

Quality-based Adaptive Video Over the Internet

Xiaoxiang Lu, Shu Tao, Magda El Zarki, Roch Guérin *

Dept of ESE, University of Pennsylvania, Philadelphia, PA 19104
Dept of ICS, University of California, Irvine, CA 92697

Keywords: Video quality, adaptation, ITS model

Abstract

IP networks and the Internet in general are subject to a wide range of fluctuations of the end-to-end bit rate available to applications. This uncertainty has motivated the development of adaptive applications, i.e., applications that can adjust their resource requirements as a function of what the network can provide. In this paper, we explore an adaptation scheme that is not directly driven by these fluctuations, but based on the impact they have on the quality perceived by the application. Our focus is on the potential benefits of such a “quality-based” adaptation approach for video applications. We begin with a brief introduction of objective video quality assessment and its integration into an adaptive transmission system. Next, we discuss the joint impact of packet loss and encoding rate on video quality, based on which a simple quality feedback-control system has been built. Adaptation is carried out by measuring the quality of the received video and comparing it to a baseline reference. Our preliminary experimental results show both the viability and the benefits of using video quality as the basis of adaptation. Possible directions of future research are also discussed at the end of the paper.

1. INTRODUCTION

Although video streaming and conferencing have become popular in the Internet, the user satisfaction of such applications cannot be guaranteed. This is due to the best-effort nature of today’s Internet. Video applications usually have tight requirements on bandwidth, loss and delay, that can often not be met in a shared network environment like Internet. There are two possible approaches to solving the problem. One can either have the network explicitly manage the utilization of its resources to make performance more predictable, or design applications adaptive to changing network conditions. Introducing QoS in the network (e.g., Interserv and DiffServ) falls in the first category, but although various network QoS solutions are well understood and have been extensively studied, the future of their deployment remains uncertain. Therefore, in this paper we assume a standard the best-effort network. In such an environment, the ability

of a video flow to adjust its behavior in response to network conditions can be instrumental in bridging the gap between available network resources and application’s requirements.

To design an adaptive application, two basic questions need to be answered: (1) what are the triggers for adaptation, and (2) how should adaptation be performed. In answering the first question, most existing adaptation schemes probe the network path and infer changes in available network resources from changes in delay or loss statistics. This in turn leads to the adaptation of the video transmission rate. For instance, the Rate Adaptation Protocol (RAP) [11] and TCP Friendly Rate Control (TFRC) [3] use packet losses or loss events as indicators of congestion. In answering the second question, most systems rely on algorithms or models to estimate the proper transmission rate. For example, RAP uses an additive-increase-multiplicative-decrease (AIMD) algorithm for rate control, and TFRC uses a throughput model of TCP. Once the transmission rate has been estimated, various scalable video encoding methods (e.g., altering encoder’s quantization parameter, changing frame rate, dropping or adding coding layers, etc.) are used to match the video data rate with the estimated transmission rate.

In this paper, we explore instead the benefits of quality-based adaptation. In other words, a change in video quality is the main trigger for adaptation, and the objective is to maintain quality as high as possible under fluctuating bandwidth conditions. The change in quality is based on comparing the quality of the video generated at the decoder to that available at the encoder. The quality information is obtained through a model introduced in [14]. Based on this model, we implemented an end-to-end video transmission system to test different adaptation schemes. Through our initial experiments, we found that quality-based adaptation can successfully maintain perceived video quality in variable bandwidth conditions. Furthermore, the performance of such a scheme does not appear to be too sensitive to variations in video content.

The rest of this paper is structured as follows. In section 2, we briefly introduce an objective video quality assessment model and how we integrate it into a video transmission system. In section 3, we investigate the impact of packet loss and quantization parameters on quality. We sketch in Section 4 an algorithm using quality feedback information to control the encoding and transmission of video data. Some preliminary experimental results on the performance of the scheme are presented in Section 5. Section 6 discusses the results of our experiments and identifies several open issues in the current

*Shu Tao and Roch Guérin are with the Department of Electrical and System Engineering, University of Pennsylvania. Email: {shutao@seas.upenn.edu; guerin@ee.upenn.edu}. Xiaoxiang Lu and Magda El Zarki are with the Department of Information and Computer Science, University of California, Irvine. Email: {lxiaoxia@ics.uci.edu; elzarki@uci.edu}. The work of the authors has been supported in part through NSF grant ANI-9906855.

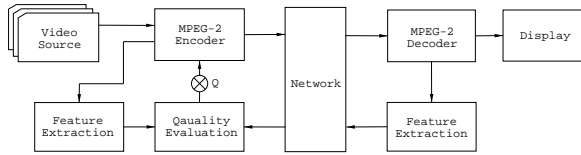


Figure 1: The architecture of a quality-based adaptive video transmission system.

scheme. Section 7 concludes the paper and outlines the various extensions we are currently exploring.

2. QUALITY-BASED ADAPTATION

The most accurate way of assessing video quality consists of subjective testing, i.e., gathering a group of people to watch and grade a video segment as it is being played back. The objective assessment approach, on the other hand, makes use of a mathematical model (based on the human visual system or signal attributes) and computes quality scores. A good objective assessment model is able to generate results matching subjective scores well, regardless of video content and distortion patterns. In order to conduct objective quality measurements, knowledge of both the original video frames and the received video frames is required in order to perform the required comparison. This is possible when performing an off-line test, but not feasible in the context of real-time adaptation, where the received and original frame sequences are obviously not both available either at the sender or at the receiver. This means that we cannot incorporate metrics like Peak Signal-to-Noise Ratio (PSNR) into a real-time adaptive system. Fortunately, the video quality metric developed by the Institute for Telecommunication Science (ITS) helps us overcome this difficulty.

2.1 The ITS Model

The ITS model [14][13] makes continuous quality measurements by (1) extracting statistics both from the reference video frames and from the frames to be measured, (2) communicating the extracted information between the sender and the receiver through an ancillary data channel, (3) computing individual parameters from the statistics that are indicative of different perceptual aspects of video quality, and (4) calculating a composite quality metric combining the individual parameters. In order to extract feature information, the model first divides video frames into Spatial-Temporal regions (S-T regions). In each S-T region, the parameters (features) characterizing spatial, temporal and chrominance activities are filtered out for future computations. By selecting the number of pixels encompassed by the S-T region appropriately, we can significantly reduce the amount of data that needs to be exchanged between sender and receiver, while keeping the resulting quality metric close to the subjective score. The composite quality index ranges from 0 (best quality) to 1 (worst quality). Experiments [14] show that this model successfully captures the correlation between measurable quality variables and human perception. The results generated by this model have been reported [14] to match subjective scores reasonably well, when testing various video clips with different types of distortions. The approach is also computationally efficient so that it can be used for real-time quality assessment.

Based on the ITS model, we implemented a video transmission soft-

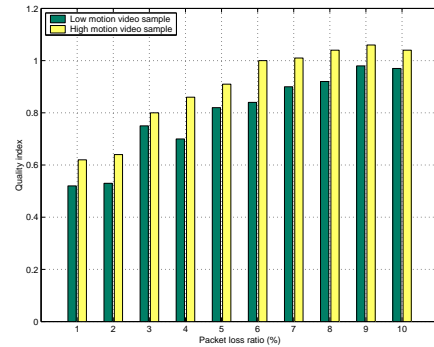


Figure 2: The impact of packet loss on video quality. The quality evaluation results of both a low motion video sample and a high motion video sample are plotted. The packet loss is uniform and the loss ratio varies from 1% to 10%.

ware with quality feedback control. The architecture of our system is shown in Fig. 1. The video source data, after being compressed by an MPEG-2 encoder, is packetized and transmitted using UDP. The features of the original frames and the received frames are extracted at the encoder and at the decoder, respectively. When video transmission begins, a TCP connection is set up between the sender and the receiver. The feature parameters of the received frames are then sent back to the sender over this connection. The evaluation of composite quality indices is performed by the sender. The results of this evaluation, after filtering, are then used to determine how to control the output rate of the encoder. In our current system, the rate control is simply achieved by altering the quantization scale (Q). More details about this implementation can be found in [8].

2.2 Why Quality-Based Adaptation

Video is different from other data applications (e.g., *ftp*) in the sense that its main target is not really maximizing throughput. The user cares more about the perceived video quality than the raw amount of bandwidth the video flow gets, even if the two are closely related. Therefore, the ultimate goal of adaptation for video applications should be to optimize the perceptual quality of the received video. Traditionally, video adaptation is built on top of congestion control mechanisms [16]. Basically during the control process, the sender reduces its bit rate in response to congestion, and increases its bit rate when no congestion is detected. In other words, congestion control is done from the network perspective, which largely ignores the needs of the underlying application.

To illustrate this, we take the AIMD rate control as an example. On one hand, although rate back-off after each packet loss or loss event is the right behavior for most data applications, it may not be the best reaction for video that can tolerate a certain level of data loss without much impairment to its perceptual quality. With error resilient coding or error concealing techniques, this tolerance threshold can be further enhanced. Furthermore, for video clips of different scene types, this threshold may vary. As a result, reacting with a rate decrease for every loss event typically results much more severe performance degradation than is necessary. Fig. 2 shows the effect of packet loss on perceptual quality for two different types of video clips. The quality of the high motion video sample gets impaired more than the low motion one, in spite of the fact that both

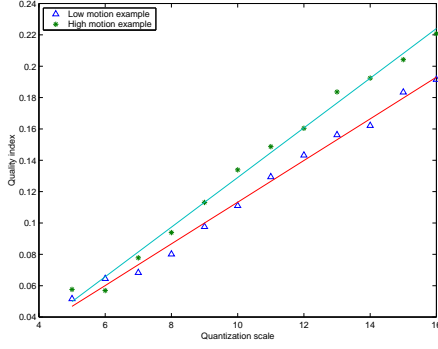


Figure 3: The effect of varying quantization scale on the encoding quality of different types of video.

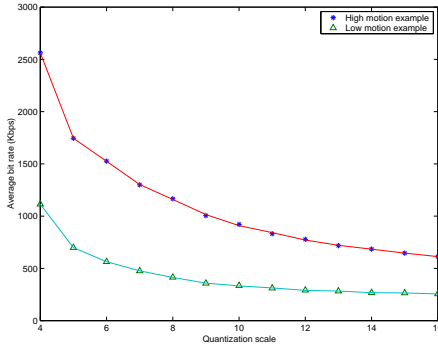


Figure 4: The effect of varying quantization scale on the average encoding bit rate of different types of video.

experience the same packet loss patterns. This further illustrates the limitation of a network based rate decrease that is oblivious to the characteristics of individual video streams. A similar limitation is present when using only network congestion (or lack thereof) to trigger rate increases. AIMD, like most other adaptation schemes, increases the data rate continuously during the non-congestion phase, until the next packet loss is detected. This is helpful to maximize throughput but does not necessarily optimize video quality. This is because a rate increase need not always result in a significant quality improvement, as video quality typically saturates¹ after the encoding rate exceeds a certain threshold [4]. It is also believed that watching a video stream that goes through frequent quality variations, even if it is of higher quality averaged over the whole clip, can be more annoying than watching a video sequence with relatively low but stable quality.

Besides loss, delay, and other performance impairments that occur during transmission, the other factor that influences video quality is the encoding process itself. A low bit rate coding reduces the probability of packet loss and, therefore, network impairments, but introduces its own impairments that can make quality intolerable from the user's perspective. Because of the differences in all these differ-

¹The saturation of quality is relative to the encoding bit rate. When the encoding bit rate becomes very high, increasing its value only results in marginal quality improvements when compared to a similar rate increase for a video with a lower encoding bit rate.

ent sources of impairments, it is difficult to define a single model that can fully account for their impacts on video quality. Furthermore, even if such a model was possible, operating an adaptation mechanism capable of taking all these factors into account would likely be too complex. Quality-based adaptation bypasses this complexity by instead focusing on the end result of all those impairments, namely, their ultimate impact on video quality.

3. APPLICATION QOS ANALYSIS

As discussed in Section 2.2, video quality is affected by many factors. Among them, the effects of encoding method and available bandwidth are of most interest. Before embarking on designing an adaptation scheme, it is therefore critical to understand the impact each one has on quality (application QoS). In this section, we carry out such an investigation for MPEG-2 VBR encoded video streams.

3.1 Joint Impact of Encoding and Bandwidth

In [12], encoding quality² is reported as exhibiting a linear relationship with Q , the encoding quantization step. Our experimental results coincide with this finding (see Fig. 3). For different types of video, the slopes of that linear function are, however, different. Typically, the higher the encoding complexity, the faster the quality decreases when increasing Q . This relationship can be described by the following formula, where q_{enc} is the encoding quality, χ_{enc} and q_0 are video-specific parameters (q_0 also depends on the reference image sequence used in the measurements).

$$q_{enc} = \chi_{enc} \cdot Q + q_0 \quad (1)$$

Meanwhile, Q determines the output bit rate of the encoder, the average value of which can be approximated by the following power function (see also Fig. 4):

$$R = \chi_R \cdot Q^{-\xi_R} \quad (2)$$

where R is the average bit rate, and χ_R and ξ_R vary according to the encoding complexity. Fig. 4 shows that the low motion video sample has lower average encoding bit rate than the high motion one when encoded with the same quantization parameters. Clearly, there is a trade-off in selecting Q : increasing Q results in a lower bit rate (therefore possibly lower packet loss ratio, or PLR), but lower encoding quality; decreasing Q improves encoding quality, while the resulting higher bit rate may cause more packet loss if bandwidth is not sufficient. Therefore, bit rate cannot obviously be considered as the only factor in designing adaptation mechanisms. For instance, one can decrease bit rate sufficiently to avoid further packet loss caused by congestion. However, the quality drop caused by such adjustment could be even more severe than that caused by the losses incurred when keeping the bit rate at its original value. Similarly, it is not always necessary to increase bit rate to fully utilize the available bandwidth, as the resulting gain in quality may be negligible.

²Here, by encoding quality we mean the quality of the decoded images with encoding loss, when compared to the reference image sequence. The reference could be either the original image sequence (i.e., without encoding loss) or the decoded image sequence from an encoded video sequence with higher encoding bit rate (i.e., with less encoding loss).

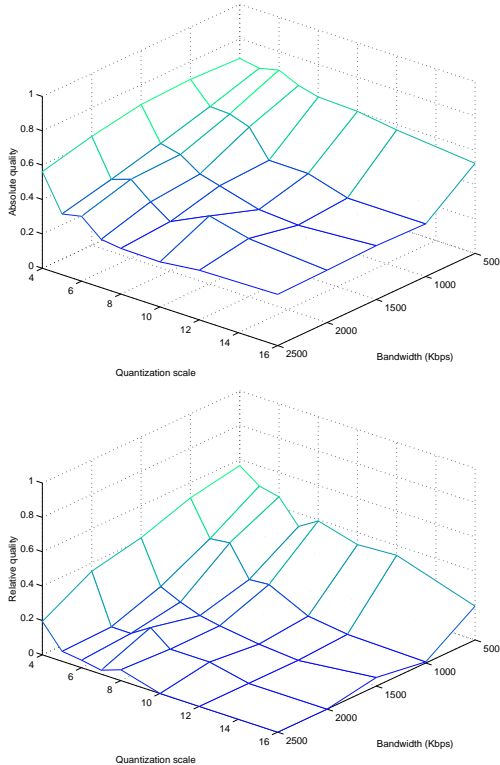


Figure 5: Joint impact of encoding and bandwidth on video quality: *absolute* quality (top) and *relative* quality (bottom).

Another important factor that affects quality is PLR. Although its relationship with quality has been studied in the literature, we found it difficult to directly convert PLR constraints into adaptation actions since different loss patterns may contribute differently to quality degradation. We therefore use the available bandwidth as the metric through which we measure the effect of the network.

A 3-D model [4] relating bandwidth and Q to quality is useful to optimize Q . However, there is no fixed model that will fit all types of video. To better understand the trends behind how bandwidth and quantization affect quality, we conducted a set of experiments using a sample video extracted from a movie. The experiments are carried out on an Ethernet testbed, which has a 10Mbps bottleneck link. We control the bandwidth available on the bottleneck link by generating a variable amount of background traffic on the link. Video streams (frame size of 240×352) with different encoding quality are transmitted through the network. We then measure the quality of the received video sequence against a reference. The results are plotted as the “absolute” quality indices in Fig. 5. Although these results are specific to the tested video, they are indicative of the general trend we observed for a number of other video clips we tested.

From Fig. 5, we observe the following: Let $q(x, y)$ denotes the quality index at point (x, y) , where x is the quantization scale and y is the available bandwidth (in Kbps). For each Q , there exists an x_t so that $q(x_1, y) > q(x_2, y)$ if $x_1 > x_2 > x_t$ (e.g., $x_t = 8$ when $y = 2500$). This means that when the bandwidth is suffi-

cient, increasing Q lowers quality. Typically, $x_t(y_1) > x_t(y_2)$, if $y_1 < y_2$, i.e., lower bandwidth calls for higher quantization in order to achieve optimal quality. For example, we see from the figure that $x_t(2500) = 8$, and $x_t(1500) = 12$. Similarly, for each Q (or x), there exists a y_t , so that $q(x, y_1) = q(x, y_2)$ if $y_1 > y_2 > y_t$. This is rather intuitive and simply means that once bandwidth is sufficient to eliminate packet losses, further increases in bandwidth do not affect quality when quantization is kept constant. For instance, $y_t = 1000$ for $x = 16$. The value of y_t depends on video stream’s average bit rate and its variation.

It should also be pointed out that none of the measurement points achieves perfect quality (a quality index of 0), even when the video experiences no packet loss. This is due to encoding losses, as we measured the quality using an absolute reference sequence encoded with $Q = 2$, while all the tested video sequences had a quantization index $Q \geq 4$. As a result, encoding losses are present in all of the tested frame sequences when compared with this reference.

The trade-off associated with PLR can also be observed from the figure. For instance, we have $q(12, 1500) < q(16, 2500)$, which means that if the available bandwidth is greater than 1.5 Mbps, having Q stay at 12 is better than increasing it. This is because even if having $Q = 12$ results in slightly more packet losses than for higher values of Q , their effect remains negligible. Actually, for each bandwidth condition y , x_t represents the optimal value of Q , corresponding to the best overall quality (considering both encoding and packet loss effects). A good adaptation scheme should, therefore, always adjust Q towards that optimal value. For example, suppose the adaptation starts at $(6, 1500)$ and the available bandwidth decreases to 1 Mbps, the operation point moves to $(6, 1000)$ if Q is not adjusted, which results in worse quality. In this case, the figure shows that the optimal point is on the right hand side of $x = 6$, hence the correct adaptation is to increase Q . Similarly, if starting from $(16, 1500)$, the optimal value can be achieved by decreasing Q close to 10.

In [4], a mathematical model was proposed to compute the optimal value of Q . However, it is difficult to parameterize that model without knowing the characteristics of the video clips in advance.

3.2 Relative Quality Versus Absolute Quality

As mentioned earlier, the best way to assess video quality is to have the original frame sequence as a reference. This reference should be without encoding loss or of lower encoding losses when compared to the sequence actually transmitted. We call the result of such a comparison “absolute quality”. For instance, this can be used with a recorded-video streaming application, for which the original frame sequence is available at the server. The server can then extract the features of these frames beforehand, and have them ready for comparison purposes. In other cases, a reference stream may, however, not be available for absolute quality comparison. This may be the case with live video streaming or conferencing. The video signal usually goes directly from the camera to the encoding card, and accessing the unprocessed data at the encoding card may be difficult. In such cases, the only reference available to the adaptation process is the output of the encoder. This reference is free of transmission losses, but already incorporates encoding losses. This means that the quality measurement in the adaptation is “relative”, reflecting mostly the effect of packet loss on the current encoding rate instead

of the joint impact of packet losses and encoding.

Nevertheless, even when using such a relative comparison, it is still possible to perform a meaningful adaptation. This is because video quality can be approximated as the sum of encoding quality and quality degradation caused by packet losses [4].

$$q = q_{enc} + \chi_L \cdot r_{loss} \quad (3)$$

Here, q is the overall quality, q_{enc} is the encoding quality and the second term is related to packet loss. χ_L is a parameter depending on both the encoding complexity and the average bit rate, r_{loss} is the packet loss ratio. According to Eq. (1), if we change quality reference, only the term q_0 will be significantly affected in encoding quality:

$$q'_{enc} = \chi_{enc} \cdot Q + q'_0 \quad (4)$$

Therefore, from Eqs. (1), (3) and (4) we know that under the same quantization scale and network conditions, the difference between the relative quality and the absolute quality can be roughly represented by

$$q_A(x, y) - q_R(x, y) = q_0 - q'_0 \approx \Delta q_0 \quad (5)$$

where $q_A(x, y)$ and $q_R(x, y)$ represent the absolute and relative quality with quantization scale x and bandwidth y , respectively; Δq_0 denotes the encoding quality difference when comparing the undamaged frame sequence with different reference frame sequences and is approximately constant. To show the correctness of this approach, we repeat the previous experiments and plot the relative quality index (each output compared with the sequence encoded at the same quantization scale instead of $Q = 2$) as the “relative” quality in Fig. 5.

As can be observed, the general trends in the two figures are very similar. Recall that for each bandwidth configuration y , there exists a value x_t such that the absolute quality curve satisfies $q_A(x_1, y) > q_A(x_2, y)$, if $x_1 > x_2 > x_t$. Similarly, for the relative quality curve there exists a value x'_t for each y , such that $q_R(x, y) = 0$ if $x > x'_t$. The only difference is that because we are using the transmitted sequence as the reference for quality assessment, increasing quantization scale will never introduce any (relative) encoding quality loss. Taking the scenario of $y = 2500$ as an example, we find $x'_t = 8$. In general, we found that x'_t is typically greater than but close to x_t . The difference between x'_t and x_t comes from the fact that the optimal quantization scale for absolute quality can sometimes be achieved even if the video stream experiences some packet losses. Specifically, as long as the quality impairment caused by packet loss is negligible compared to the encoding loss associated with increasing Q , the value of Q can remain optimal in terms of absolute quality. However, the differences between x_t and x'_t are both rare and small, so that we can assume $x'_t = x_t$. This means that adaptation based on relative quality measurements can be done by decreasing Q as long as the quality index remains equal to 0.

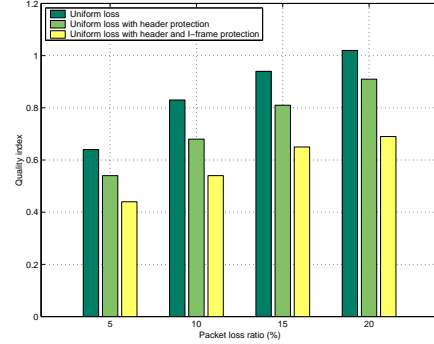


Figure 6: The impact of different data loss on video quality. We test the video sample with uniform loss without protection, uniform loss with header protection, and uniform loss with header and I-frame protection, respectively.

4. QUALITY ENHANCEMENT

We rely on several mechanisms to enhance quality, and in this section we briefly review the main ones. They include error concealment, quality filter, and quality adaptation. Error concealment, while not the main topic of this paper, cannot be ignored as it is often a powerful mechanism to limit the impact of errors. It typically consists of different coding schemes aimed at protecting sensitive data, e.g., frame headers, from losses. Similarly, while quality is a key input to our adaptation scheme, the process used to measure quality, i.e., determine when it has changed, can also play an important role in triggering adaptation, and we briefly describe the method we use for that purpose. Finally, adaptation itself is the main driver behind quality enhancements in the context of this paper, and we describe several parameters it involves and their uses.

4.1 Error Concealment

As an inter-frame encoding method, MPEG encoding typically generates highly structured bit streams. The reference frame (e.g., I-frame) carries more critical information than the other frames (e.g., B-frame and P-frame). A packet loss happened in I-frame can not only affect the quality of the frame itself, but also have the error propagate to other dependent frames throughout the group of pictures (GoP). Besides the existence of differently encoded frames, ancillary data is needed by the decoder for synchronization, identification and characterization of the source information [5]. This data is structured in headers, which can be categorized as sequence header, GoP header, picture header, etc. Because the information contained in these headers is critical for decoding, header loss can be tremendously detrimental to quality. To illustrate the effects of different losses, we simulate a video bit stream with different packet losses. The tested scenarios include: (1) uniform loss, (2) uniform loss with header protection, and (3) uniform loss with header and I-frame data protection. As shown in Fig. 6, although the PLR’s are identical, the resulting quality differs significantly.

Several methods have been proposed for solving this problem. For example, Lee *et al.* [7] proposed an FEC-based multiple description coding method for loss recovery. Feamster *et al.* [1] uses selective retransmission to recover the most important data in the bit stream. Developing sophisticated mechanisms for error recovery or conceal-

ment is beyond the scope of this paper. However, we do realize that this is an important aspect as far as quality is concerned and we used some simple methods to mitigate the impact of loss of those critical data.

We noticed that for some lost headers, the information can be recovered from the previously received headers. For instance, the `picture_coding_type` field in picture header can be inferred from the types of the previous pictures and following pictures, as long as the encoding frame sequence setting (e.g., “IBBP BBP BBP BBI...”) is known. For header loss that is unrecoverable and results in syntax error, we simply skip the associated data until the decoder is resynchronized. This is necessary because decoding from data with syntax error could produce pictures with extremely bad quality. Before synchronization is re-gained, the decoder simply repeats the output of the last decoded frame. In case of frame data loss, we conceal errors in units of “slice.” More specifically, if one slice is missing, the decoder replaces it with the corresponding slice from the previous frame. To keep track of the location of different slices, we encapsulate each of them into a single RTP packet [6]. The RTP header contains the frame number and the slice number that can be used to decide in which frame and where in the frame the slice should be copied to. Through such simple error concealing mechanisms, we found that quality can be enhanced and its variations substantially reduced.

4.2 Feedback Filter

In the current system, quality assessment is conducted every 6 frames³. However, we cannot apply these results directly to control the encoder, as it may result in too rapid encoding fluctuations. Instead, the quality scores of a number of frames is accumulated to decide whether the quality is good or not. There are mainly two reasons for doing this. First, the isolated quality variation happened only in 6 frames is hardly perceivable to human eyes. The authors of the ITS model suggest that the quality scores should be processed with some temporal collapsing function so as to capture the perceptual impacts. The suggested temporal collapsing interval is 1 to 2 seconds [15]. Second, in order to maintain a stable playback quality, the controller should not be too sensitive to the feedback quality score. Some filtering is needed to balance the controller’s reaction to both the long-term and short-term quality variations.

In our experiments, we use a low-pass filter for this purpose. Specifically, let $s(k)$ denote the quality score of the k th measurement, $q_{est}(k)$ denote the overall quality estimation after k steps, then

$$q_{est}(k) = (1 - \alpha) \cdot q_{est}(k - 1) + \alpha \cdot s(k) \quad (6)$$

where $0 < \alpha < 1$. A large α makes the system more sensitive to transient quality changes, while a small α forces a slower adaptation to quality changes. In our experiments, we observe quality adaptation results while fixing the available bandwidth (congestion level) and changing α . We compute the mean quality of 5-minute frame sequences and use it as the major metric when evaluating the impact

³The reason for choosing this number of frames is because the objective result of the ITS model shows the highest correlation with the subjective result for an S-T region of 8 lines \times 8 pixels \times 6 frames [15].

Table 1: The comparison of different choices of α

α	Quality	\bar{Q}	$dev(Q)$	Bit rate (Kbps)
0.01	0.112	10.225	4.740	804.772
0.05	0.079	9.838	4.127	709.540
0.1	0.062	10.668	3.641	610.097
0.15	0.054	10.224	3.771	664.131
0.2	0.066	10.318	3.510	617.620

of α . Secondly, we measure the mean (\bar{Q}) and deviation ($dev(Q)$) of Q . Lower values of both \bar{Q} and $dev(Q)$ are desirable, since the former means higher encoding quality and the latter means less quality fluctuation. We also measure the average bit rate for each α , which reflects bandwidth utilization. Table 1 shows a set of typical results with different α ’s. In comparison with other values, $\alpha = 0.15$ appears to give the best overall results in our initial experiments. We therefore use this value for the remainder of our experiments. A possible explanation for why a relatively small α gives the best result is because a user usually prefers a more consistent playback situation, albeit at a lower overall image quality, with less frame damage, rather than a varying playback situation with higher quality frames but with more frequent frame damage.

4.3 The Adaptation Algorithm

To design the adaptation algorithm, several important parameters need to be determined. First, we need threshold(s) of quality evaluation to decide when to make adaptation. We currently use two static thresholds: q_{high} and q_{low} . When $q_{est}(k)$ is greater than q_{high} , the quality is considered as bad; when it falls under q_{low} , it is considered as good. Having two different thresholds ($q_{high} > q_{low}$) helps avoiding frequent oscillations of Q . In the following experiments, we choose $q_{high} = 0.2$ and $q_{low} = 0.1$.

Second, the value of Q is usually limited within $[Q_{min}, Q_{max}]$. Because on one hand, as described in Section 2.2, it is unnecessary to decrease Q further when the encoding quality saturates. On the other hand, making Q very large will result in extremely bad encoding quality without changing the frame size or frame rate. With the current configuration, $Q_{min} = 4$, $Q_{max} = 16$.

Third, using a low-pass filter to process the feedback information is by itself not enough to guarantee stable adaptation. Every time Q is changed, the system needs to be observed for a certain period before starting the next round of adaptation. One of the reasons is that there usually are frames generated before the adaptation took effect, which are still stored in the receiver’s playback buffer. These frames still carry outdated information about the encoding and transmission conditions. Therefore, the quality feedback extracted from these frames does not incorporate the effect of the last adaptation decision. The use of an “observation period” ensures that the effects of those frames are flushed out of the output of the low pass filter. Furthermore, for rate increase and decrease phases, the observation periods are different and denoted as k_{inc} and k_{dec} , respectively. Their selection will be explained further in Section 5.

The basic strategy used for adaptation is as follows: when the available bandwidth cannot guarantee an acceptable quality, increase Q to avoid further packet loss; while when quality becomes stable, i.e., very little packet losses are experienced (perceived), decrease Q to

improve encoding quality. In the rate decrease phase, we borrow the idea of AIMD [11], making the average bit rate after adaptation less than or equal to half of the previous value. In the rate increase phase, we decrease Q by 1 in every round of adaptation. The algorithm is as follows:

```

for (;) {
    k ← k + 1;
    qest(k) ← (1 - α) · qest(k - 1) + α · s(k);
    if (k < kinc && k < kdec) {
        continue;
    } else if (k ≥ kdec && qest > 0.2) {
        Q ← xopt : xopt = arg minx {qenc(x) | B(x) ≤ 0.5Bcur};
        k ← 0;
    } else if (k ≥ kinc && qest < 0.1) {
        Q ← Q - 1;
        k ← 0;
    }
}

```

Here $B(x)$ denotes the average encoding bit rate when $Q = x$, B_{cur} is the current average bit rate; x_{opt} is the optimal value of x which satisfies two conditions: (1) the corresponding bit rate is no larger than half of the current value; (2) the encoding quality (q_{enc}) is the best among all values of x satisfying (1). The relationships between B , x , and q_{enc} are defined by equation (1) and (2), and the parameters in these equations are measured at run-time. The algorithm starts with $Q = 8$.

5. EXPERIMENTAL RESULTS

We conducted a series of experiments between two sites located at UPenn and UC Irvine, respectively. The path between these two sites goes over the Internet2 network, and therefore rarely experiences continuous bandwidth shortage. However, bursty packet losses are observed from time to time. This configuration is, therefore, used to test the tolerance of our adaptive system to bursty losses. In order to test performance of the scheme under relatively long-term bandwidth fluctuations, we used a traffic generator to introduce cross traffic on the data path. The results in the remainder of this section are for either of these two configurations.

5.1 Dealing With Short-Term Losses

In an under-utilized network like the Internet2, congestion happens mostly at a very short time scale. As a result, video quality experiences short-term and transient disturbances. Such disturbances, unless long enough to become annoying to the viewer, should not cause a rate decrease.

The lowpass filter described earlier handles this situation reasonably well. In Fig. 7, we provide results collected for a 320-second video transmission from UCI to UPenn. The cross traffic on this path introduces bursty loss. However, none of the loss events contains more than 20 contiguous or near contiguous lost packets. The four diagrams, from top to bottom, represent the number of lost packets, the adaptation of Q , the variation of the transmission rate, and the measured/estimated quality, respectively. It should be mentioned that both packet losses and bit rate are collected for a period of 1 sec-

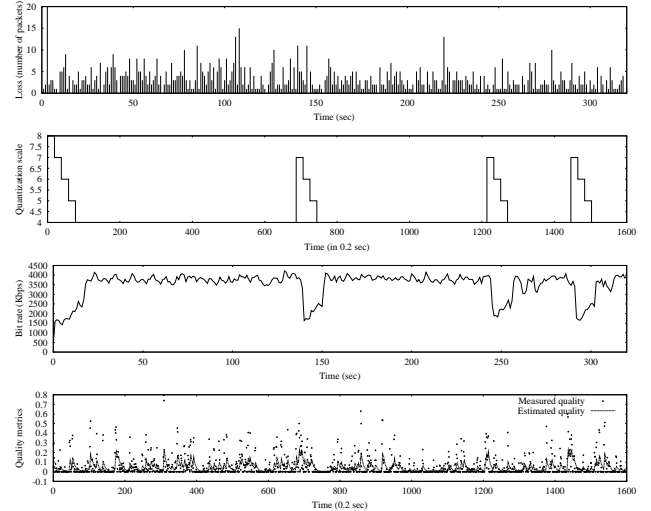


Figure 7: The behavior of the adaptation algorithm under bursty loss condition: the packet loss process, the adaptation of quantization scale (Q), the variation of encoding bit rate, and the video quality (from top to bottom).

ond, while the other data are recorded over periods of 0.2 seconds⁴. The measured quality in the last diagram is $s(k)$, and the estimated quality is $q_{est}(k)$, both are relative quality results. One can observe from the figure that most packet losses did not affect quality very much. Among all the detected loss events, only three of them had substantial impacts on quality and triggered the adaptation of Q , which happened at 135 seconds, 242 seconds and 283 seconds, respectively, after the transmission was started. Note that if a standard rate-based adaptation mechanism had been used, each one of those loss events would have caused a rate back-off, which is unnecessary from the application’s perspective.

5.2 Dealing With Long-Term Bandwidth Variations

Besides having a short-term bursty nature, the Internet traffic also exhibits long-term variations [17] [9]. Our adaptation algorithm needs to be able to deal with such a situation as well. In order to create an experimental setting where such long term variations are present, we use a traffic generator to periodically inject cross traffic into the transmission path, and we observe the reactions of the adaptation mechanism.

Obviously, if we use the overall quality of the received video as a metric, the optimal adaptation algorithm depends on the bandwidth variation pattern. However, the purpose of our experiments here is not to find out the “best” algorithm, but to show how tuning some specific parameters will affect the quality.

Among all the tunable parameters, k_{inc} and k_{dec} are of our particular interests. k_{inc} and k_{dec} control the length of the “observation periods” in the rate increase phase and decrease phases, respectively.

⁴This is the length of a single measurement. 6 frames last 0.2 seconds for the tested video with frame rate of 30 frames per second.

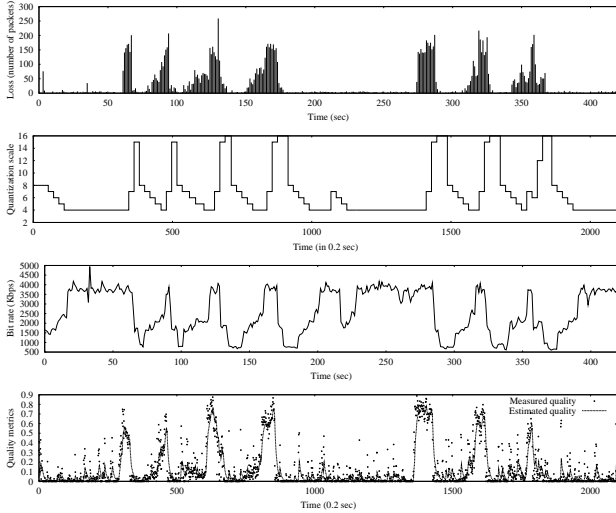


Figure 8: The behavior of the adaptation algorithm under varying bandwidth condition with $k_{inc} = k_{dec} = 15$: the packet loss process, the adaptation of quantization scale (Q), the variation of encoding bit rate, and the video quality (from top to bottom).

We first tested a configuration where $k_{inc} = k_{dec} = 15$. Recall that each measurement contains 6 frames (0.2 seconds). Therefore, both observation periods are 3 seconds in this case. In other words, if the quality is consistently bad for 3 seconds, the transmission rate will be reduced, while if the quality is consistently good for 3 seconds, the sender will try to increase the encoding resolution. The available bandwidth alternates between 3 Mbps and 5 Mbps (5 Mbps to 3Mbps at 65 seconds, and back to 5Mbps at 175 seconds, then back to 3 Mbps again at 225 seconds). The results are shown in Fig. 8. For example, when the bandwidth drops at 65 seconds, the adaptation algorithm reacts quickly by adjusting the value of Q from 4 to 7 and then to 15. Since the average bit rate at $Q = 15$ is less than 3 Mbps, the sender attempts to increase encoding resolution slowly (e.g., at 70 seconds, 100 seconds, and 145 seconds). This initially results in quality degradation after the rate decreases, and then the quality slowly becomes stable again. Considering the overall quality pattern, there are still quite a few “bad” quality periods during the playback.

We then keep k_{dec} unchanged while increasing k_{inc} to 30. We repeated the same experiments as before, except that the changes in available bandwidth took place at 45 seconds (decrease), 120 seconds (increase), and 210 seconds (decrease), respectively. Because of the new value of k_{inc} , observation period before rate increase is doubled. Namely, the sender is more “cautious” about increasing the bit rate. As a result (see Fig. 9), the quality is well maintained when the bit rate is less than available bandwidth. This helps provide the user with a playback without frequent quality oscillations. From the transition process of Q , we can tell that such stability in quality is achieved by sacrificing some encoding resolution (bandwidth utilization). To compare the effects of both configurations, we measured the “absolute” quality of the two received video streams against the same reference. The result shows an average quality improvement of 0.1 during the phase of lower bandwidth with the

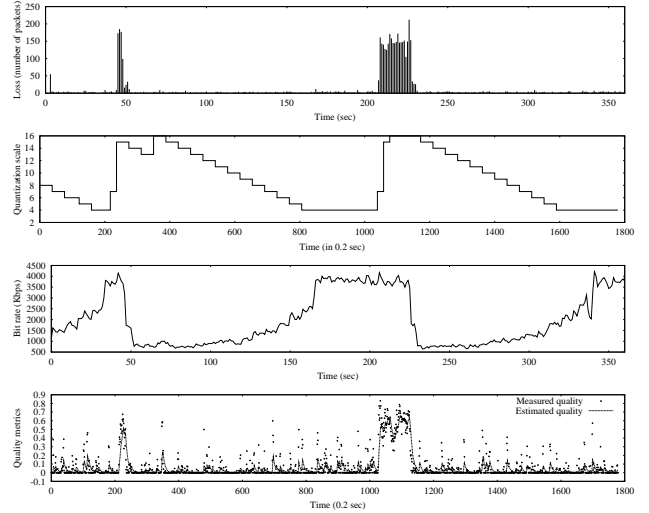


Figure 9: The behavior of the adaptation algorithm with the same configurations as in Fig. 8, except for $k_{inc} = 30$ and $k_{dec} = 15$.

second configuration.

In summary, the experiments with long-term bandwidth variation suggest that to improve quality, we should *decrease rate quickly when congestion happens, and increase rate cautiously when quality becomes stable again*.

6. DISCUSSION

Quality-based adaptation deals with the mismatch between the application’s requirements and the network conditions from an application’s perspective. In other words, changes in application’s behavior are triggered in response to the impact that variations in network resources have on application level quality. A possible criticism of such a scheme is that it may fail to be compatible with other network-based congestion control schemes. Floyd *et al.* [2] proposed that end-to-end congestion control mechanisms should be “TCP-friendly”, i.e., a flow should not use more bandwidth than the most aggressive conformant TCP implementation would under the same circumstances. Our adaptation scheme cannot always satisfy this requirement. For instance, the sending rate does not necessarily back-off when losses occur, while TCP usually does so. Nonetheless, one should also realize that this scheme may not be as aggressive as TCP when extra bandwidth is available. How such a behavior will affect other TCP flows in the long term is an open issue. Meanwhile, our adaptation does nevertheless exhibit some “friendliness” and would avoid congestion collapse [2], because excessive packet losses will ultimately result in quality degradation that will be followed by rate reduction. We believe that the danger of congestion collapse comes primarily from entirely unresponsive flows, like many of today’s commercial video applications, and not from flows employing different adaptation mechanisms. Furthermore, we argue that TCP-friendliness should not be considered as the only possible criterion for measuring a flow’s responsiveness.

In real applications, implementing quality measurement and evaluation may not be always feasible. Although the information trans-

mitted between the sender and the receiver is limited, the process of feature extraction requires intensive computations, especially for high resolution video frames. In a scenario where a server is performing live streaming of different videos to multiple clients, the server can easily become overloaded if required to handle such computations. Therefore, our scheme may not be suitable for all configurations and simple, albeit less efficient, methods of performing quality-based adaptation need to be explored. The simplest one is to allow the viewer to directly interact with the streaming process and adjust video resolution according to his/her perception (in this case the adaptation becomes subjective quality based). However, we believe that our implementation provides a benchmark against which other possible adaptation algorithms can be evaluated.

In addition to its intrinsic limitations, there are other limitations that are specific to the current implementation of our quality adaptation scheme. In particular, the current implementation relies on a VBR encoding card with tunable quantization parameters. This is not a typical configuration for most Internet applications. However, this is only for demonstration purposes, and other coding schemes such as layered encoding [10] and multiple description encoding can be used for the same purpose.

7. CONCLUSIONS

In this paper, we have discussed the viability of quality-based adaptive video transmission over the Internet. A simple analysis of the relationship between quality and network/coding factors was also provided. Based on this analysis, we designed a video adaptation scheme that relied on the tunability of MPEG-2 coding. We investigated the performance of our adaptation algorithm with the help of a prototype system built on top of the real network. The behavior of our algorithm in face of both long-term and short-term bandwidth variations was investigated. Preliminary experimental results showed that quality-based adaptation, which deals with network congestion from an application's perspective, can help enhance video quality. Although this is by no means a complete study of all the issues involved with real-time video streaming and adaptive quality control, the work described here is a first step towards integrating application QoS into a networking environment.

We plan to extend our work in several directions. First of all, we are interested in exploring other quality adaptation mechanisms, e.g., layered coding, object-based coding, etc. Secondly, the relation between video quality and network QoS parameters needs to be further investigated. As stated before, accurately modeling such a relation is difficult. However, it might be feasible to identify simple rules-of-thumb that can help develop more intelligent network mechanisms, such as application specific routing, content delivery network, etc., in support of video applications.

8. ACKNOWLEDGEMENTS

We are grateful to Stephen Wolf and Margaret Pinson at the Institute for Telecommunication Science for providing us with the VQM software tool and for inspiring discussions. We also thank Xiunan Xuan at the University of California, Irvine for helping us in configuring the testbed and in collecting experimental results.

9. REFERENCES

- [1] Feamster, N. and H. Balakrishnan. 2002. "Packet Loss Recovery for Streaming Video". In *Proceedings of International Packet Video Workshop*, Pittsburgh, PA, April.
- [2] Floyd, S. and K. Fall. 1999. "Promoting the Use of End-to-End Congestion Control in the Internet". *IEEE/ACM Transactions on Networking*, August.
- [3] Floyd, S.; M. Handley; J. Padhye; J. Widmer. 2000. "Equation-based Congestion Control for Unicast Applications". In *Proceedings of ACM SIGCOMM*, Stockholm Sweden, August.
- [4] Frossard, P. 1998. "MPEG-2 over Lossy Packet Networks: QoS Analysis and Improvement". Technical report, EPFL, Switzerland.
- [5] Haskell, B.G.; A. Puri; A. N. Netravali. 2000. *Digital Video: An Introduction to MPEG-2*. Kluwer Academic Publishers.
- [6] Hoffman, D.; G. Fernando; V. Goyal; M. Civanlar. 1998. "RTP Payload Format For MPEG-1/MPEG-2 Video". *RFC 2250*.
- [7] Lee, K.-W.; R. Puri; T. Kim; K. Ramchandran; V. Bharghavan. 2000. "An Integrated Source Coding and Congestion Control Framework for Video Streaming in the Internet". In *Proceedings of IEEE INFOCOMM*, Tel Aviv, Israel.
- [8] Lu, X.; R. Morando; M. El Zarki. 2002. "Understanding Video Quality and Its Use in Feedback Control". In *Proceedings of International Packet Video Workshop*, Pittsburgh, PA, April.
- [9] Paxon, V.. 1999. "End-to-End Internet Packet Dynamics". *IEEE/ACM Transactions on Networking*, June.
- [10] Rajaie, R.; M. Handley; D. Estrin. 1999. "Quality Adaptation for Congestion Controlled Video Playback over the Internet". In *Proceedings of ACM SIGCOMM*, Cambridge, MA, August.
- [11] Rajaie, R.; M. Handley; D. Estrin. 1999. "RAP: An End-to-End Rate-based Congestion Control Mechanism for Realtime Streaming in the Internet". In *Proceedings of IEEE INFOCOMM*, Toronto, Canada.
- [12] Verscheure, O.; P. Frossard; M. Hamdi. 1999. "User-Oriented QoS Analysis in MPEG-2 Video Delivery". *Journal of Real-Time Imaging*, October.
- [13] Wolf, S. and M. Pinson. 1998. "In-Service Performance Metrics for MPEG-2 Video Systems". In *Proceedings of Measurement Techniques of the Digital Age Technical Seminar*, Montreux, Switzerland.
- [14] Wolf, S. and M. Pinson. 1999. "Spatial-Temporal Distortion Metrics for In-Service Quality Monitoring and Any Digital Video System". In *Proceedings of SPIE International Symposium on Voice, Video, and Data Communications*, Boston, MA.
- [15] Wolf, S. and M. Pinson. 2001. "The Relationship Between Performance and Spatial-Temporal Region Size for Reduced-Reference, In-Service Video Quality Monitoring Systems". In *Proceedings of SCI/ISAS*.

- [16] Wu, D.; T. Hou; Y.-Q. Zhang. 2000. "Transporting Real-time Video over the Internet: Challenges and Approaches". In *Proceedings of the IEEE*, December.
- [17] Zhang, Y.; N. Duffield; V. Paxson; S. Shenker. 2001. "On the Constancy of Internet Path Properties". In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, San Fransisco, CA, November.