \times \frac{bitsReceivedCount}{N \times (1 - PLR)}, \quad \text{packetsPerPicture} > 1 \quad (10)$$

Finally, the *BR*, *FR*, and PLR estimates are provided to a standard G.1070 video quality estimator which calculates the corresponding video quality. Note that the parameters are estimated over a window of *N* frames. This means that the quality estimate at a frame is obtained from the statistics of the *N* preceding frames. The proposed system generates a video quality estimate for each frame, except during the initial buffering of *N* frames. No quality measurement is generated for lost frames.

2.3 Experimental results

The performance of the proposed video parameter estimation methods are validated by experimental results in Section 4. The proposed methods were implemented in

Table 1 Outputs of the feature extractor

Output feature (per packet)	Description
timeScale	The reference clock frequency of the transport format. For example, if we consider the transport of video over RTP, the standard clock frequency is 90 kHz.
timeStamp	Display time of the frame to which the packet belongs.
bitCount	The number of bits in the packet.
codedUnitType	Type of data in the packet. For example, in the case of H.264, the coded unit type corresponds to the NAL-unit type.
sequenceNumber	The sequence number of the input packet.

Table 2 Outputs of feature integrator

Output feature (per window)	Description
timeScale	Same as described in Table 1.
timeIncrement	The time interval between two adjacent video frames in display order.
bitsReceivedCount	The number of video coding layer bits received over the *N*-frame window. The determination of whether the bits belong to the video coding layer is based on the input *codedUnitType*. For example, in H.264, the SPS and PPS NAL-units do not belong to video coding layer and hence are not included in the calculation.
packetReceivedCount	The number of packets received over the *N*-frame window.
packetLostCount	The number of packets lost over the *N*-frame window. This can be determined by counting the discontinuities in the sequence number information.
packetsPerPicture	The number of video coding layer packets per picture.

a prototype system as a proof-of-concept and several experiments were performed with regard to the estimation accuracy of bit rate, frame rate, and PLR using a variety of bitstreams with different coding configurations. The experimental results in Section 4 show not only a high accuracy of estimation but also high robustness of the bit rate and frame rate estimation in the presence of packet loss.

3 Enhanced content-adaptive G.1070

The G.1070 model is originally designed for estimating the quality of video conferencing content, i.e. head-shoulder shots with limited motion. While this model provides reasonable quality prediction for such content, its correlation with the perceptual quality of video content with a wide range of characteristics is questionable. For example, it is generally "easier" for a video encoder to compress a simple static scene than a complex scene with plenty of motion. In other words, using similar bit rates (at the same frame rate without packet loss), simpler scenes can be compressed at a higher quality level than complex scenes. However, the G.1070 model, which considers only bit rate, frame rate, and PLR, will output similar quality estimates in this case. Figure 3 shows one such example wherein different CIF-resolution video scenes are encoded at a similar bit rate 128 kps and frame rate 30 fps (with no packet loss). We can see that G.1070 shows little variation since the input parameters of the scenes are similar (instantaneous bitrate can vary slightly depending on the bit rate control algorithm used). As a widely accepted reduced-reference *pixel-domain* video quality measure, NTIA-VQM [28], used as an estimate of mean opinion score (MOS) here, shows a significant quality variation to account for the changes in content characteristics. Another example in which G.1070 does not correlate

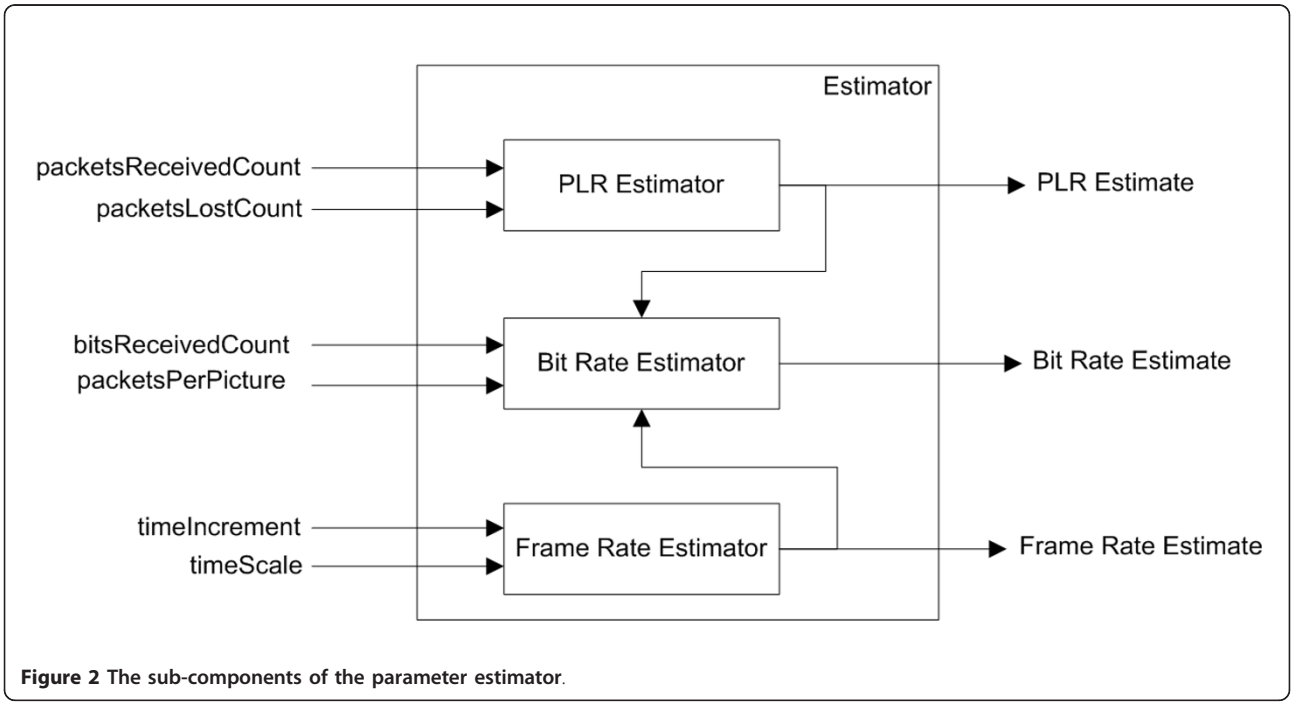


Figure 2 The sub-components of the parameter estimator.

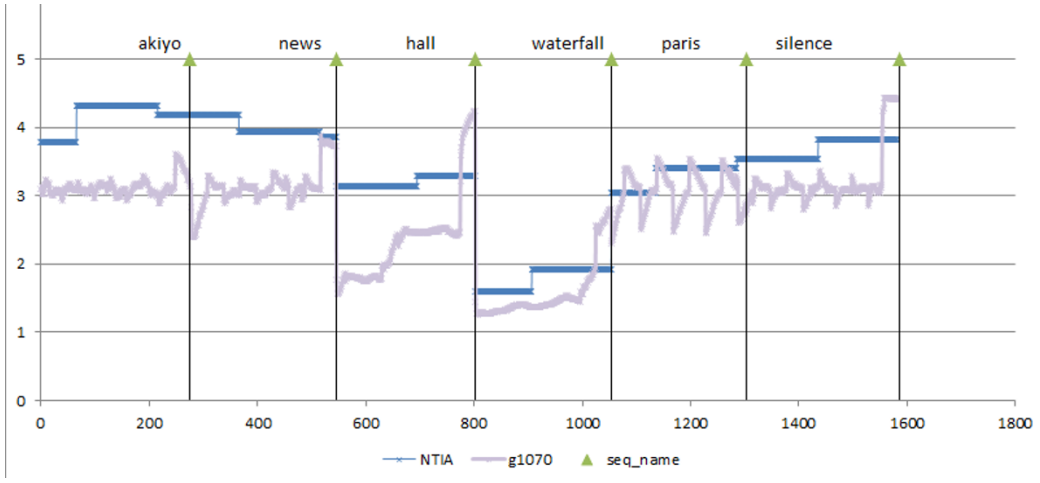


Figure 3 G.1070 quality prediction for video scenes with varying content characteristics.

with perceived video quality is when video bitstreams are encoded with different bit rate control algorithms, even if the bit rate budget is similar.

To address this issue, we propose a modified G.1070 model [29] that takes into consideration both the frame complexity and the encoder's bit allocation behavior. Specifically, we propose an algorithm that normalizes the estimated bit rate by the video scene complexity estimated from the bitstream. Figure 4 illustrates this enhanced G.1070 system (henceforth referred to as "G.1070E"). For a given frame of the input bitstream, the Parameter Estimation module computes the bit rate, frame rate, and PLR as shown in Figures 1 and 2. Additionally, in G.1070E, this module also extracts the quantization stepsize matrix, the number of coded macroblocks, and the number of coded bits for this frame. This information is used by the Frame complexity Estimator which computes an estimate of the frame complexity, as described in the next section. The frame complexity estimate is then used by the Bitrate Normalizer to normalize the bit rate. Finally, the frame rate estimate and PLR estimate from the Parameter Estimation

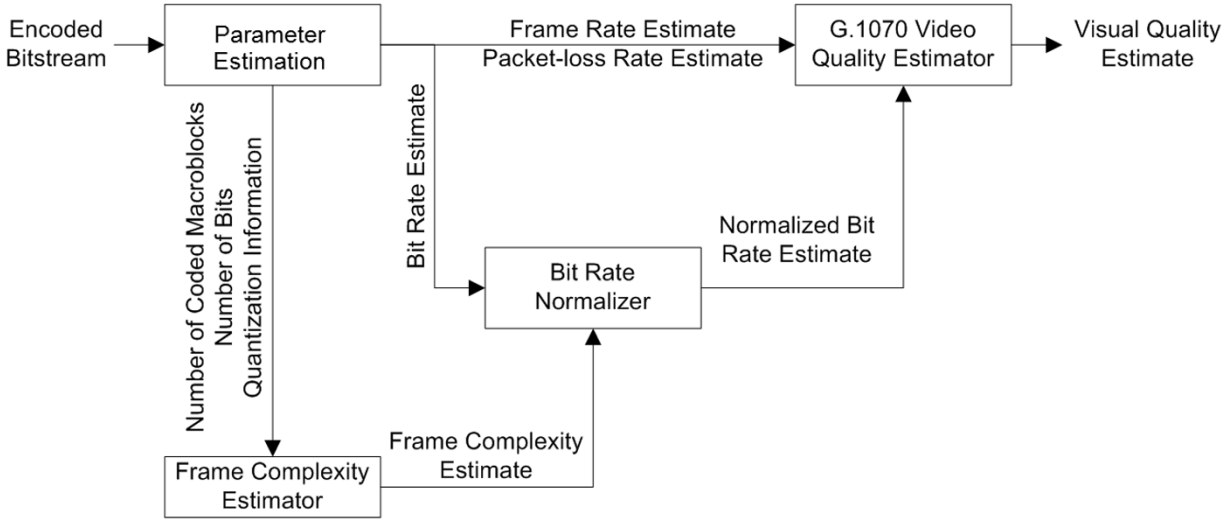


Figure 4 An extension of the G.1070 video quality model to include bit rate normalization based on an analysis of frame complexity.

module as well as the normalized bitrate from the Bitrate Normalizer are used by the G.1070 Video Quality Estimator to yield the video quality estimate.

3.1 Generalized frame complexity estimation

The complexity of a frame is a combination of the spatial complexity of the picture and the temporal complexity of the scene in which it is found. Pictures with more detail have higher spatial complexity than those with little detail. Scenes with high motion have higher temporal complexity than those with little or no motion. Compared to the previous works which investigate the frame complexity in the pixel domain [30,31], we proposed a novel frame complexity algorithm in the bitstream domain, which does not need to fully decode and reconstruct the videos and has much lower computational complexity. In a general video compression process, for a fixed level of quantization, frames with a higher complexity yield more bits. Similarly, for a fixed target number of bits, frames with higher complexity result in larger quantization step sizes. Therefore, the coding complexity can be estimated based on the number of coded bits and the level of quantization. These two parameters are used to estimate the number of bits that would have been used at a particular quantization level (denoted as *reference* quantization level), which is then used to predict complexity. The following derivation applies to many video compression standards including MPEG-2, MPEG-4, and H.264/AVC.

Let us refer to the matrix of actual quantization step sizes as M_{Q_input} and the matrix of reference quantization step sizes as M_{Q_ref}. Here, *Q_input* and *Q_ref* refer to some *quantization index* used to set the quantization step sizes, e.g. H.264 calls this the QP. For a given frame, the number of bits that would have been used at the reference quantization level, denoted by *bits* (M_{Q_ref}), can be estimated by the actual bits used to encode this frame, denoted by $bits(M_{Q\_input})$, and the two quantization matrices as shown in Equation 11. Under a packet-loss environment, *bits* (M_{Q_input}) is the actual bits which have been received for that frame. The quantization step size matrices M are either 8 × 8 or 4 × 4 depending on the specific video compression standard. Thus, each quantization step size matrix has either 64 or 16 entries. In Equation 11, the number of entries in the quantization step size matrix is denoted by N:

$$bits(M_{Q\_ref}) \approx \frac{\sum_{i=0}^{N-1} a_i \times m_{Q\_input\_i}}{\sum_{i=0}^{N-1} a_i \times m_{Q\_ref\_i}} \times bits(M_{Q\_input}) \quad (11)$$

The reference quantization step size matrix M_Q is arranged in zigzag order and m_Q is an entry in the matrix. To evaluate the effects of the quantization step size matrix, we consider a weighted sum of all the elements m_Q where the averaging factor, a, for each element depends on the corresponding frequency. In natural imagery, the energy tends to be concentrated in the lower frequencies. Thus, quantization step sizes in the lower frequencies have more impact on the resulting number of bits. The weighted sums in Equation 11 allow the lower frequencies to be weighted more heavily than the higher frequencies.

In many cases, different macroblocks can have different quantization step size matrices. Thus, the matrices specified in Equation 11 are averaged over all the macroblocks in the frame. Some compression standards allow macroblocks to be skipped. This usually occurs when the macroblock data can be well predicted from previously coded data. Hence, to be more specific, the quantization step size matrices specified in Equation 11 are averaged over all the coded (not skipped) macroblocks in the frame. To extract the QP and MB mode for each MB, the variable length decoding is needed, which is about 40% cycle complexity of the full decoding. Compared to the header only decoding, which is about 2-4% cycle complexity in the decoding progress, the proposed algorithm pays higher computational complexity to get more accurate quality estimation. However, compared with the video quality assessments in the pixel domain, our model has much lower complexity.

Equation 11 can be simplified by considering only binary averaging factors, a. The average factors associated with low frequency coefficients are assigned a value of 1 and the average factors associated with high frequency coefficients are assigned a value of 0. Since the coefficients are stored in zig zag order, which is roughly ordered from low frequency to high, Equation 11 can be rewritten as Equation 12:

$$bits(M_{Q\_ref}) \approx \frac{\sum_{i=0}^{K-1} m_{Q\_input\_i}}{\sum_{i=0}^{K-1} m_{Q\_ref\_i}} \times bits(M_{Q\_input}) \quad (12)$$

We have found that for matrices that are 8 × 8, the first 16 entries represent low frequencies and thus we set K = 16. For 4 × 4 matrices, the first 8 entries represent low frequencies and thus we set K = 8. If we define a *quantization complexity factor, fn* (M_{Q_input}), as

$$fn(M_{Q\_input}) = \frac{\sum_{i=0}^{K-1} m_{Q\_input\_i}}{\sum_{i=0}^{K-1} m_{Q\_ref\_i}}, \quad (13)$$

then Equation 12 can be rewritten as

$$bits(M_{Q\_ref}) \approx fn(M_{Q\_input}) \times bits(M_{Q\_input}) \quad (14)$$

Finally, in order to derive a measure of frame complexity that is resolution independent, we normalize the estimate of the number of bits necessary at the reference

quantization level by the number of 16 × 16 macroblocks in the frame (*frame_num_MB*). This gives the hypothetical number of bits per macroblock at the reference quantization level:

$$\begin{aligned} frame\_compexity &= \frac{bits(M_{Q\_ref})}{frame\_num\_MB} \\ &\approx \frac{fn(M_{Q\_input}) \times bits(M_{Q\_input})}{frame\_num\_MB} \end{aligned} \tag{15}$$

The frame complexity estimation is designed for all video compression standards. Different video standards use different quantization step size matrices and, in the following text, we derive the frame complexity functions for H.264/AVC and MPEG-2. Note that these derivations may also be used for MPEG-4, which uses two quantization modes wherein mode 0 is similar to MPEG-2 and mode 1 is similar to H.264.

3.2 H.264 frame complexity estimation

H.264 (also known as MPEG-4 Advanced Video Coding or AVC) uses a QP to determine the quantization level. The QP can take one of 52 values [32]. The QP is used to derive the quantization step size, which in turn is combined with a scaling matrix to derive the quantization step size matrix. An increase of 1 in QP results in a corresponding increase in quantization step size of approximately 12%. As shown in Equation 13, this change in QP results in a corresponding increase in *quantization complexity factor* of a factor of approximately 1.1 and a decrease in the number of frame bits by a factor of $\frac{1}{1.1}$. Similarly, a decrease of 1 in QP results in an increase by a factor of 1.1 in the number of frame bits.

When calculating the quantization complexity factor, $fn\ (M_{Q\_input})$, for H.264, the reference QP used is 26 (the midpoint of possible QP values) to represent average quality. This factor, defined in Equation 13, is shown specifically for H.264 in Equation 16. The denominator, the reference quantization step size matrix, is that obtained using a QP of 26 and the numerator is the average of the quantization step size matrices of the coded macroblocks in the frame. The average QP is got by averaging QP values over all the coded macroblocks in the frame, and it does not need to be an integer. If the average QP in the frame is 26, then the ratio becomes unity. If the average QP in the frame is 27, then the ratio is 1.1, an increase by a factor of 1.1 from unity. Each increase in QP by 1 increases the ratio by another factor of 1.1. Thus, the ratio in Equation 13 can be written with the power function shown on the right-hand side of Equation 16:

$$\begin{aligned} fn(M_{Q\_input}) &= \frac{\sum_{i=0}^{7} m_{frame\_QP\_input\_i}}{\sum_{i=0}^{7} m_{QP26\_i}} \\ &= 1.1^{(frame\_QP\_input-26)} \end{aligned} \tag{16}$$

The frame complexity can then be calculated using Equations 15 and 16.

3.3 MPEG-2 frame complexity estimation

In MPEG-2, the parameters *quant_scale_code* and *qscale_type* specify the quantization level [33]. The *quant_scale_code* specifies a *quant_scale* which is further weighted by a weighting matrix, *W*, to obtain the quantization stepsize matrix (Equation 17). The mapping of *quant_scale_code* to *quantizer_scale* can be linear or non-linear as specified by the *q_scale_type*:

$$M = quant\_scale \times W \tag{17}$$

MPEG-2 uses an 8 × 8 DCT transform and the quantization step-size matrix is 8 × 8, resulting in 64 quantization step-sizes for 64 coefficients after DCT transform. The low frequency coefficients contribute more to the total coded bits. In Equation 12, we set K = 16, and the average factors associated with the first 16 low frequency coefficients are assigned a value of 1 and the average factors associated with the high frequency coefficients are assigned a value of 0. Therefore, Equation 13 becomes

$$\begin{aligned} fn(M_{Q\_input}) &= \frac{\sum_{i=0}^{15} m_{Q\_input\_i}}{\sum_{i=0}^{15} m_{Q\_ref\_i}} \\ &= \frac{\sum_{i=0}^{15} w_{input\_i} \times quant\_scale_{input\_i}}{\sum_{i=0}^{15} w_{ref\_i} \times quant\_scale_{ref\_i}} \end{aligned} \tag{18}$$

In MPEG-2, the *quant_scale_code* has one value (between 1 and 31) for each macroblock. The *quant_scale_code* is the same at each coefficient position in the 8 × 8 matrix. Thus, the $quant\_scale_{input}$ and $quant\_scale_{ref}$, in Equation 18, are independent of i and can be factored out of the summation. For the reference, we choose 16 as the reference *quant_scale_code* to represent the average quantization. We use the notation *quant_scale* [16] to indicate the value of *quant_scale* when the *quant_scale_code* = 16. For the input bitstream, we calculate the average *quant_scale_code* for each frame over the coded macroblocks, and we denote it as $quant\_scale_{input\_avg}$.

The weighting matrix, *W*, used for intra-coded blocks is typically different from that used for non-intra blocks. Default weighting matrices are defined in the standard; however, the MPEG-2 encoder can define and send its own weighting matrix rather than use the defaults. For example, the MPEG-2 encoder developed by the MPEG Software Simulation Group (MSSG) uses the default

weighting matrix for intra-coded blocks and provides a non-default weighting matrix for non-intra blocks [34]. In the denominator of Equation 19, we use the MSSG weighting matrices as the reference:

$$fn(M_{Q\_input}) = \frac{quant\_scale_{input\_avg} \times \sum_{i=0}^{15} w_{input\_i}}{quant\_scale[16] \times \sum_{i=0}^{15} w_{ref\_i}} \quad (19)$$

To simplify, *quant_scale* [16] = 32 for linear mapping and *quant_scale* [16] = 24 for non-linear mapping. Also, the sum of the first 16 MSSG weighting matrix components for non-intra coded blocks is 301 and that for intra-coded blocks is 329. Thus, the denominator in Equation 19 is a constant and $fn(M_{Q\_input})$ can be rewritten as

$$fn(M_{Q\_input}) = \frac{1}{fnD}\left(quant\_scale_{input\_avg} \times \sum_{i=0}^{15} w_{input\_i}\right) \quad (20)$$

where

$$fnD = \begin{cases} 9632 & \text{linear, non-intra} \\ 7224 & \text{non-linear, non-intra} \\ 10528 & \text{linear, intra} \\ 7896 & \text{non-linear, intra} \end{cases} \quad (21)$$

The frame complexity can then be calculated using Equations 21 and 15.

3.4 Bitrate normalization using frame complexity

As discussed earlier, the bitrate estimate is normalized by the calculated frame complexity to provide an input to G.1070 that will yield measurements better correlated to subjective scores. Since the number of the frame bits is used in the frame complexity estimation [Equation 15], it can be seen that normalization will cause the bit rate to be canceled out. To maintain some consistency with the current G.1070 function inputs (bit rate, frame rate, and PLR), we want to prevent this cancelation, so the normalization process is revised. It is generally observed that, as the bit rate decreases, fewer macroblocks are coded (more macroblocks are skipped). Therefore, the percentage of macroblocks that are coded can be used to represent the bit rate in Equation 15. Thus, we can compute the normalized bit rate as follows:

$$bitrate\_norm = \frac{bitrate}{frame\_complexity} = \frac{bitrate}{\left(\frac{num\_coded\_MB}{frame\_num\_MB}\right) \times fn(M_{Q\_input})} \quad (22)$$

3.5 Discussion

The proposed G.1070E model takes the video content into consideration by normalizing the bitrates using the frame complexity. It reflects the subjective quality more accurately than the standard G.1070 model. In order to illustrate this, Figure 5 shows the performance of G.1070E, compared to G.1070, with respect to the pixel-domain reduced-reference NTIA-VQM score [28] for the same sequence as shown earlier in Figure 3. It can clearly be seen that, unlike G.1070, the quality predicted by G.1070E adapts to the variation of video content characteristics. The superior performance of G.1070E is demonstrated in Section 4.2 by providing experimental results over several video datasets with MOS scores.

4 Experimental results

In this section, experimental results are provided to demonstrate the effectiveness of the parameter estimation methods proposed in Section 2 as well as the quality prediction accuracy of the enhanced G.1070E model proposed in Section 3.

4.1 Parameter estimation accuracy evaluation

To evaluate the accuracy of parameter estimation, 20 original standard sequences of CIF resolution were used. Overall, 100 test bitstreams were generated by encoding these original sequences using a H.264 encoder with various combinations of bit rates and frame rates. These test bitstream files were further degraded by randomly erasing RTP packets at different rates. Overall 900 test bitstreams with coding and packet-loss distortions were used. Table 3 summarizes the test content and the conditions used for testing.

4.1.1 Bit rate estimation

In order to evaluate the accuracy of bit rate estimation with increasing PLR, the estimates of bit rate at non-zero PLRs were compared with the 0% packet-loss case which is considered as the ground truth.

Figure 6 shows the plot of estimated bitrate for the *akiyo* sequence having an overall average bitrate of 128 kbps at 30 fps for PLRs of 0, 1, 3, 5 and 10%. From the plot, it can be noticed that as the PLR increases, the bitrate estimation accuracy decreases. However, over most of the sequence duration, the bitrate estimation does not stray much from the 0% packet-loss case, and thus is quite robust to packet loss. Figure 7 shows the plot of estimated *normalized* bitrate for the *akiyo* sequence having an overall average bitrate of 128 kbps at 30 fps for PLRs of 0, 1, 3, 5 and 10%. Here too, it may be observed that the normalized bit rate estimation is robust to packet loss. Notice that as packet loss increases the number of bit rate estimates decreases, since fewer video frames are received at the decoder.

Figure 8 shows the scatter plots of ground truth bitrate estimation at 0% PLR versus bitrate estimation at non-zero PLRs for the entire test sequence suite. Note that for perfect estimation the scatter plot should be a

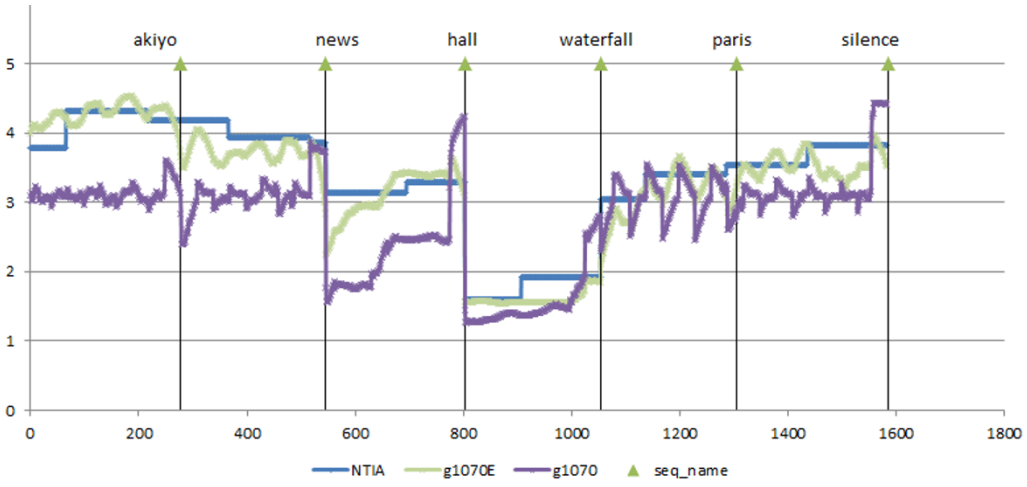


Figure 5 G.1070E quality prediction for video scenes with varying content characteristics.

45◦ line. From the figure, it can be noticed that for 1% PLR, the scatter plot is very close to a 45◦ line. As the PLR increases to 3, 5 and eventually 10%, the scatter plot deviates more from the ideal 45◦ line. However, the estimation accuracy is still very high. This is confirmed by the very high Pearson correlation coefficient (CC) values and very small root mean squared errors (RMSEs).

4.1.2 Frame rate estimation

Similar to the preceding analysis, the accuracy of frame rate estimation is evaluated by comparing the estimates at various PLRs with those at 0% packet loss, which is considered to be the ground truth. It was observed that the scatter plots of ground truth frame rates at 0% PLR versus frame rates estimated at 1, 3, 5 and 10% PLR's were identical. Figure 9 shows the scatter plot for the 10% PLR case. It can be observed that the frame rate estimation is very accurate with a CC of 1 and RMSE of 0.

Additionally, the frame rate estimation was subjected to stress testing in order to test its robustness to high PLR. To do so, each original test bitstream is degraded with different PLR's starting from 0% and going up to 95% in steps of 5%. The frame rate estimates are compared with the ground truth frame rates for every packet-loss impaired bitstream. From the results, it is observed that the frame rate estimates obtained are accurate for all the test cases as long as the bitstreams were decodable. If the bitstream is not decodable (generally for PLR greater than 75%), there can be no frame rate estimation.

Note that the proposed frame rate estimation algorithm will fail in the rare event wherein packets belonging to every alternate frame get dropped before reaching the decoder, in which case no two consecutive timestamps can be received during the buffer window (here, set to 30 frames). However, this is only a failure insofar as the goal is to obtain the actual encoded frame rate and not the frame rate *observed* at the decoder (which in this case is exactly half the encoded frame rate).

4.1.3 PLR estimation

Accurate estimation of PLR is crucial because it is used as a correction factor for the bit rate estimate when packet loss is present. In order to analyze the accuracy

Table 3 Summary of test content and test conditions used for parameter estimation accuracy testing

Bitstreams	akiyo, bridge-close, bridge-far, bus, coastguard, container, flower-garden, football, foreman, hall, highway, mobile-and-calendar, mother-daughter, news, paris, silent, Stefan, table-tennis, tempete, waterfall
Bit rates	32 kbps, 64 kbps, 128 kbps, 256 kbps
Frame rates	6 fps, 10 fps, 15 fps, 30 fps
Packet-loss rates	0%, 1%, 2%, 5%, 10%
Loss patterns	2 random patterns

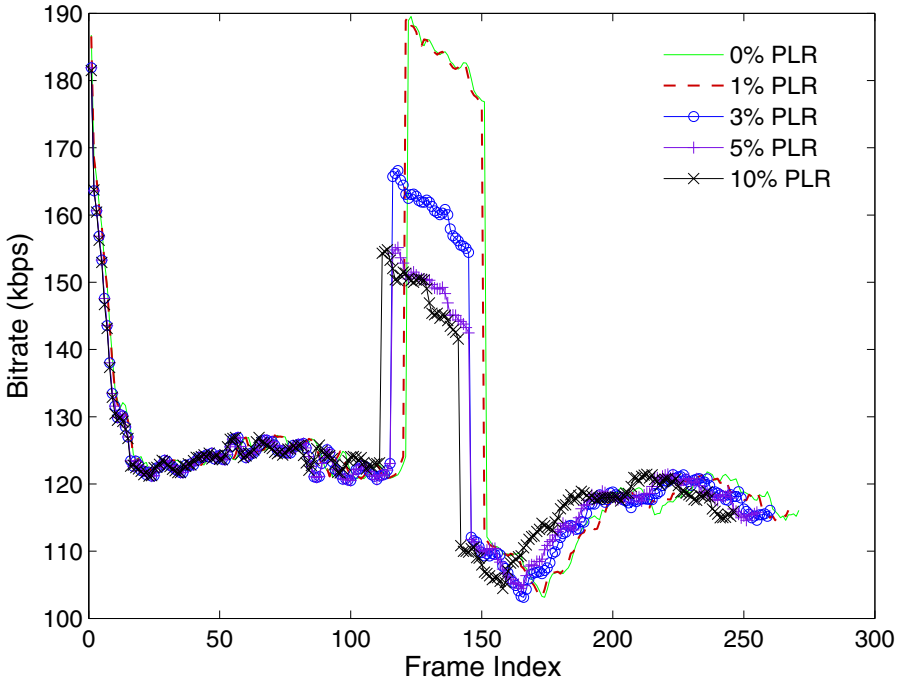


Figure 6 Plot of estimated bitrate for the *akiyo* sequence having an overall average bitrate of 128 kbps at 30 fps for various packet-loss rates.

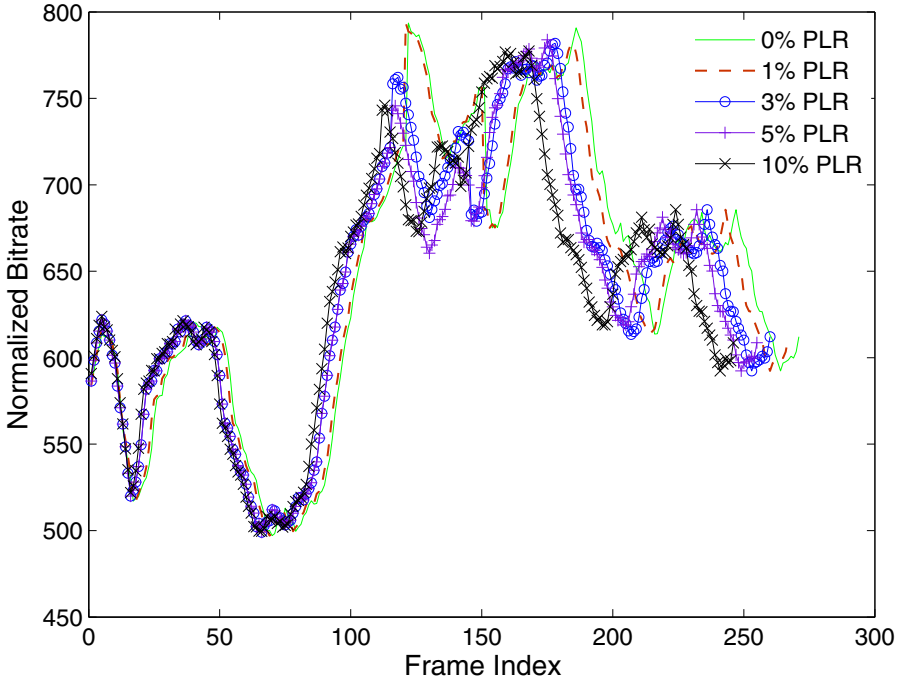
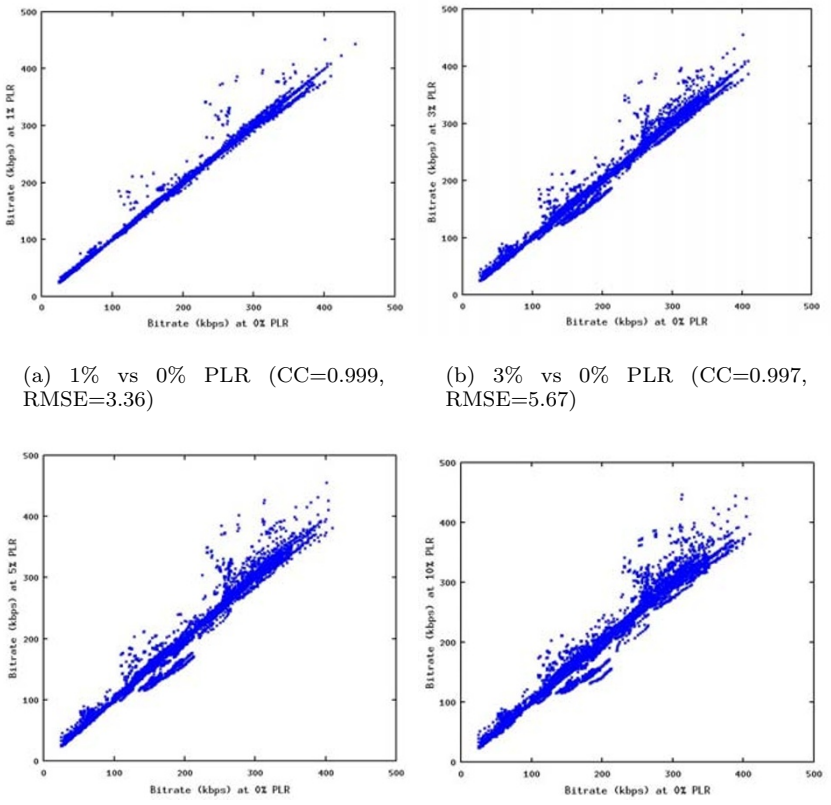


Figure 7 Plot of estimated normalized bitrate for the *akiyo* sequence having an overall average bitrate of 128 kbps at 30 fps for various packet-loss rates.



(a) 1% vs 0% PLR (CC=0.999, RMSE=3.36)

(b) 3% vs 0% PLR (CC=0.997, RMSE=5.67)

Figure 8 Scatter plots of ground truth bit rate estimation at 0% PLR vs. bit rate estimation at non-zero packet-loss rates.

of PLR estimation, we use the EPFL PoliMi database [35], which consists of CIF and 4CIF resolution videos that have 18 and 32 slices per frame, respectively, where each slice is encapsulated in one packet. This database was chosen for two reasons: (a) it provides tools to extract the location of packets lost, and (b) it enables a good visual representation of PLR estimation since it has a finer granularity of packet loss (i.e. sufficiently high number of packets per frame).

Figure 10 shows the estimated PLR (using the algorithm in Section 2.2.3) on the y-axis against the packet index on the x-axis for the standard CIF-resolution *Foreman* sequence degraded with 3% PLR. The vertical lines in the lower portion of the plot represent the actual location of packets lost. Note here that the PLR estimates are instantaneous values over an N-frame window and may not always be equal to the long-term average PLR. Thus, in Figure 10, the instantaneous PLR values range from about 0.5 to 7%. However, the average PLR over the whole sequence is close to the expected value of 3%.

Note that the impact of actual packets lost on the PLR can also be clearly seen. For example, for a short duration after 1000 packets, the number of packets lost increases causing a corresponding increase in the instantaneous PLR. Similarly, the number of packets lost between 2500 and 3500 is lower and this causes a drop in instantaneous PLR.

4.2 G.1070E quality prediction accuracy evaluation

In this section, we present experiment results comparing the performance of G.1070 (using the proposed parameter estimation methods in Section 2) and the proposed G.1070E method (Section 3), using three different testing datasets. According to the methods described in the G.1070 Recommendation, the 12 coefficients of G.1070 and G.1070E are trained on the same video dataset. In our experiments, the performance of the proposed methods are similar for H.264 and MPEG-2 bitstreams.

One experiment was conducted using a dataset with MOSs provided by the Image Group of Instituto de Telecomunicacoes, Instituto Superior Tecnico (IT-IST) [36]. The video GOP structure in this dataset is IBBP. Figure 11 shows the comparison between G.1070E and G.1070 for H.264 encoded sequences, and Figure 12 shows the comparison for MPEG2 encoded sequences. Based on the scatter plots shown in Figures 11 and 12 and the performance metrics in Tables 4 and 5, it may be observed that the proposed G.1070E outperforms G.1070.

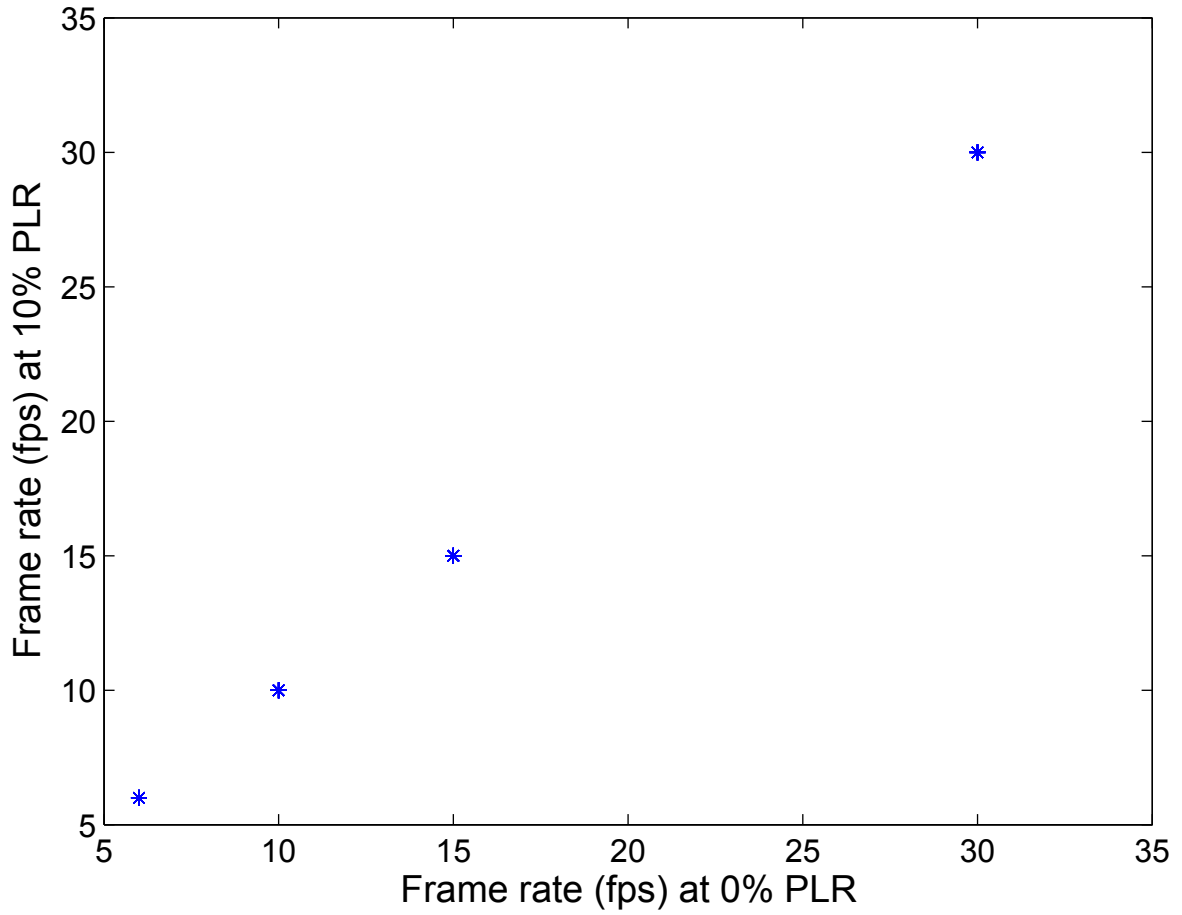


Figure 9 Scatter plot of ground truth frame rate estimation at 0% PLR vs. frame rate estimation at 10% packet-loss rate.

There is no packet loss in the IT-IST dataset. However, we also conducted the experiments using EPFL PoliMI Video Quality Assessment Database [35], which provides MOS scores by two academic institutions: Politecnico di Milano (PoliMI), and Ecole Politechnique Federale de Lausanne (EPFL). We used the video contents at 4CIF resolution and with six different PLR's [37]. The videos have the same GOP structure as IT-IST dataset. The frame-copy error concealment method has been used here. The scatter plots are shown in Figures 13 and 14, for EPFL MOS scores and PoliMI MOS scores, respectively. As shown in Table 6, the proposed G.1070E has a higher CC and lower RMSE than G.1070. In other words, even in the presence of packet loss, the proposed G.1070E can reflect the subjective scores better than G.1070.

Like G.1070, G.1070E is also a NR bitstream-domain objective video quality measurement model. Experimental result shows that G.1070E has a significantly higher correlation with subjective MOS scores and can reflect the quality of video experience better than G.1070. The expense paid for this improvement in quality prediction accuracy is the complexity involved in extracting additional parameters, e.g. QP, number of coded and total macroblocks, and in computing frame complexity.

5 Quality monitoring system and applications

The quality measurement tools described above have been incorporated into a real-time video quality monitoring system. We introduce the notion of a video quality agent. This is a software process that can analyze a bitstream and output a quality measurement. In order to calculate the G.1070 measurement, the agent must first estimate the bit rate, frame rate, and PLR as described in Section 2. Thus, it must partially decode the input bitstream to extract the main features: bit counts, time scales, time stamps, coded unit types, and sequence numbers. For calculation of the enhancements described in Section 3, the agent must also extract the quantization step size matrix for each macroblock. Thus, the agent does the decoding necessary to extract these features. Alternatively, the feature extraction can be built into an existing decoder. For example, a video player or transcoder can be modified to extract the features needed by the quality agent during decoding for

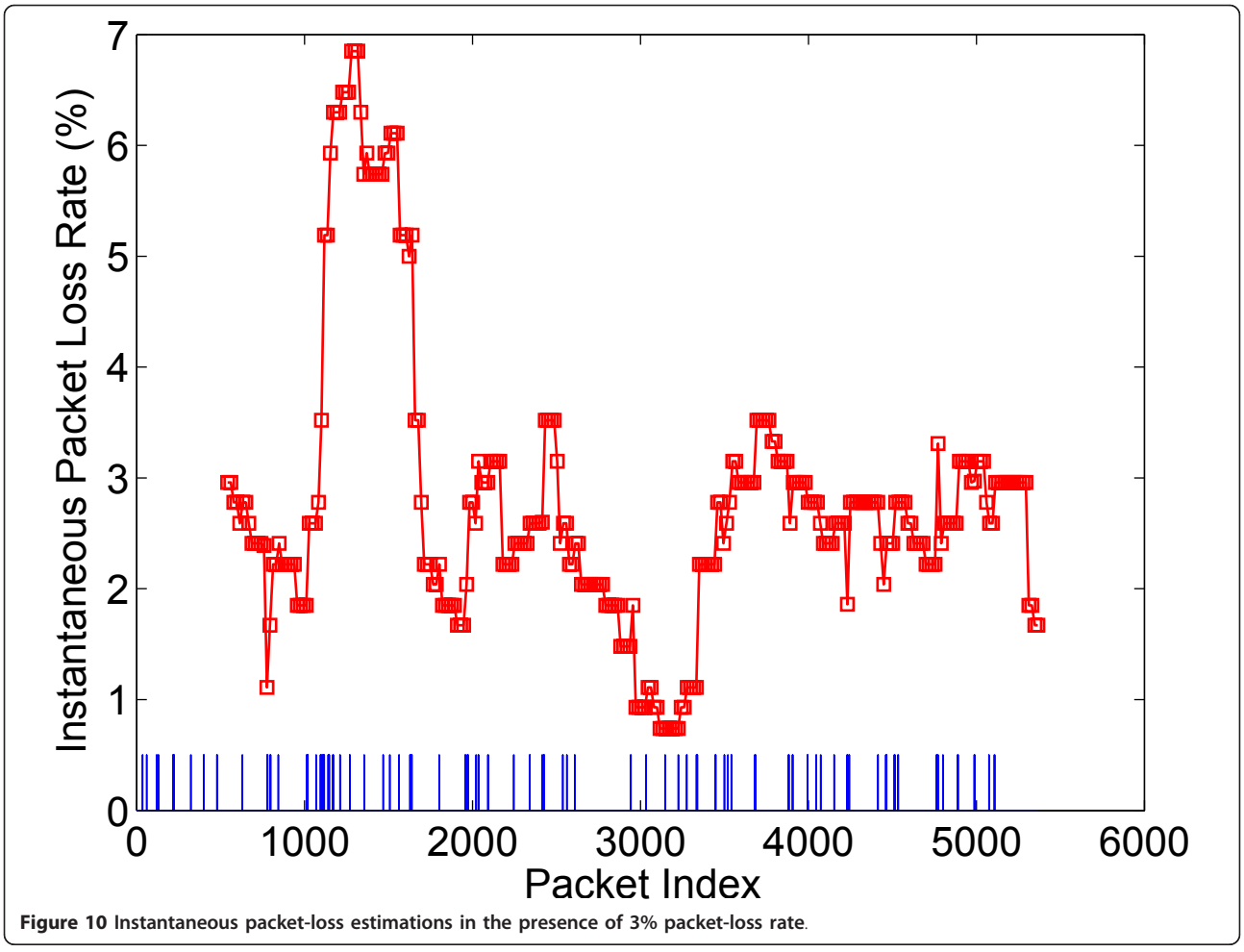


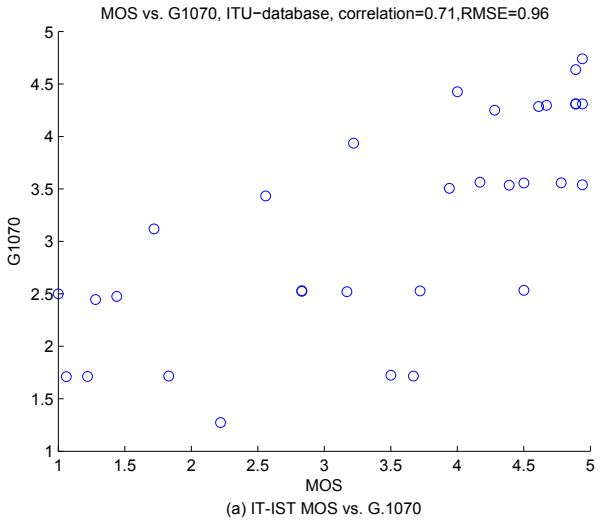
Figure 10 Instantaneous packet-loss estimations in the presence of 3% packet-loss rate.

playback. We use the term 'video quality agent' to refer to a software process, integrated with an existing decoder or with its own decoding ability, that can analyze a bitstream, extract the necessary features, estimate the necessary parameters, calculate the quality estimates, and finally, communicate those measurements to another software process running in the network.

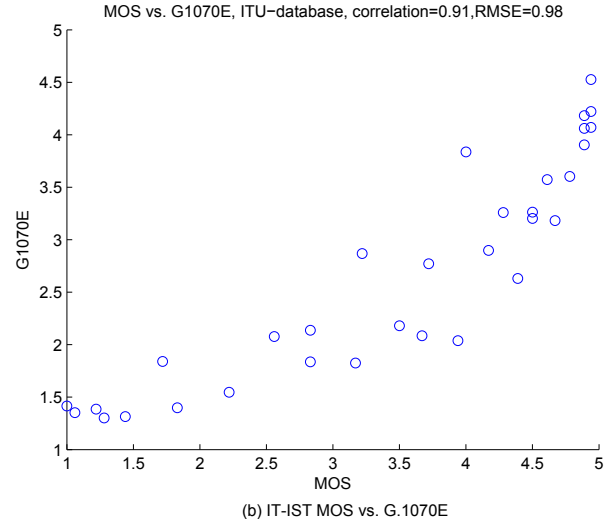
A video quality monitoring system is a collection of video quality agents all reporting their measurements back to a central network collection point where the measurements are aggregated for further analysis. As mentioned above, video quality agents can be embedded into video players on mobile handsets, in set-top boxes, on computers, etc. In addition, agents with their own decoding capabilities can be deployed at a streaming server, transcoder, or router.

Consider the illustration in Figure 15 in which a number of video quality agents are deployed to monitor the quality of a video stream as it is transcoded, packaged, and served to a mobile phone. In this example, the bold lines are video streams and the thin dashed lines represent quality data sent to an *aggregator*. This communication of quality data to the aggregator occurs in real-time. At the extreme, each agent is generating a quality measurement for each frame of video and those measurements are immediately sent to the aggregator.

In the small system of Figure 15, the aggregator is receiving quality measurements about the same video stream from four different agents. By synchronizing these four streams of data, the aggregator can monitor the degradation in quality as the video passes through the transcoder, packager, server, and transmission network. The transcoder is expected to degrade the video quality. The goal of transcoding in this system is to modify the source content to match the bit rate, frame rate, and codec type supported by the target network and media player. By comparing the quality measurements from before and after transcoding, this damage can be quantified and compared to pre-established thresholds. Alerts can be issued when the drop in quality exceeds these thresholds. The packaging and serving processes are not expected to degrade the video quality.

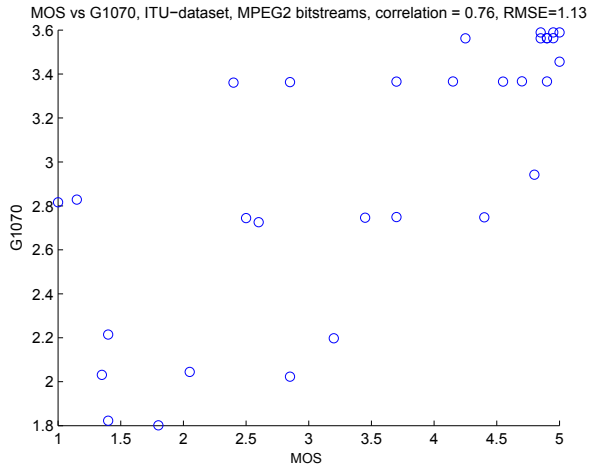


(a) IT-IST MOS vs. G.1070

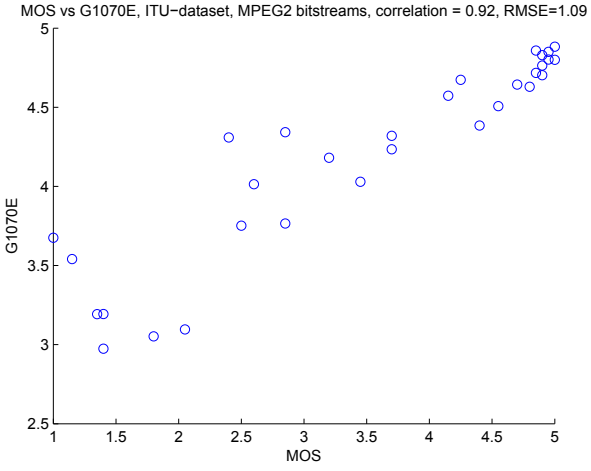


(b) IT-IST MOS vs. G.1070E

Figure 11 Scatter plots of predicted quality against MOS data for the IT-IST H.264 dataset.



(a) IT-IST MOS vs. G.1070



(b) IT-IST MOS vs. G.1070E

Figure 12 Scatter plots of predicted quality against MOS data for the IT-IST MPEG2 dataset.

Differences in quality measurements between these two points can indicate problems in the video data paths. Finally, measurements from the handset represent the user experience. Differences in quality between the video served and that received can be attributed to the communication network. In considering the changes in quality, the aggregator is constructing a measure of the fidelity of the channel between measurement points. This allows the aggregator to identify the source of quality degradations and fits nicely into the standard network management paradigm.

A number of video service applications can be modeled with a generalized version of Figure 15. Consider the case in which the devices are operated by different companies. At each hand-off point, there are service level agreements (SLA) specifying a minimum quality of

Table 4 The comparison between G.1070E and G.1070 for the IT-IST H.264 encoded sequences

	G.1070	G.1070E
Correlation coefficient	0.71	0.91
Spearman rank correlation	0.81	0.94

Table 5 The comparison between G.1070E and G.1070 for the IT-IST MPEG2 encoded sequences

	G.1070	G.1070E
Correlation coefficient	0.76	0.92
Spearman rank correlation	0.82	0.94

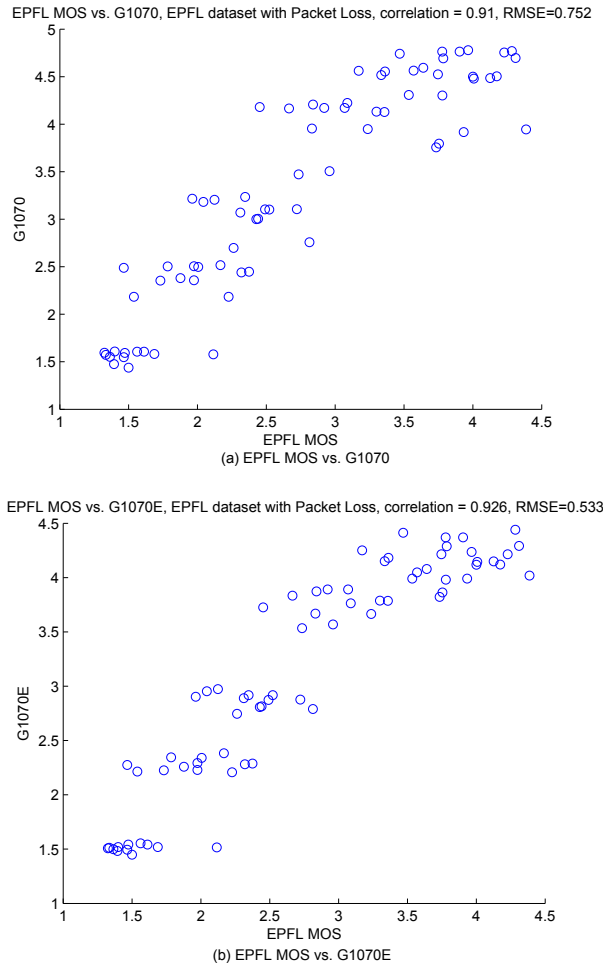


Figure 13 Scatter plots of predicted quality against MOS data for the EPFL PoliMI Video Quality Assessment Database. MOS values collected by EPFL.

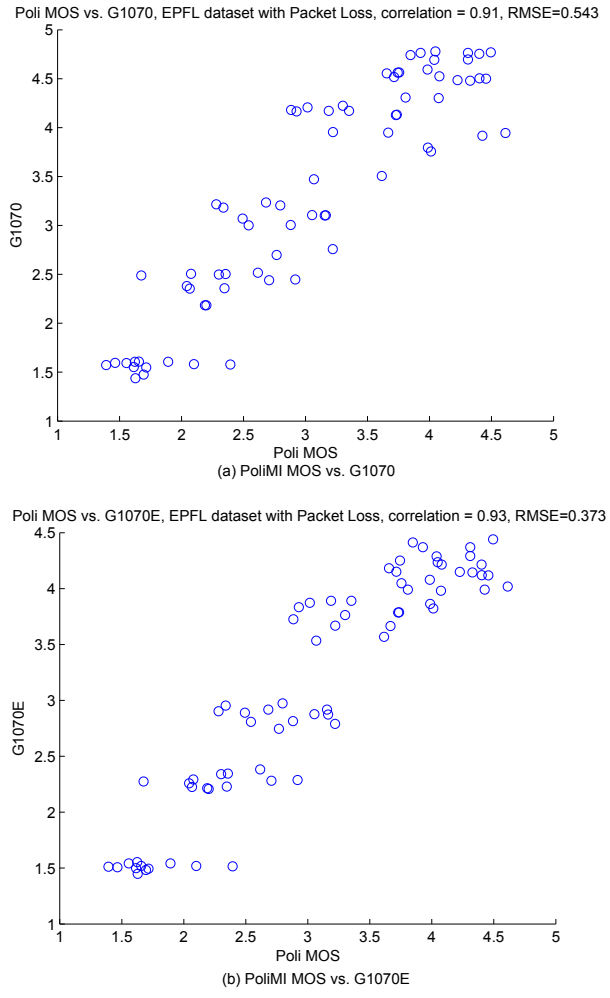


Figure 14 Scatter plots of predicted quality against MOS data for the EPFL PoliMI Video Quality Assessment Database. MOS values collected by PoliMI.

service. But these SLAs could also specify a maximum amount of degradation to the video quality. With the ability to measure quality, systems could manage their bandwidth usage, insuring that the amount of bandwidth used is just enough necessary to meet the quality targets. Similarly, network operators can establish tiered services in which the video quality delivered to the viewer depends on the price paid. More expensive plans deliver higher quality video. To do this, the quality of the video must be measured and controlled. A final example is quality assurance of end user video. Most video network operators today are not aware of any video quality problems in their network until they receive a complaint from a customer. A network instrumented to measure video quality will give operators the ability to identify and troubleshoot problems more quickly.

In many cases, it seems that the quality measurements shown in Figure 15 can be made with a reference. For example, if the video gateway is modifying the stream, it can measure the quality of the output relative to the input and thus report the level of degradation for which it is responsible. It is not clear, however, how a number of these relative quality measurements can be collected to provide insight into the overall impact on quality (it is likely that a simple linear summation or average would be insufficient). Further, in many applications, the various components in the network are controlled by different parties who each have an incentive to report

Table 6 The comparison between G.1070E and G.1070 for the EPFL PoliMI Video Quality Assessment Database

	EPFL G.1070	EPFL G.1070E	PoliMI G.1070	PoliMI G.1070E
Correlation coefficient	0.91	0.93	0.91	0.93
Spearman rank correlation	0.90	0.93	0.88	0.91

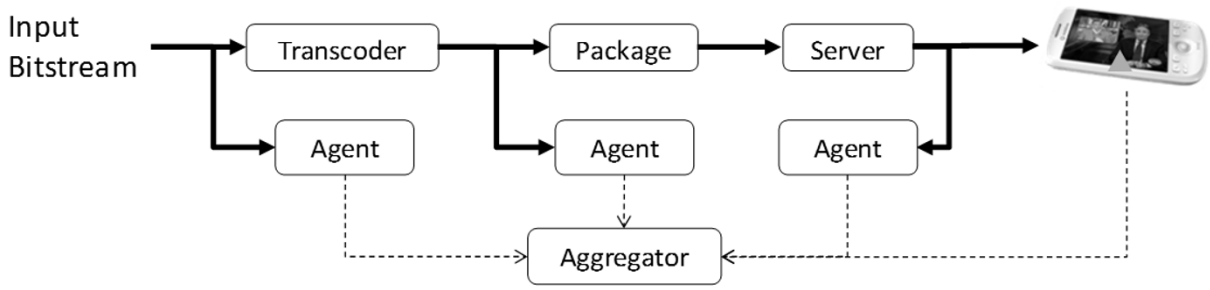


Figure 15 Video Quality Monitoring System composed of a number of video quality agents and an aggregator. (The '*triangle*' symbol placed on the cellphone represents the embedded agent.)

very slight, if any, degradation in quality; true or not. For these reasons, we propose this agent-aggregator general system structure with the use of NR video quality models to measure relevant aspects of the video.

As we seek to use the proposed quality models in the context of a system like Figure 15, a number of practical challenges needs to be properly addressed. There are two synchronization issues that arise in the implementation of a system similar to that shown in Figure 15. First, consider multiple network devices (many versions of server, network, end-point all running in parallel), all reporting quality measurements to a single aggregator. The system must be able to establish which measurements can serve as references to which other target measurements. Once that first synchronization issue has been addressed, the two streams of measurement data, target and reference, must be temporally aligned. A tight computational and memory constraints at some measurement points is another concern. The mobile devices usually have limited available resources including battery power, memory, and compute cycles. Since most mobile devices will decode the received bitstreams and display the video anyway, fortunately, the extra computation of applying the proposed quality metric in these devices is minor (some experimental statistics of the overhead related to the quality calculation are presented in Section 3). However, computational challenges exist in less likely spots. A video server or switch may have very powerful processors, large memory footprints, and plenty of electrical power, but these devices are also tasked with serving a large number of streams simultaneously. Adding a partial decoding/extraction process to each stream may bring considerable burden to some network nodes.

6 Conclusion

The ITU-T standardized G.1070 video quality model is widely used as a video quality *planning* tool for video conferencing applications. It takes as inputs the target bitrate and frame rate as well as the expected PLR of the channel. However, there are two technical challenges to extend this model for real-time quality *monitoring* for general video applications.

First, in the quality monitoring scenario, the bit rate and frame rate of the bitstreams and the actual PLR of the network are not known and need to be estimated. Second, the video content characteristics significantly impact the encoded bitrate of different video scenes at similar quality levels. This content-sensitivity issue may not be obvious in the context of video conferencing where the content is homogeneous, but its impact is felt when measuring the quality of general videos with varying characteristics.

To address the above problems, we first enable quality monitoring using G.1070 by presenting methods to continuously estimate the bit rate, frame rate, and PLR from received bitstreams. Then, we proposed a novel enhanced G.1070 (G.1070E) system, which compensates for the impact of varying video content characteristics on encoding bit rate by normalizing the bit rate with estimated video complexity. The improved quality prediction accuracy of the proposed G.1070E model is validated by experimental results comparing the predicted quality with MOS data collected from subjective tests.

Finally, we have presented an emerging application that can efficiently use the proposed real-time video quality monitoring method for diagnosing network problems and ensuring end user video quality.

Competing interests
The authors declare that they have no competing interests.

Received: 6 June 2011 Accepted: 6 December 2011
Published: 6 December 2011

References

1. K Seshadrinathan, R Soundararajan, A Bovik, L Cormack, Study of subjective and objective quality assessment of video. IEEE Trans Image Process. **19**(6), 1427–1441 (2010)
2. S Winkler, *Digital Video Quality: Vision Models and Metrics*, (Wiley, New York, 2005)

3. P Marziliano, F Dufaux, S Winkler, T Ebrahimi, Perceptual blur and ringing metrics: applications to JPEG2000. Signal Process Image Commun. **19**, 163–172 (2004). doi:10.1016/j.image.2003.08.003
4. R Ferzli, L Karam, A human visual system-based model for blur/sharpness perception, in *International Workshop on Video Processing and Quality Metrics (VPQM)* (2006)
5. D Liu, Z Chen, F Xu, X Gu, No reference block based blur detection, in *International Workshop on Quality of Multimedia Experience (QoMEX)* (2009)
6. Z Wang, H Sheikh, A Bovik, No reference perceptual quality assessment of JPEG compressed images, in *IEEE International Conference on Image Processing (ICIP)* (2002)
7. R Babu, A Perkis, An HVS-based no-reference perceptual quality assessment of JPEG coded images using neural networks, in *IEEE International Conference on Image Processing (ICIP)* (2005)
8. Z Wang, A Bovik, B Evans, Blind measurement of blocking artifacts in images, in *IEEE International Conference on Image Processing (ICIP)* (2000)
9. R Muijs, I Kirenko, A no-reference blocking artifact measure for adaptive video processings, in *European Signal Processing Conference* (2005)
10. Z Lu, W Lin, BC Seng, S Kato, S Yao, E Ong, XK Yang, Measuring the negative impact of frame dropping on perceptual visual quality. SPIE Human Vision and Electronic Imaging. **5666**, 554–562 (2005)
11. KC Yang, CC Guest, K El-Maleh, PK Das, Perceptual temporal quality metric for compressed video. IEEE Trans Multimedia. **9**, 1528–1535 (2007)
12. YF Ou, Z Ma, T Liu, Y Wang, Perceptual quality assessment of video considering both frame rate and quantization artifacts. IEEE Trans Circuits Syst Video Technol. **21**(3), 286–298 (2011)
13. RR Pastrana-Vidal, JC Gicquel, Automatic quality assessment of video fluidity impairments using a no-reference metric, in *International Workshop on Video Processing and Quality Metrics (VPQM)* (2006)
14. R Babu, A Bopardikar, A Perkis, OI Hillestad, No-reference metrics for video streaming applications, in *International Workshop on Packet Video* (2004)
15. H Rui, C Li, S Qiu, Evaluation of packet loss impairment on streaming video. J Zhejiang Univ Sci. **7**(Suppl I), 131–136 (2006)
16. A Reibman, D Poole, Predicting packet-loss visibility using scene characteristics, in *International Workshop on Packet Video* (2007)
17. TL Lin, S Kanumuri, Y Zhi, D Poole, P Cosman, A Reibman, A versatile model for packet loss visibility and its application to packet prioritization. IEEE Trans Image Process. **19**(3), 722–735 (2010)
18. T Liu, *Perceptual quality assessment of videos affected by packet-losses*, (PhD thesis, Polytechnic Institute of New York University, 2010)
19. S Mohamed, G Rubino, A study of real-time packet video quality using random neural networks. IEEE Trans Circuits Systems Video Technol. **12**(12), 1071–1083 (2002). doi:10.1109/TCSVT.2002.806808
20. A Takahashi, K Yamagishi, G Kawaguti, Recent activities of QoS/QoE standardization in ITU-T SG12, in *NTT Technical Review* (2008)
21. O Verscheure, P Frossard, M Hamdi, User-oriented QoS analysis in MPEG-2 video delivery. Real-time Image. **5**(5), 305–314 (1999). doi:10.1006/rtim.1999.0175
22. F Yang, S Wan, Q Xie, H Wu, No-reference quality assessment for networked video via primary analysis of bit stream. IEEE Trans Circuits Syst Video Technol. **20**(11), 1544–1554 (2010)
23. Recommendation ITU-T G1070: Opinion Model for Video-telephony Applications (2007)
24. K Yamagishi, T Hayashi, Parametric packet-layer model for monitoring video quality of IPTV services, in *IEEE International Conference on Communications* (2008)
25. B Belmudez, S Moller, Extension of the G.1070 video quality function for the MPEG2 video codec, in *International Workshop on Quality of Multimedia Experience (QoMEX)* (2010)
26. J Joskowicz, J Ardao, Enhancements to the opinion model for video-telephony applications, in *Fifth International Latin American Networking Conference* (2009)
27. N Narvekar, T Liu, D Zou, J Bloom, Extending G.1070 for video quality monitoring, in *IEEE International Conference on Multimedia and Expo (ICME)* (2011)
28. S Wolf, M Pinson, Video quality measurement techniques. National Telecommunications and Information Administration (NTIA) Report (2002)
29. B Wang, D Zou, R Ding, T Liu, S Bhagavathy, N Narvekar, J Bloom, Efficient frame complexity estimation and application to G.1070 video quality monitoring, in *International Workshop on Quality of Multimedia Experience (QoMEX)* (2011)
30. J Yang, Q Zhao, L Zhang, The study of frame complexity prediction and rate control in H.264 encoder, in *International Conference on Image Analysis and Signal Processing (IASP)* (2009)
31. L Tian, Y Sun, S Sun, Frame complexity prediction for H.264/AVC rate control, in *IEEE International Conference on Multimedia and Expo (ICME)* (2009)
32. T Wiegand, G Bjontegaard, G Sullivan, A Luthra, Overview of the H.264/AVC video coding standard. IEEE Trans Circuits Syst Video Technol. **13**, 560–576 (2003)
33. ISO/IEC 13818-2 MPEG2 (1995)
34. MPEG-2 video decoder version 12http://www.mpeg.org/MPEG/MSSG
35. EPFL PoliMI Video Quality Assessment Database (version 2.0)http://mmspl.epfl.ch/vqa
36. Instituto Superior Tecnico of Instituto de Telecomunicacoes datasethttp://amalia.img.lx.it.pt
37. FD Simone, M Naccari, M Tagliasacchi, F Dufaux, S Tubaro, T Ebrahimi, Subjective assessment of H.264/AVC video sequences transmitted over a noisy channel, in *International Workshop on Quality of Multimedia Experience (QoMEX)* (2009)

doi:10.1186/1687-6180-2011-122
Cite this article as: Liu *et al.*: Real-time video quality monitoring. *EURASIP Journal on Advances in Signal Processing* 2011 **2011**:122.