The Impact of Network Performance on Perceived Video Quality in H.264/AVC

Arum Kwon[†], Jin Xiao[*], Sin-seok Seo[†], James Won-Ki Hong[†*], Raouf Boutaba[*‡]

[†]Dept. of Computer Science and Engineering, POSTECH, Pohang, Korea
email: {arumk, sesise, jwkhong}@postech.ac.kr

[*]Division of IT Convergence Engineering, POSTECH, Pohang, Korea
email: jinxiao@postech.ac.kr

[‡]David R. Cheriton School of Computer Science, University of Waterloo, Canada
email: rboutaba@cs.uwaterloo.ca

***Abstract*—Multimedia services have become a dominant part of the network and content provider's service portfolio. A formal modeling methodology for video quality assessment not only affords the providers a clear cause-effect management view of their services, but also helps to guide management and planning operations. We examine the relation between network performance and perceived video quality through VIDAR, a comprehensive VIDeo quality Analyzer in Real-time. VIDAR links network performance to objective frame quality (eSSIM), and modify eSSIM values by subjective filters. Through experiments, we show that VIDAR helps providers to better understand and assess the impact of the network performance on perceived video quality.**

***Index Terms*—video quality management, H.264, QoE**

I. Introduction

Multimedia services have become a dominant part of the network and content provider's business portfolio [1][2]. Due to the best-effort nature of IP networks, network performance is known to fluctuate over time. Multimedia service is particularly sensitive to network QoS parameters, more specifically to loss, delay and jitter. Thus, effective and efficient video quality management is of paramount importance to the providers today. Unlike network QoS metrics, video quality metrics can be difficult to model and to quantify. This is in part due to the many factors that jointly influence video quality (network transport, video encoding/decoding process, compression, packetization techniques, error resilience algorithms, video content, etc.). It is also in part due to the inherent subjective nature of human perception and experience. Human vision system is highly non-linear and viewer's mean opinion score (MOS), an established evaluator of user experience, is also known to be non-linear and difficult to capture without explicit user feedback.

Nevertheless many work exist today that deal with some aspects of video quality assessment. In large, they can be categorized as *white-box* or *black-box* approaches. In a white-box approach [3][4], key indicator metrics are computed. In a black-box approach [5], the relations between lower level performance metrics (such as network QoS) and the higher level user experience (MOS) are estimated through experiments. In comparison, a white-box approach has the advantage of granting a clear and precise understanding of the correlational causality among metrics. However, it can also be computationally expensive. A black-box approach is simple to implement and is good in capturing the general performance trends, however it does not precisely capture the causality among metrics that is crucial to enable effective management operations.

In this paper, we present VIDAR (VIDeo quality Analyzer in Real-time), an analytical framework for real-time video quality monitoring and analysis (Section III). We follow a white-box approach. To this end, we first investigate the correlation between network QoS and objective video frame quality. In this paper, we focus on packet loss. Although it is known that predictive packet loss can be used to estimate video signal distortion [3], there is no direct correspondence to user perception [6][7]. We use Structural Similarity Index (SSIM) [8] for this reason as it better models the human vision system. Because SSIM computation is full reference based (Section II), we propose a Reversed Reduced Reference (R3) model (Section IV) that not only provides good real-time SSIM estimation (eSSIM) with minimal communication overhead, but also preserves the correlational causality between network QoS and video quality. As the basis of our investigation, we choose H.264/AVC [9] as the video technology. Because SSIM is not subjective and is specialized for image assessment, we propose Vidi model to correlate eSSIM to perceived video quality (Section V). Our Vidi model considers the subjectivity in human perception over both spatial and temporal observations. Through experiments (Section VI), we show the effectiveness of VIDAR and report on the impact of network performance on perceived video quality.

II. Related Work

Video quality assessment approaches can be classified based on the need for source- and client-side videos as: full-reference

978-1-4673-0269-2/12/$31.00

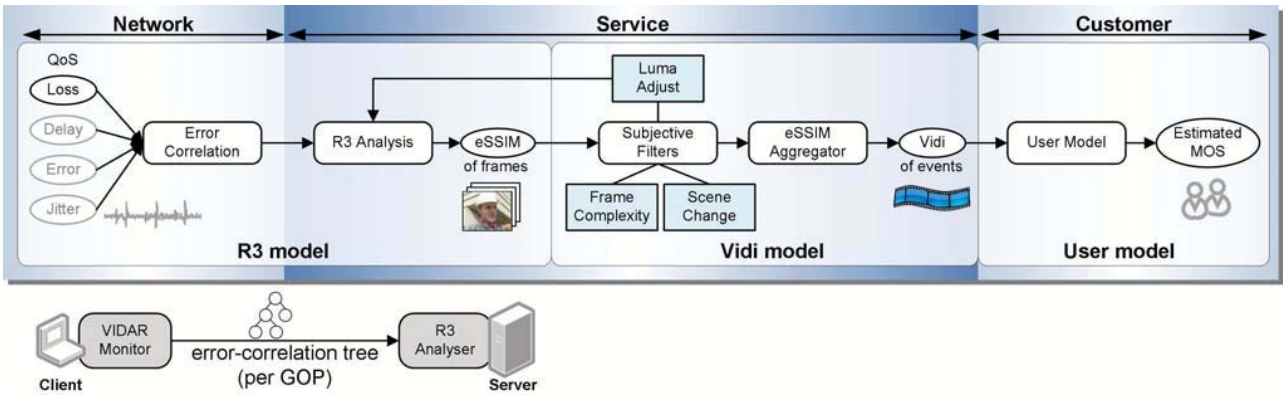


Fig. 1. Process Flow of VIDAR Framework

(FR), reduced-reference (RR), and no-reference (NR). FR approaches require access to both the source video and the client-side video, while RR approaches extract some features from the source video to aid in computation. NR approaches only require access to the client-side video. In terms of suitability for real-time implementation, NR approaches are the best, but in terms of accuracy, they are the worst.

Among the different objective metrics used to assess video quality, Peak Signal-to-Noise Ratio (PSNR) / Mean Squared Error (MSE) is a popular choice. It is easy to compute and has good RR and NR estimations [10]. It has been reported [6][7] that PSNR/MSE does not match well with human perception due to its bottom-up design. SSIM [8] is another popular metric that compares the structural similarity between two images. It is shown to be a good metric for image assessment and has since been extended to video quality assessment [4]. Because SSIM does not consider temporal relations, its extension to video quality assessment remains ineffective. Some work on RR models of SSIM for image quality assessment [11][12] have been proposed, but they require substantial feature information and thus are not suitable for analyzing large volumes of data such as a video stream. Video Quality Model (VQM) [13] is designed for video quality assessment. Although it shows good correlation with perceived video quality, it is not suitable for real-time implementation due to high computational complexity and information processing overhead.

MOS is the de facto metric for evaluating user experience. Extensive user testing is typically conducted over large sets of user experiments. Some existing work try to estimate MOS from the objective video quality measure. MintMOS [5] estimates MOS in a black-box approach. They construct QoE (Quality of Experience) space based on a survey of MOS values under different network conditions and application parameters. In operation, MOS is estimated by finding the closest fit based on distance. Their evaluation shows good result with a small parameter set. Considering the large parameter space of network performance, video encoder/decoder behaviours, user preferences, etc., collecting a reasonable data set can become intractable. In [3], a white-box approach is used to estimate PSNR. The work studies the impact of packet loss on video quality as measured by PSNR. They proposed the rPSNR metric which does not rely on codec parameters and thus is computationally inexpensive for real-time monitoring. In our work, we relate network performance to estimated SSIM (eSSIM), and use the subjective Vidi model for mapping eSSIM to perceived video quality. The work of [14] qualitatively assess the impact of losing different frame types on perceived video defects, however no formal correlations are presented. Our observations confirm their results.

III. Overview of the VIDAR Approach

In this section, we present an overview of VIDAR, a comprehensive video quality analyzer in real-time. VIDAR takes as input the network QoS conditions observed at the client side, and estimates their impact on the quality of video frames, the perceived video quality, and user experience. It considers key parameters across multiple management layers including varying network conditions (Network layer), encoder/decoder algorithms and error recovery techniques (Service layer), video content (Service layer) and viewer subjectivity (Customer layer). As shown in Figure 1, VIDAR is comprised of three models: the R3 model, the Vidi model, and the user model. In this paper, we only deal with the R3 model and a part of the Vidi model. The rest of the Vidi model and the user model will be investigated in the future work.

The R3 model relies on a lightweight client-side monitor (the VIDAR monitor) and a server-side R3 analyzer (Section IV). The VIDAR monitor is a modified video decoder that can be used in place of the client decoder or as a standalone monitor that sniffs the client video stream. Its aim is to collect and generate correlated error logs of defective video frames in the form of error-correlation trees, and to send this information to the server-side R3 analyzer. Although we focus only on network packet loss in this paper, we observe that many other network defects such as excessive delays and packet errors produce similar defects at the application layer in which the VIDAR monitor operates. More specifically, video data that arrives later than the scheduled play time are counted

as losses, and corrupted packets that cannot be corrected are also counted as losses. The R3 analyzer computes estimated SSIM (eSSIM) for each distorted video frame. It is a reversed RR model in that existing work on estimating SSIM requires access to the client-side distorted video, while our R3 model requires only access to the source video at the server side. This design has three salient advantages: 1) the amount of information transferred between the client and the server is minimized; 2) client side can be kept lightweight, ideal for power-constrained mobile devices; 3) better preservation of network-video causalities. SSIM is chosen as the key objective indicator metric because it models the human vision system better than signal-based metrics such as PSNR.

The Vidi model is a subjective model that generates subjective eSSIM values and relates eSSIM to events of perceived video defects. The design rationale behind this model is as follows: a good estimation of image quality is not sufficient to evaluate perceived video quality on its own. This is because human perception of video is temporal and subjective. For instance, it is known that minor to moderate defects in dark scenes are much less noticeable to the viewers [4]; busy scenes can distract the viewers from some defects; and the residual image can hide defects in frames immediately following a scene change [15]. In our experiments, we've also observed that very short bursts of distortions in video are only registered as quick flashes to the viewer. Our Vidi model takes into account these subjective factors through the use of subjective filters: the luma adjust filter, the frame complexity filter and the scene change filter. These filters modify the eSSIM values and help the eSSIM aggregator in combining defective video frames into events of video defects. The subjective filters will be explained in Section V, and the process of eSSIM aggregator and the user model will be explained in the future work. To experiment with our models, we have chosen VLC as the application for video streaming and playback. Our source videos are encoded in H.264/AVC and decoded with FFmpeg decoder.

IV. Mapping Network Performance to SSIM: R3 Model

A. Error-Correlation Tree

At the client side, a lost network packet may affect multiple slices. Since macroblock is the basic unit of image processing, it is ideal to track error correlation at the macroblock level. However, this level of inspection requires decoding the video data which is quite expensive in real-time. Therefore, we choose to work at the slice level, where only the video headers need to be parsed. For ease of discussion, we denote a slice containing lost data as a *lost slice*. Some strategies are employed by the client side to recover a lost slice, depending on its frame type and the severity of the loss across an entire frame. I-, P- and B-frames are used in H.264/AVC. Because P- and B-frames rely on their reference frames for decoding, a lost slice of a P- or B-frame is reconstructed from its reference slices. As I-frames do not have dependency on other frames, a lost slice of an I-frame is reconstructed from its spatially neighboring slices in the same I-frame. If a frame contains too many lost slices, it may be discarded. As we observe through experiment, this condition is more frequent when packet loss is bursty. How a discarded frame is treated is also decoder dependent. The decoder may choose to skip the frame entirely during playback (generally the case with a single frame discard) or may choose to replace the discarded frame with its previous frame in play time order. Since I-frames and P-frames can serve as reference frames in the same Group of Pictures (GOP), the error caused by a lost slice in an I- or P-frame will propagate throughout the GOP.

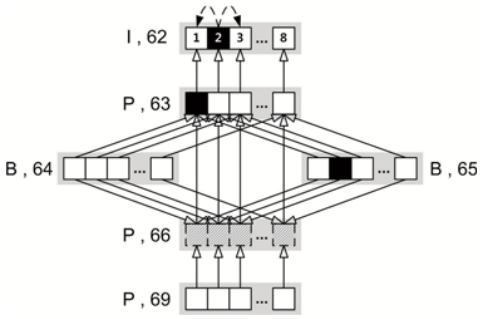


Fig. 2. An Example Error-Correlation Tree

The impact of a network packet loss is frame type dependent, which is in turn video content and codec dependent. This is a strong motivation for preserving their correlations during analysis. Fortunately, given the deterministic nature of the decoding and error recovery process, it is possible to represent the error correlation information of a GOP as an error-correlation tree (Figure 2). Error-correlation tree is recorded as a log file, and is transmitted to the server side. Each node on the tree records a slice. Directed edge leading away from the slice depicts inter-frame dependencies. Solid edge shows dependencies of reference frames. Dashed edge shows reconstruction dependencies. The error-correlation tree is rooted at the I-frame. Each node contains: the frame and slice ID, type, offset, and status. The status are as follows:

- *slc_ok*: the slice itself is healthy. It may still be subject to error propagation.
- *slc_drop*: the slice is dropped and skipped during playback.
- *slc_dup*: the slice is replaced entirely by the previous frame in play time order.
- *slc_frm_ref*: the lost slice is reconstructed from its reference slices.
- *slc_frm_oth*: the lost slice is reconstructed from non-reference slices (error recovery algorithm).

B. Estimating SSIM

The SSIM value of each frame is computed based on the error-correlation tree. Two factors contribute to the distortion of a video frame: slice reconstruction and error propagation. To estimate the error of slice reconstruction, we follow the intuition that a lost slice is reconstructed based on some reference

information, thus the more similar the reference information is to the lost data, the less distortion the resulting reconstruction contains. Therefore, we can leverage the strength of SSIM in comparing the structural similarity of the missing slice and the reference slice and use it as the basis of estimation. For error propagation, because SSIM is a normalized metric, the severity of error does not drift significantly across a GOP, therefore we carry errors across linearly. The standard computation of SSIM is given in [8]. The value of SSIM is bounded between 0 and 1, where 1 indicates a perfect match. Our estimated SSIM (eSSIM) computation is only performed for frames with lost slices and frames affected by error propagation since the eSSIM value of a healthy frame is 1.

Instead of computing the SSIM value over an entire frame, we constrain our sampling window to the slice of interest and its reference slice. This modification significantly reduces the computational cost of SSIM. Let $SSIM_{ms}(I, I_r)$ denotes the SSIM computation between a missing slice I of an I-frame and the reference slice I_r that is used for reconstruction. Then the eSSIM between the original I-frame I and the distorted I-frame I' is computed as:

$$eSSIM(I, I') = 1 - Q_3 \times \frac{\sum_{ms}(1 - SSIM_{ms}(I, I_r))}{eSSIM(I_r, I'_r) \times S_n}$$

I_r is either the spatial neighboring slices in the same I-frame or the corresponding slice in the previous frame (in play time order) that are used for reconstruction. S_n is the number of slices in the frame I. Q_3 denotes the reconstruction error of the decoder algorithm. In the case of FFmpeg which does not perform error concealment for I-frames, we find $Q_3 = 1.55$ through experiments. The computation of eSSIM for a P-frame consists of a reconstruction term and an error propagation term:

$$\begin{aligned} eSSIM(P, P') &= 1 - Q_2 \times \frac{\sum_{ms}(1 - SSIM_{ms}(P, P_r))}{eSSIM(P_r, P'_r) \times S_n} \\ &\quad - \frac{1 - eSSIM(P_r, P'_r)}{[SSIM(P, P_r)]} \times \frac{\Phi}{S_n} \end{aligned}$$

The reconstruction term is normalized based on the SSIM condition of the reference frame $eSSIM(P_r, P'_r)$. Φ is the number of healthy slices in a frame (i.e., $\Phi = S_n - |ms|$). Q_2 is the reconstruction parameter that reflects the outcome of the decoder's reconstruction algorithm. For FFmpeg that uses rudimentary error concealment schemes, we find $Q_2 = 1.1$. $SSIM(P, P_r)$ is conditional in that it is only used if P is *long-range dependent* on P_r. We say frame A is long-range dependent on frame B if and only if A's reference is missing and thus A becomes dependent on the missing reference's reference, B. In Figure 2, *Frame 69* is long-range dependent on *Frame 63*. The computation of eSSIM for B-frame is more elaborate as its reconstruction depends on two reference frames. We first consider the case where both reference frames B_{r1} and B_{r2} are present:

$$eSSIM(B, B') = 1 - reconstruct - error\_prop$$

$$\begin{aligned} recostruct &= Q_2 \times \Big(\frac{\sum_{ms}(1 - SSIM_{ms}(B, B_{r1}))}{eSSIM(B_{r1}, B'_{r1}) \times 2S_n} \\ &\quad + \frac{\sum_{ms}(1 - SSIM_{ms}(B, B_{r2}))}{eSSIM(B_{r2}, B'_{r2}) \times 2S_n} \Big) \end{aligned}$$

$$error\_prop = \frac{\Phi((1 - eSSIM(B_{r1}, B'_{r1})) + (1 - eSSIM(B_{r2}, B'_{r2})))}{2S_n}$$

In the case when B-frame has only a single reference, we compute its eSSIM similar to the P-frame computation:

$$\begin{aligned} eSSIM(B, B') &= 1 - Q_2 \times \frac{\sum_{ms}(1 - SSIM_{ms}(B, B_{r1}))}{eSSIM(B_r, B'_{r1}) \times S_n} \\ &\quad - \frac{1 - eSSIM(B_{r1}, B'_{r1})}{SSIM(B, B_{r1})} \times \frac{\Phi}{S_n} \end{aligned}$$

Here the factor $SSIM(B, B_{r1})$ is not conditional when computing the error propagation term. This accounts for the drift bias B sustains as it is reconstructed from a single reference frame. If both reference frames of a B-frame are missing, the B-frame is discarded.

V. Relating Frame Quality to Perceived Video Defects: Vidi Model

The Vidi model takes eSSIM of frames as input and modify it using subjective filters. As outlined in Section III, the design of our Vidi model is motivated by the observation that human perception is subjective both in spatial and temporal dimensions. Frame quality metric eSSIM values are modified by the subjective filters. In the case of luma adjust filter, the modification is applied to the eSSIM computation process itself; while in the case of frame complexity and scene change filters, the modification is applied to the eSSIM values of the frames.

A. Luma Adjust Filter

It is known that human perception is luminance dependent. Low to moderate levels of video distortions are not as perceivable in dark scenes as in well-lit scenes. Luminance adjustment techniques can be used to incorporate this phenomenon. Our luma adjust filter is modified based on the technique of [4]. It augments the R3 model in which eSSIM is computed. More specifically, SSIM computation is carried out over a series of sampling windows. In the standard computation, the SSIM value of each sampling window contributes the same weight to the SSIM value of the frame. With luma adjust, we weigh the SSIM value of each sampling window based on a w_j factor ($0 \leq w_j \leq 1$) depending on the luminance of the sampling window. The value of w_j is determined as follows:

If $\mu_x^Y \leq 50$ and $\mu_y^Y \leq 50$,

$$w_j = \begin{cases} 0 & \text{, if } \frac{\mu_x^Y + \mu_y^Y}{2} \leq 40 \\ \frac{(\frac{\mu_x^Y + \mu_y^Y}{2}) - 40}{10} & \text{, if } 40 < \frac{\mu_x^Y + \mu_y^Y}{2} \leq 50 \end{cases}$$

Else, $w_j = 1$.

(a) Low Frame Complexity

(b) Medium Frame Complexity

(c) High Frame Complexity

Fig. 3. Comparison of Distorted Frames with Same SSIM=0.76

B. Frame Complexity Filter

The busyness of a frame has a direct correspondence to how perceivable a distortion can be, especially when the degree of distortion is not severe. Figure 3 illustrates three distorted video frames as generated due to packet loss. They all have the same SSIM value (source vs. distorted) of 0.76, yet we observe significant difference in defect visibility. This effect is more pronounced when motions are involved. Busy scenes in video clips are typically associated with wide area motions or highly complex structures. Accordingly, we introduce a weight factor α_{FC} that moderate the eSSIM value of a frame.

To determine α_{FC}, a good indicator of frame complexity is needed. Frame complexity is a known concept in video rate controllers. Their goal is to estimate the compression rate of input video stream such that efficient rate control algorithm can be implemented. Mean Absolute Difference (MAD) is often used as the frame complexity metric. In their case, MAD is computed over the entire frame to obtain a complexity score. We find that such computation is too coarse for our purpose and does not show a good indication of the level of busyness in a frame. But when we combine MAD with the idea of sampling windows in SSIM computation, we can obtain a good frame complexity indicator given an appropriate choice of window size. To illustrate, the complexity of each frame in Figure 3 are 10.28, 21.22, and 29.52 respectively.

Our computation of frame complexity is as follows:

$$FC = \frac{\sum_{r \in R} \sum_{k \in m,n} |v^Y_{k,r} - \mu^Y_r|}{Rmn}$$

R is the number of samples of size $m \times n$ pixels to be taken in a frame. $v^Y_{k,r}$ is the luminance component of the pixel k in sample window, and μ^Y_r is the mean of luminance in the sample window r. The weight modifier α_{FC} can then be specified as follows:

If $eSSIM \geq 0.60$, $eSSIM = \alpha_{FC} \times eSSIM$

$$\alpha_{FC} = \begin{cases} 0 & \text{, if } FC \geq 30 \\ 1 - (FC - 25) \times 0.2 & \text{, if } 30 > FC > 25 \\ 1 & \text{, if } FC \leq 25 \end{cases}$$

C. Scene Change Filter

Human vision system is affected by residual image effect in that if there is a sudden change of scene (e.g., high speed motion, or a scene cut), the first few frames immediately following the scene change are not perceived nearly at the same degree of clarity as the frame before change. It was reported in [15] that immediately after a scene change, even if severe distortions are present in video frames, the viewers do not register any perceived defects. This effect lasts for approximately 0.33 second. This phenomenon is leveraged to achieve better video compression by reducing the image fidelity. In their case, the degradation to image quality is uniform across the frame, while in our case the loss-induced distortions are concentrated locally on a frame. The latter is far easier to be perceived by a viewer, thus we designed a modified version of the scene change filter to take into account this difference.

Scene change detection is accomplished by computing the color difference between two consecutive frames. A number of approaches can be used to realize this. We choose a block-based color histogram approach due to its low computation cost and effectiveness. We compute the scene change factor as follows:

$$SC(f_{i-1}, f_i) = \frac{\sum_{c \in RGB} \sum_{b=1}^{4 \times 4} \sum_{k=0}^{L^c} |H(f_{i-1}, c, b, k) - H(f_i, c, b, k)|}{mn}$$

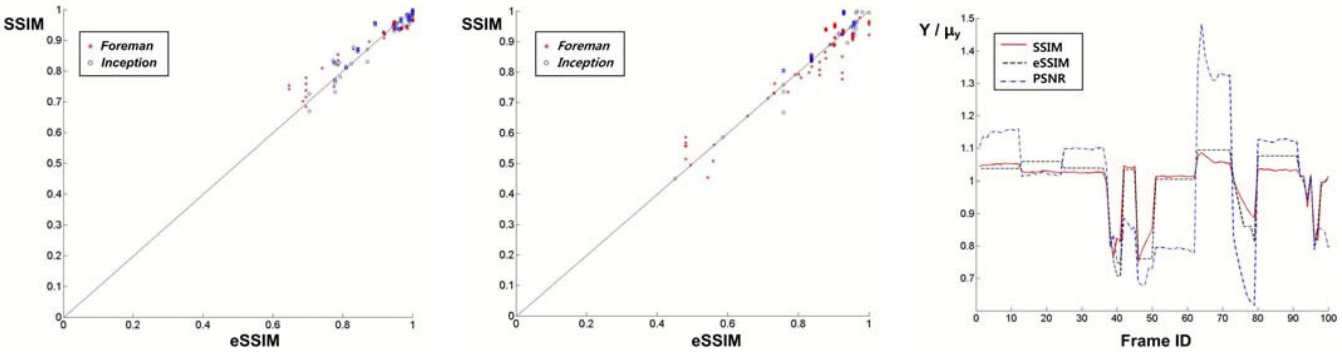
H is the histogram function. In summary, we break up a frame into 16 blocks (4×4). For each block, we construct the color histogram for each of the R, G and B color components over all possible integer values k in range L^C. The sum of differences among the corresponding histograms between two consecutive frames (f_{i-1} and f_i) are then divided by the block size to obtain the scene change value. A frame's weight factor β_{SC} is given below. The threshold values are obtained through experiments. Consider (f_{i-1}, f_i) as a scene change shortly before a frame k is played.

$eSSIM = \beta_{SC} \times eSSIM$

$$\beta_{SC} = \begin{cases} 0 & \text{, if } SC(f_{i-1}, f_i) \geq 3.0 \text{ \& } \textit{Condition A} \\ 0 & \text{, if } 3.0 > SC(f_{i-1}, f_i) \geq 1.7 \text{ \& } \textit{Condition B} \\ 1 & \text{, otherwise} \end{cases}$$

Condition A: k is within 0.17 sec. after frame *i*-1

Condition B: k = *i*



(a) SSIM-eSSIM Bernoulli (b) SSIM-eSSIM GE (c) SSIM-eSSIM-PSNR

Fig. 4. Performance of eSSIM in Bernoulli and GE loss model

VI. Experiment Study

In this section, we report on the experiment studies carried out using VIDAR and show the performance of eSSIM and how network performance affects video quality in H.264/AVC. We illustrate the effects of varying key parameters including loss rate, loss pattern and GOP size.

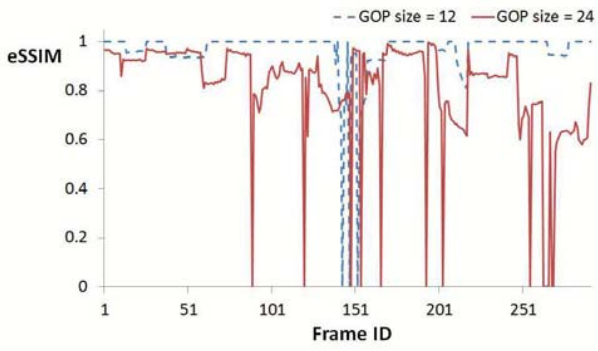
The computation of eSSIM is normalized to [0, 1]. In our experiments, we find that eSSIM is a good estimation of SSIM. Figure 4 illustrates our findings. We have conducted experiments using *Foreman* and *Inception* video clips. Two different types of network error models are used, namely the Bernoulli and Gilbert-Elliott (GE) packet loss models as used in [3]. The Bernoulli loss model exhibits uniform random loss patterns for which we set the loss probability to 2%. Figure 4a shows our eSSIM as computed by R3 model over 200 distorted frames. We observe a strong correlation between eSSIM and SSIM. Similarly, we studied the performance of R3 under the GE packet loss model. It is a bursty loss model for which we set the expected burst length to 1.67 and the loss event probability to 1.93% (p=0.02, q=0.60). Figure 4b suggests that eSSIM is a good estimation of SSIM. Figure 4c plots SSIM, eSSIM and PSNR. We normalized the PSNR value of each frame based on its mean, and normalized the SSIM and eSSIM values of each frame based on the SSIM mean. We took the eSSIM computation that showed the worst correlation among all of the test scenarios: the *Foreman* video under 2% Bernoulli loss model. From the figure, we see that SSIM and eSSIM capture the variations in frame quality over time (the frame IDs are in sequence of play order), while the PSNR shows significant fluctuations. This further confirms that SSIM is indeed a better image assessment metric than PSNR, and eSSIM is a good estimator of SSIM.

To examine the effect of varied network performance on video quality, two source videos are used. *Foreman* video is a reference video containing both a portrait scene and a landscape scene. However, there is no rapid movement or scene cuts. We therefore also use a 10 sec. clip from *Inception* as the second source video. We study two packet loss models: the Bernoulli uniform loss model and the Gilbert-Elliott (GE) bursty loss model. Figure 5 presents the computed eSSIM values of the frames. The frame IDs are in playback order, and an eSSIM value of 0 indicates that the frame is discarded. Figure 5a shows the effect of changing GOP size in *Foreman*. The result is significant even under a moderate 2% uniform packet loss. We see periods of prolonged video distortions when the GOP size is set to 24. This is expected as any distortion to the P- and I-frames tend to propagate over the entire GOP. Figure 5b shows the effect of varying the loss rate of Bernoulli uniform model in *Foreman*. We see that the increase in packet loss is uniformly mapped to distortions across the frames. A loss rate of 4% generates significantly more distortions, and at higher intensity and longer duration. We observe that the distortion grows non-linearly with loss rate. Figure 5c shows the effect of varying burst length in *Foreman*, the expected burst lengths are 1.67 (p=0.02, q=0.60) and 2.2 (p=0.02, q=0.45) packets respectively. We see that H.264/AVC is more sensitive to bursty loss than to uniform loss. Even at a low loss rate (1.93% and 1.91%), the distortion is prolonged and sometimes severe. The number of missing and discarded frames are also significantly higher than that of the uniform loss case. Figure 5d shows the results of *Inception* video under the same GE parameter. Since *Inception* has smaller GOP size due to multiple scene cuts, we observe slightly less frame drops.

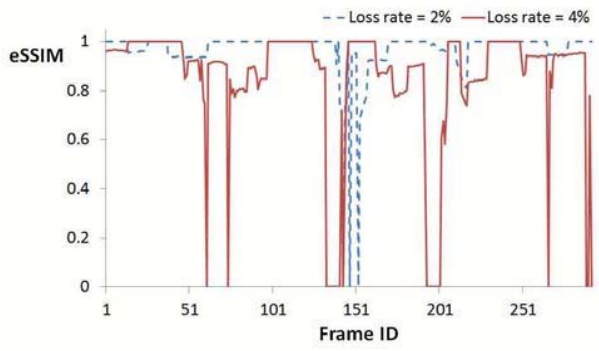
Our examination of eSSIMs under varied network and codec parameters gave us a good overview of the impact of network performance on video defects at the frame level.

VII. Conclusion

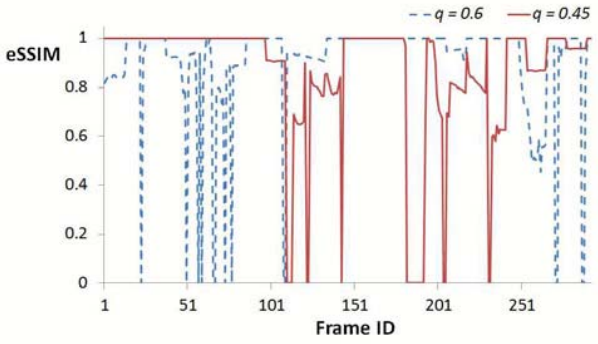
In this paper, we presented the VIDAR framework for video quality assessment. Following a white-box approach for modeling, we correlated the effect of network performance degradations to video frame distortions. An estimation of SSIM (eSSIM) is used as the key indicator metric for this as computed in our R3 model. We modified eSSIM by using subjective filters in our Vidi model. Through experiments, we showed the effectiveness of VIDAR in capturing metric correlations across different management layers and in assessing



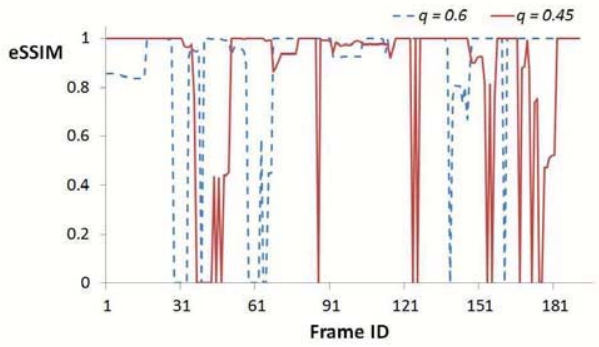
(a) *Foreman* uniform loss 2%, GOP size = 12 and 24



(b) *Foreman* uniform loss 2% and 4%



(c) *Foreman* GE loss $p = 0.02$, $q = 0.60$ and 0.45



(d) *Inception* GE loss $p = 0.02$, $q = 0.60$ and 0.45

Fig. 5. Frame Distortion Under Varied Network Conditions

the impact of network performance on perceived video quality. As future work, we will suggest an event-based video quality metric and elaborate on the user model and validate VIDAR through further experimentation and user studies.

ACKNOWLEDGEMENT

This research was supported by World Class University program funded by the Ministry of Education, Science and Technology through the National Research Foundation of Korea (R31-10100) and the KCC(Korea Communications Commission), Korea, under the "Novel Study on Highly Manageable Network and Service Architecture for New Generation" support program supervised by the KCA(Korea Communications Agency) (KCA-2011-10921-05003).

REFERENCES

[1] Cisco, "Cisco visual networking index: Forecast and methodology, 2008–2013," White Paper, Cisco Systems, July 2009.

[2] C. Labovitz, S. Iekel-Johnson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *ACM SIGCOMM*, August 2010, pp. 75–86.

[3] S. Tao, J. Apostolopoulos, and R. Guerin, "Real-time monitoring of video quality in IP networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 5, pp. 1052–1065, October 2008.

[4] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural distortion measurement," *Signal Processing: Image Communication*, vol. 19, no. 2, pp. 122–132, 2004.

[5] M. Venkataraman and M. Chatterjee, "Inferring video QoE in real time," *IEEE Network*, vol. 25, no. 1, pp. 4–13, January-February 2011.

[6] B. Girod, "Whats wrong with mean-squared error," in *Digital Images and Human Vision*, A. B. Watson, Ed. the MIT Press, 1993, pp. 207–220.

[7] A. M. Eskicioglu and P. S. Fisher, "Image quality measures and their performance," *IEEE Transactions on Communications*, vol. 43, no. 12, pp. 2959–2965, December 1995.

[8] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[9] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[10] A. R. Reibman, V. A. Vaishampayan, and Y. Sermadevi, "Quality monitoring of video over a packet network," *IEEE Transactions on Multimedia*, vol. 6, no. 2, pp. 327–334, April 2004.

[11] A. Rehman and Z. Wang, "Reduced-reference SSIM estimation," in *2010 17th IEEE International Conference on Image Processing (ICIP)*, September 2010, pp. 75–86.

[12] A. Albonico, G. Valenzise, M. Naccari, M. Tagliasacchi, and S. Tubaro, "A reduced-reference video structural similarity metric based on no-reference estimation of channel-induced distortion," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2009, pp. 1857–1860.

[13] M. H. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *IEEE Transactions on Broadcasting*, vol. 50, no. 4, pp. 312–322, September 2004.

[14] J. Greengrass, J. Evans, and A. C. Begen, "Not all packets are equal, part 2: The impact of network packet loss on video quality," *IEEE Internet Computing*, vol. 13, no. 2, pp. 74–82, March-April 2009.

[15] A. R. Reibman and D. Poole, "Predicting packet-loss visibility using scene characteristics," in *Packet Video 2007*, November 2007, pp. 308–317.