On the Accuracy of PSNR and PSQA Video Quality Measures

Samir Mohamed and Gerardo Rubino

Abstract

We have recently proposed a novel, non–intrusive, real–time approach to measuring the quality of video streams transmitted over a packet network. The proposed approach takes into account the diversity of the factors that affect video quality, including encoding parameters and network impairments. The goal of this method is to overcome the limitations of the quality assessment techniques currently available in the literature, such as the low correlation with subjective measurements, or the need to access the original signal, which precludes real–time applications.

Our approach correlates well with human perception, it is not computationally intensive, does not need to access the original signal, and can work with any set of parameters that affect the perceived quality, including parameters such as FEC and the frame rate, which are usually not taken into account in other methods. It is based on the way human observers react to the streams. This fact, together with the very good correlation with real human perception (that is, with subjective testing) explains the name we have given to it: Pseudo-Subjective Quality Assessment (PSQA).

In this paper we compare the performance of PSQA to that of other assessment techniques found in the literature. In a nutshell, using subjective testing results as the reference, PSQA always reaches very high correlation values while PSNR (a popular metric designed for analyzing coding effects) hardly reaches values around 80% (due to network impairments).

Keywords: Packet video, Neural Networks, Real-time video transmission, Video quality assessment, Objective and subjective quality.

I. Introduction

Many real-time video transmission applications over the Internet have appeared in the last few years. We find today: video phones, video conferencing, video streaming, tele-medical applications, distance learning, telepresence, video on demand, etc., with a different number of requirements in bandwidth and in perceived quality. The interest in being able to quantify the quality of perceived video as perceived by a human observer and, if possible, in real time, follows from this reality and from the increasing importance of multimedia traffic over the Internet.

IRISA/INRIA, Campus de Beaulieu, 35042 Rennes Cedex, France. E-mails: {Samir.Mohamed, Gerardo.Rubino}@irisa.fr
The correspondent author is Prof. Gerardo Rubino. Tel. +33 299 847 296, Fax. +33 299 842 529

In [7], we proposed a new methodology to evaluate automatically and, if necessary, in real time, the quality of video transmitted over packet networks , that we propose to call PSQA (Pseudo-Subjective Quality Assessment). We showed that this method has several good properties, namely: Our approach correlates well with human perception, it is not computationally intensive, it does not need to access the original signal, and it can *a priori* work with any set of parameters that affect the perceived quality, including parameters such as the frame rate (when the encoded sequence has lower rate than the original), which are usually not taken into account in other methods. It is based on the use of a Random Neural Network (RNN) [1, 2], which is trained to assess video quality as an average human being.

We give here a summary of the PSQA method. We start by choosing a set of parameters supposed to have an impact on the perceived quality (even if we do not know how exactly this impact occurs. We also choose their ranges out of which quality is trivially bad, and a discretization of them. Then we choose original video sequences. Each of them is distorted after being sent through a network under controlled conditions, which means that the affecting selected parameters (source bit rate, end-to-end loss rate, . . .) take different values (a set of values for the parameters is called a *configuration* in [7]). For each original sequence many distorted versions are built corresponding to many different configurations, such that the state space of all possible configurations is in some sense covered. Next step is to perform a subjective test for each sequence, which receives thus a human evaluation. This evaluations are mapped into the different configurations that were considered to modify the sequences, and the idea is then to train a RNN (a quite special type of Neural Network having very good mathematical properties) to capture that mapping.

The final evaluation tool consists of two modules: the first one collects the current values of the selected parameters in the transmission and the second one uses the trained RNN to give a value to the quality. For more details about this method and its applications, see [7–9]. In the operating mode when integrated into a video system, our tool will act as shown in Figure 1. At the sender side, the video source is encoded, packetized and sent by the transport protocol (e.g. RTP/RTCP) to the receiver. At the receiver, the flow is decoded and displayed to the end-user. The interaction between our tool and the other elements is as follows. The parameters' collector part probes all the working parameters from the encoder, the decoder, the packet network and the transport protocol. Then the trained RNN part evaluates video quality as a function of these parameters. This means that the quality at the receiver can be evaluated continuously, that is, with a very high frequency. If the sender needs feedback from the receiver, the quality value can be sent back by the transport protocol from time to time (for instance, if the RTCP protocol is used, it can be sent every 5 sec.). Observe however that many implementations of RTCP allow the modification of this delay by the application as defined by the RFC standard of RTP protocol stack.

Concerning the PSNR measure and its choice in this comparison paper, it is a widely used metric which basically evaluates the difference between the original sequence before encoding and the received sequence (at the other side of the network) after decoding. Its expression is given below (relation (1)). Recently, other efforts have been made in order to design other techniques that measure this difference between two signals. The most known are ITS [13], MPQM, CMPQM and NVFM [10, 11] metrics. These metrics operate on both the original and the distorted video sequences.

This limitation makes it impossible to work in real time and to include these metrics in designing new mechanisms (rate control or video codecs to take into account the user's perception and the network factors). A second disadvantage is that the obtained results do not always correlate with subjective data. A third drawback is that some of them are very computationally extensive, especially the ones built based on VHS model (some of them cannot be used to evaluate the quality for video sequences of length greater than 1 sec.) [10]. Some of these metrics are designed and optimized basically to consider encoding impairments, but they do not work efficiently when they used in other conditions (ex. distortion due to the transmission over the network). Another observation is that not all the quality-affecting parameters can be considered. For example, the frame rate effect cannot be considered as they compare the original and distorted. This means that both sequences must have the same frame rate and the decoded picture of the processed sequence must correspond to the encoded picture in each frame of the original sequence, otherwise the results will degrade.

In this paper, we show the properties of our PSQA measure against the value given by PSNR in different cases. We show how PSNR can mislead the point when trying to assess the degradation of a sequence after traveling through the network, and at the same time we explain how, *by construction*, the PSQA approach doe not suffer from these drawbacks. We also discuss the properties of the other trials previously mentioned (ITS, MPQM, CMPQM and NVFM). Finally, we describe a tool we developed to studying the way a selected set of parameters affect perceived quality.

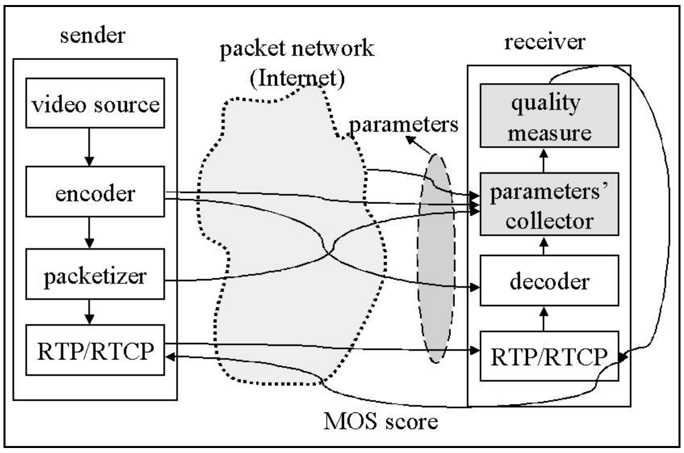


Fig. 1. Operation mode for the tool in real-time video system.

The rest of the paper is organized as follows: In Section II, we describe the platform we used to generate the distorted video sequences and the application we realized to measure video quality. The comparison between the performance of PSQA and PSNR is detailed in Section III. The performance of ITS, MPQM, CMPQM and NVFM metrics is the subject of Section IV. Finally, we conclude our work in Section V.

II. Description of the experimental context

A. Platform Description

In this Section, we describe the platform that we used in order to build the quality evaluator. In addition we use it to build an application that is used to evaluate video quality in real time (see Section II-B). The platform consists of an

H263 [4] encoder, an H263 decoder and a network transport simulator.

1) H263 encoder: The encoder takes as inputs the original video sequence in YUV format and the resolution of the video, which can be any of the standard resolutions or a customized one. The start and the end of the frames to be encoded may be controlled. The size of the search window to detect the motion vectors can also be controlled. The quantization parameter of encoded frames may be fixed to control the output bit rate. However, the encoder supports several types of bit rate control methods: variable and constant bit rates. Thus, some of them override the quantization parameter option. It is known that the common frame sequence of H263 encoding is IPPPPP . . . , where the only frame encoded as I is the first one and all the others are encoded as P frames. The size of I frames may be five times larger than the size of P frames. Hence, to provide smoothing of the output bit rate, the encoder has the ability of varying the quantization parameter of the first I frame separately.

H263 behaves, by default, as a one-layer codec. However, it supports an extra enhancement layer (with the ability of choosing its output bit rate and other options). The encoder also takes the frame rate of the original video sequence as input. The output frame rate can be controlled by allowing the encoder to skip some images between two encoded ones. For example, if the input sequence is encoded at 30 frames/s, by skipping one image between each two encoded images, the output frame rate will be 15 frames/s.

ITU recommendation for H263 [4] defines some basic operations that all the implementations should support. In addition, some other functionalities are defined separately in annexes as options.

2) H263 decoder: The H263 decoder reads the streams encoded by the encoder as well as the control information generated to playback the encoded sequence. Like the encoder, it has several command-line options to fine-tuning its operation. The default operation is displaying the video on the screen, but the decoder has the ability to store the output into a specified file in several formats. Regarding the frame rate, it is possible to specify any value instead of the default (which is the rate of the encoded stream).

3) Network Transport Simulation: To integrate the effect of network conditions on the streams, we used a simulator developed by the TEMICS team in our laboratory [6]. In this tool, several functionalities have been added to the codec in order to simulate its operation over packet network. The encoder generates marks in the output bit streams to indicate the start and end of each packet. In addition, the encoder provides some ways to cope with packet loss effects (error resiliency) [6]. The ability of inserting a header at the beginning of each group of blocks (slice) is allowed to deal with the spatial encoding problem due to loss. In addition, it is possible to force Intra mode coding of macroblocks in certain way in order to avoid the problem of losses due to the temporal encoding. The maximum packet size can be varied to suit the requirements of the network supporting the transmission. When the simulated transmission is supposed to be over the Internet, the default size is set to 536 bytes (this is to avoid the fragmentation of packets between routers). The packetization process is done in conformance with IETF RFC 2429 [4]. The program can also generate packet loss patterns using Gilbert model. Other models of packet loss can be defined. In addition, the decoder has supplementary way to deal with loss distribution through the sequence numbers of packets to be dropped. We further modified the simulator to suit our needs when generating the video sequences, in order to carry out subjective quality tests and to implement the video application described in the next Section.

B. *An Application to Study Video Quality*

Let us describe the application we used to explore the perceived quality of video streams transmitted over the Internet. The application served for the comparisons which main results are descried in this paper, and also for the analysis of the impact on quality of the following parameters: at the source side, the Bit Rate (BR), the Frame Rate (FR) and the Intra to Total Ratio (RA); considering the network, the Loss Rate (LR) and the Consecutive Loss Parameter (CLP) (average size of loss bursts). In an experiment, these parameters can be selected or varied using a GUI. In our current implementation, the LR can be varied from 0 to 10% and CLP goes from 1 to 6. These two variables are used to characterize the loss process in the network. The remaining three parameters are used to control the source encoding. Knowing that each video sequence has a specific maximal $\mathrm{BR}_{\max}$ value, the BR parameter takes one of the following values: $\{1, 0.75, 0.50, 0.25\} \times \mathrm{BR}_{\max}$. The FR parameter takes the following values: 6, 10, 15 and 30 frames/s. Finally, the RA parameter can take one of four values. These values depend on the values chosen for BR and FR.

The application is implemented based on the H263 decoder introduced in Section II-A. It incorporates two operating modes: manual and automatic. Due to space limitation, we show only a screen-dump of the Manual mode in Figures 8. In the manual mode, the five parameters can be changed manually. The network parameters (LR and CLP) can be varied smoothly during video playback. The codec's parameters (BR, FR and RA) are chosen before starting the playback of the video sequence under study. The parameters' collector module probe the values of the parameters regularly, at a predefined interval (currently fixed to 300 ms). The trained RNN module measures the quality based on the values of the parameters. In the automatic mode, the application plays back the video sequences and distorts them from a predefined sets of parameters' values. The goal of this mode is to study the variation of the quality with the parameters (varying one of them and keeping the others fixed).

Another function of the tool is the possibility of measuring the PSNR during the playback of the processed video sequence. We added this feature to allow us comparing the performance of our PSQA measure against that of the standard PSNR one.

For a video sequence of K frames each having NxM pixels with m-bit depth, the Mean Square Error (MSE) is calculated as follows:

$$\mathrm{MSE} = \frac{1}{N.M.K} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{m=1}^{M} [x(i,j,k) - \bar{x}(i,j,k)]^2,$$

where $x(i,j,k)$ and $\bar{x}(i,j,k)$ are the respective values of the pixels at position i,j in frame k, for the original and the distorted sequences. The Root MSE (RMSE) is calculated using $\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$. The PSNR is then defined as

$$\mathrm{PSNR} = 10 \log_{10} \frac{m^2}{\mathrm{RMSE}^2}. \tag{1}$$

Both MSE and RMSE measure the difference between the original and distorted sequences, and thus, the higher the PSNR the closer the two (original and distorted) sequences. However, performing some experiments (for instance, with our tool) rapidly show that the correlation between PSNR and subjective measures is often very bad. In the tool, we use a sliding window to calculate the PSNR. This sliding window can be modified. By default, it is fixed to 10 frames (about 333 ms.).

During playback, the following values are drawn each in a curve:

1) MOS obtained by PSQA (ranges from 1 to 9),
2) PSNR value in dB (ranges from 0 to 50),
3) PSNR mapped to MOS scale (ranges from 1 to 9),
4) LR chosen by the user (ranges from 1 to 10 %),
5) CLP chosen by the user (ranges from 1 to 6).

In the current implementation, we integrated three standard video sequences (namely Stefan, Children, and Foreman). Each of these sequences has different video characteristics and nature. “Stefan” is characterized by fast motion, while “Children” is characterized by slow motion, more spatial redundancy and more colors saturation. “Foreman” sequence is a moderate one, somehow between “Children” and “Stefan”. It should be noted that the BR of the best quality of the encoded versions are 1345, 1200, 1000 KByte/s for Stefan, Foreman, and Children respectively. By using the algorithm described in [7], our application works quite well for all the three video sequences regardless of the variability of the BR from sequence to another.

C. *Mapping PSNR to 9-point ITU Scale*

In this paper, we give examples of cases where PSNR has bad correlation with human perception. The strongest way to do this is to exhibit examples where, changing one parameter (such as the frame rate FR, or the end-to-end loss rate in the network, LR) the behavior of PSNR and PSQA (which follows human perception) are qualitatively different. For instance, consider Figure 2 where, at the left, PSNR in dB is plotted against the Frame Rate FR in fps (for three different values of the loss bursts, CLP parameter). We see that the PSNR decreases in two of the three cases, and oscillates some way in the case of CLP = 4, while humans give increasing MOS when FR increases, for all values of CLP. The same qualitative differences can be seen in Figure 3, where again PSNR in dB at the left and MOS at the right are plotted against FR, now for two different video sequences.

While in general, PSNR is given in dB, it can be simpler to map the dB scale into the MOS one, to make comparisons more clear. For this purpose, we map the PSNR in dB into the ITU quality scale. A possible mapping using the MOS ITU 5-point scale is as shown in Table I, as found in [5].

To build some of the pictures, we used the previous mapping and we calculated a polynomial approximation of it, in order to easily obtain many points to draw curves. For instance, such an approximation is given by the following piece-wise polynomial:

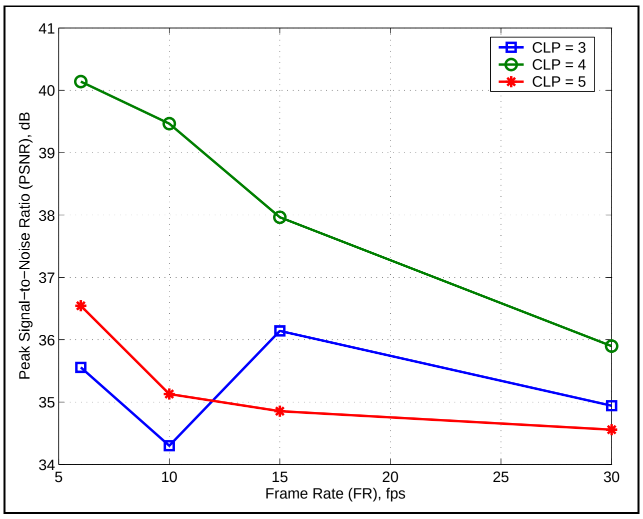
TABLE I

POSSIBLE PSNR TO MOS CONVERSION (SOURCE [5]).

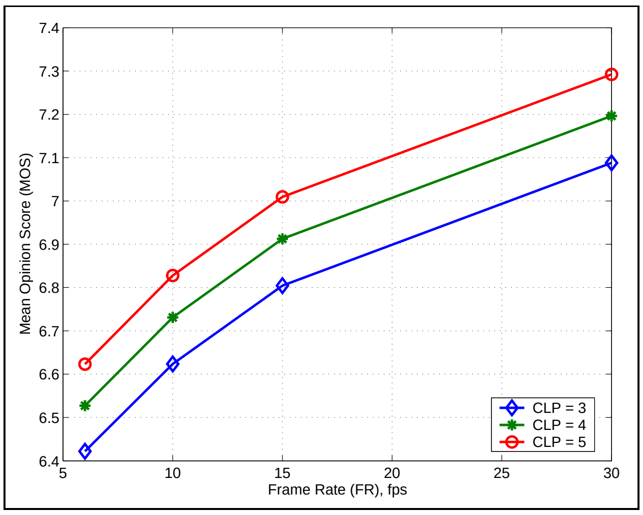
PSNR (dB)	MOS
≥ 37	5 (Excellent)
31 – 37	4 (Good)
25 – 31	3 (Fair)
20 – 25	2 (Poor)
≤ 20	1 (Bad)

$$\mathrm{MOS} = \begin{cases} -13.02 + 1.268x - 0.0287x^2 & \\ \qquad + 2.5\ 10^{-4}x^3 + 9.431\ 10^{-7}x^4 & \text{if } 16 \leq x \leq 37, \\ 1 & \text{if } x < 16, \\ 9 & \text{if } x > 37, \end{cases}$$

where x is the PSNR in dB and MOS is given in the ITU 9-point scale (shown in Fig. 7). Since the ITU 5-point scale is also very used, we can transform the value from the 9-point scale (say, MOS_9) to the 5-point one (say MOS_5) using $\mathrm{MOS}_5 = 0.5(\mathrm{MOS}_9 + 1)$.

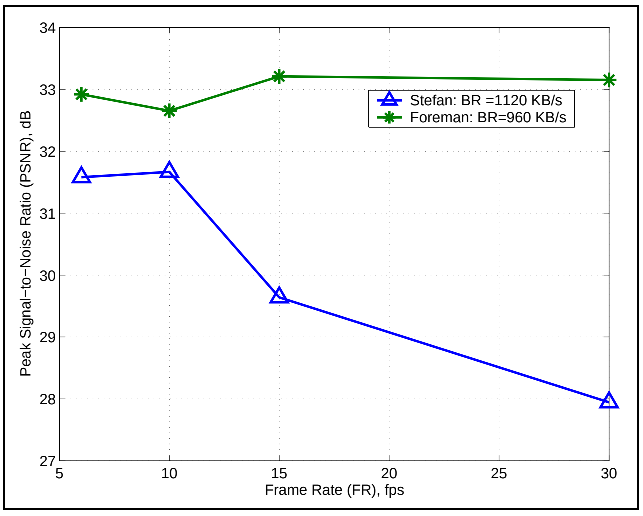


(a) PSNR obtained for different CLPs

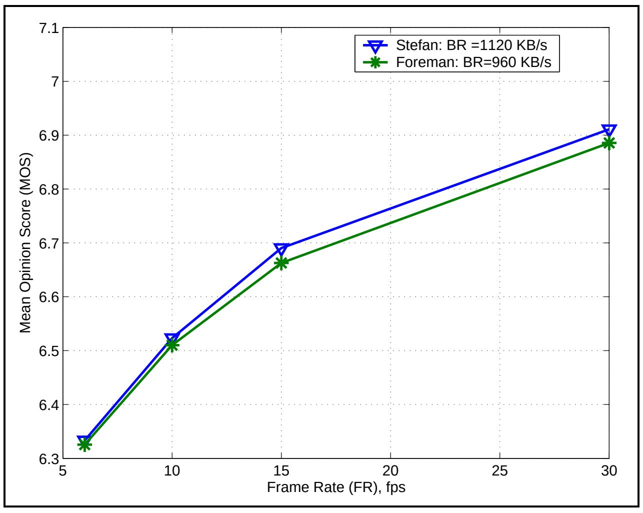


(b) MOS obtained by our application for different CLPs

Fig. 2. Keeping all the parameters fixed during each playback, but changing only FR for each playback. Foreman video sequence, LR=1%, BR=1200KB/s



(a) PSNR in dB



(b) MOS obtained by our application

Fig. 3. Keeping all the parameters fixed during each playback, but changing only FR for each playback, Stefan and Foreman sequences (LR=1%, CLP=1).

III. Comparison between PSNR and PSQA

To build our PSQA metric we used, for instance, a first set of 94 video samples, each corresponding to different source-network conditions [7]. For each one, we computed the PSNR in dB (together with the MOS using a subjective testing procedure [3]), and we mapped it into the ITU 9-points scale (see Sec. II-C). We show in Fig. 4 the PSNR value against the real MOS. In addition, we show in Fig. 5 a scatter plot showing the correlation between PSNR and MOS. We obtained for this experiment a correlation coefficient of 0.8012.

Using our methodology, we compare the MOS obtained by subjective test and those obtained by our RNN implementing PSQA. As we divided the set of samples into two parts, we show in Fig. 6(a) and in Fig. 6(b) scatter plots showing the correlation between our PSQA measure and by the subjective test for, respectively, the training and the testing (or validation) samples. The obtained correlation coefficient for each case is respectively 0.9801 and 0.9821.

As we can see from these Figures, the performance of our PSQA measure is very good with respect to the evaluation done through the PSNR. From Fig. 4, we see that for some samples the evaluation with PSNR is wrong. For example, in sample number 2, PSNR gives a value of 5.7 instead of 8.8. We would like to mention that in [14], researchers in VQEG argue that most of available objective quality metrics give approximately the same performance as PSNR. According to this claim, PSQA compares favorably with respect to the considered objective ones.

In the sequel, we analyze the impact of the different selected parameters on the accuracy of both measures.

A. *Changing LR*

We show in Fig. 8 a screen-dump of the application when changing manually the LR parameter for Stefan video sequence. The remaining parameters are kept constant having the values 1, 1400 KB/s, 30 frames/s and 0.02 for CLP,

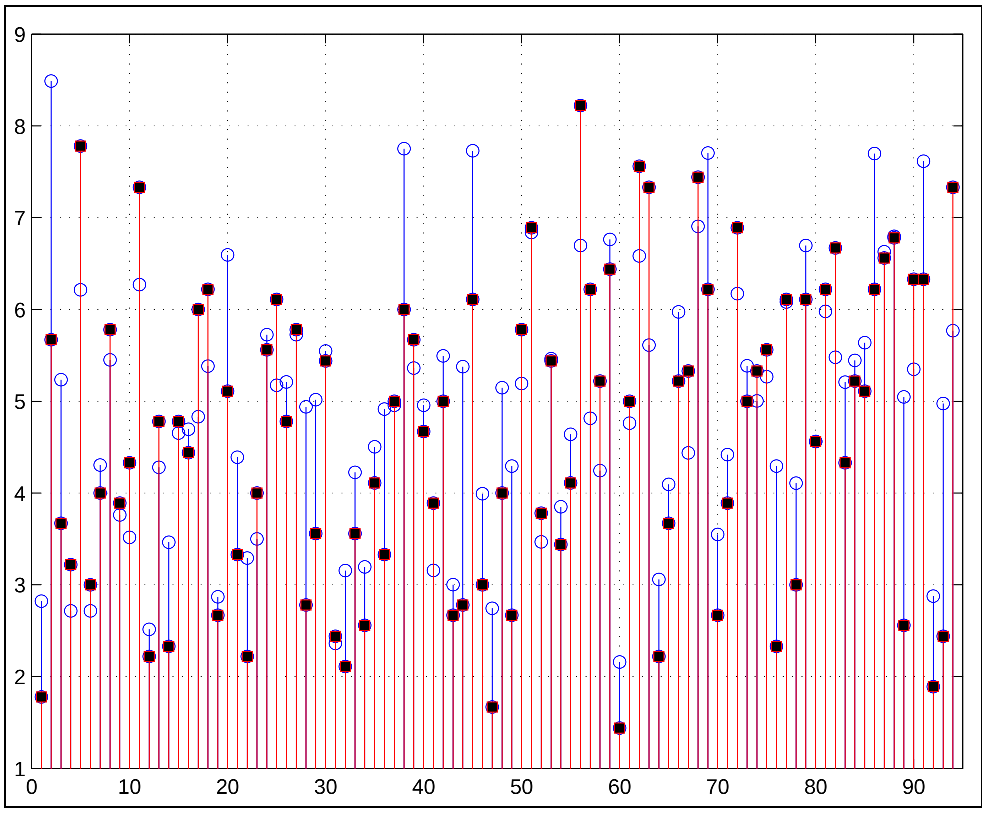


Fig. 4. The correlation between MOS and PSNR mapped to MOS.

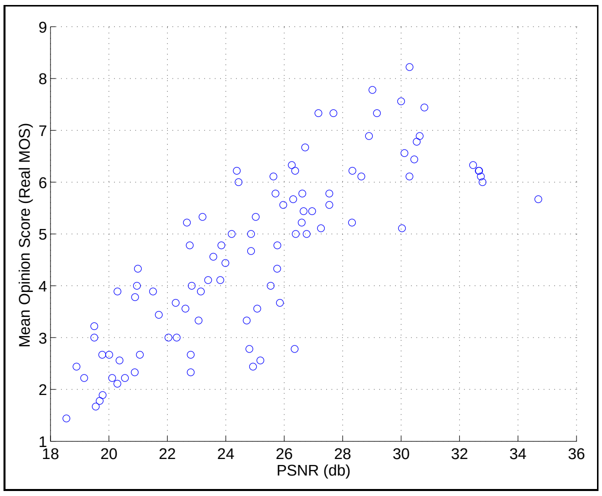
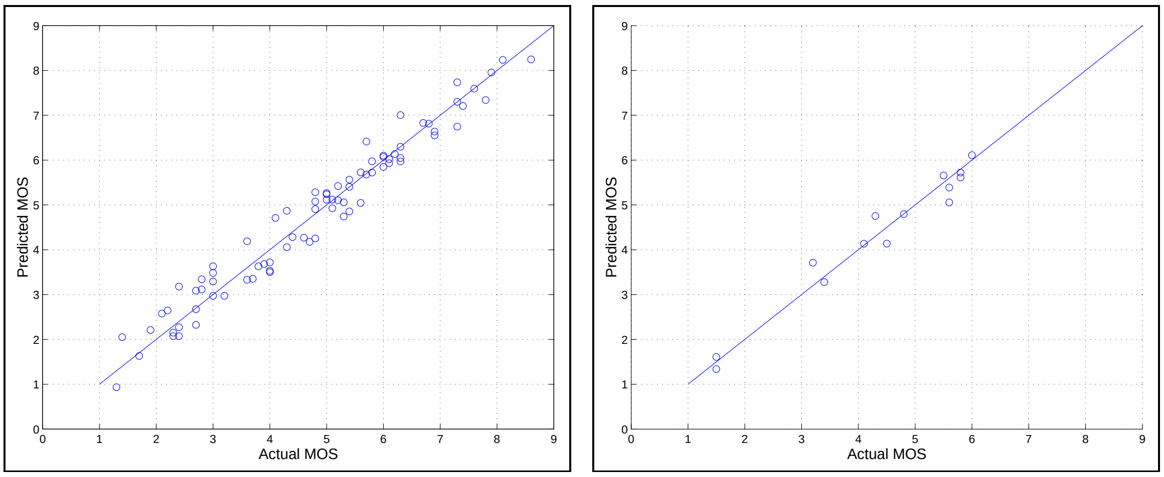


Fig. 5. Scatter plot showing the correlation between MOS and PSNR.

BR, FR and RA respectively. We show in Fig. 9 the variation of the PSQA (that, basically of MOS) and the value of PSNR mapped to the same scale (henceforth referred as MOS(PSNR)) as well as the corresponding value of LR with respect to time. (Note that the x-axis has discrete values for the time that corresponds to the sampling period: fixed to 300 ms). In most of the experiments, we fixed the sliding window size to 10 frames (about 333 ms).

As we can see PSNR cannot measure correctly the perceived video quality when changing LR. From Fig. 9 , at discrete time 10 PSNR gives a score of 2.8 while PSQA gives a score of 8.8 (a difference of 6 points). At that



(a) Training DB

(b) Testing DB

Fig. 6. Scatter plots showing the correlation between Actual and Predicted MOS scores using our application.

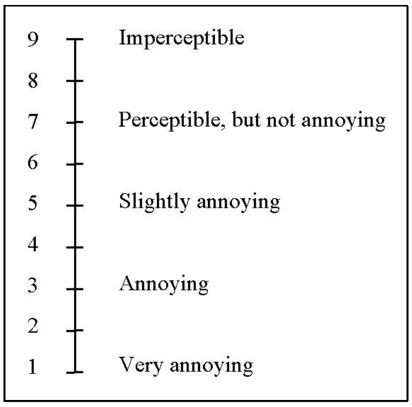
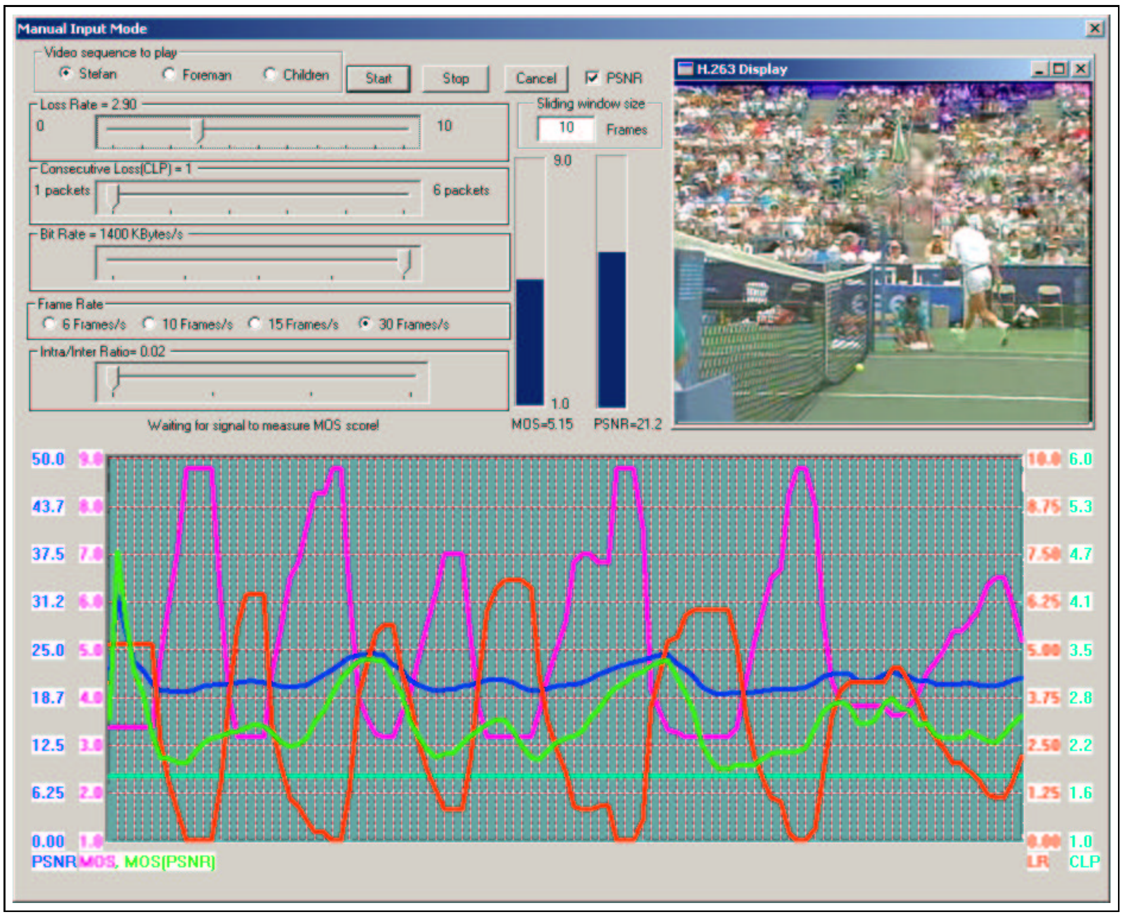


Fig. 7. The ITU-R nine-point quality scale

point LR=0% and all other parameters' values are set to the values that give the better quality. We can see also at the beginning of the figure (time 1 to 4), there is a peek given by PSNR. This is normal because H.263 codec has IPPPP . . . encoding mode, the only encoded as Intra mode picture is the first one. Knowing that Intra encoded picture is approximately the same as the original one and that PSNR uses pixel-to-pixel comparison, the PSNR value of the first frame is too high. The peek decreases rapidly from 7 to 3 as we fixed LR at 6% for "Time" < 10.

B. *Changing CLP*

Similarly to what we did concerning LR, we have varied CLP parameter from 1 to 5 then back to 1 and so on, as shown in Fig. 10. All other variables are kept constant. We can see that PSNR gives results that have no relation with the monotonically increasing or decreasing behavior of CLP. Actually, it is expected that when increasing only one parameter, while keeping all others fixed, the quality has to increase or to decrease monotonically. We can see that it is the case for the results obtained by PSQA, but not the case for PSNR. As an example, when CLP changes from 4


Manual Input Mode

Video sequence to play: (✓ Stefan) Foreman Children

Start Stop Cancel

PSNR [✓]

Loss Rate = 2.90 (0 – 10)

Sliding window size: 10 Frames

Consecutive Loss(CLP) = 1 (1 packets – 6 packets)

Bit Rate = 1400 KBytes/s

Frame Rate: 6 Frames/s, 10 Frames/s, 15 Frames/s, (✓ 30 Frames/s)

Intra/Inter Ratio= 0.02

Waiting for signal to measure MOS score!

MOS=5.15 PSNR=21.2

Fig. 8. A screen-dump showing manual mode for Stefan when varying manually the LR parameter.

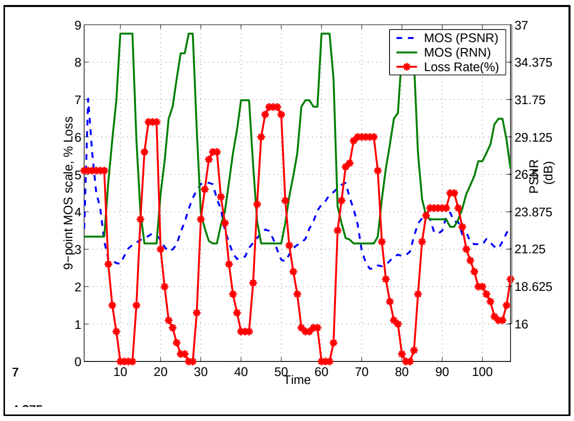


Fig. 9. A comparison between PSNR [MOS(PSNR): PSNR mapped to 9-point MOS scale] and PSQA [MOS(RNN): measuring subjective video quality by RNN] when changing manually LR and keeping all other parameters fixed (Data obtained from Fig. 8).

to 5 packets, PSNR changes from 6 to 2 (about 4 MOS points). That means that dropping 5 packets at a time instead of 4 packets degrades the quality significantly. However, dropping 4 packets instead of 3 packets tends to improve the quality. This is obviously not correct. (Again, the peek at the beginning for the results obtained by PSNR is due to the IPPPP . . . encoding mode, see before.)

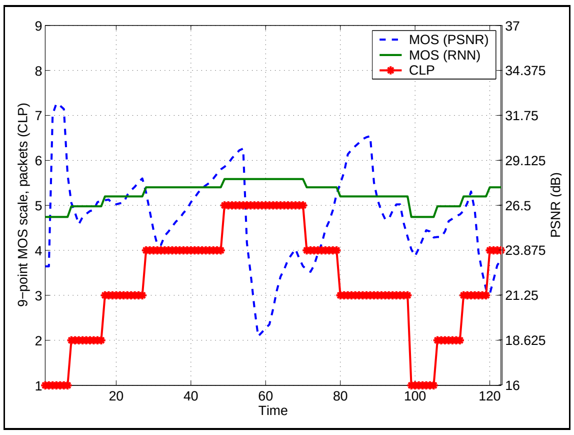
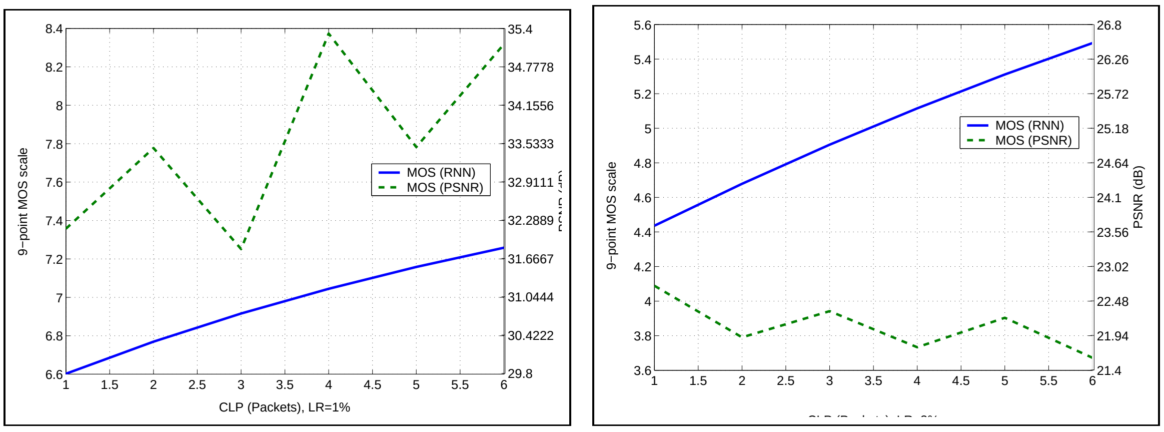


Fig. 10. Changing only CLP for Stefan video sequence. (LR=2.1%, BR=1120 KB/s, FR=30 frames/s, RA=0.02.)



(a) Foreman video sequence, LR=1%, BR=1200KB/s, FR=30 frames/s, RA=0.02

(b) Children video sequence, LR=3%, BR=1000KB/s, FR=30 frames/s, RA=0.01

Fig. 11. Keeping all the parameters fixed during each playback, but changing only CLP for each playback.

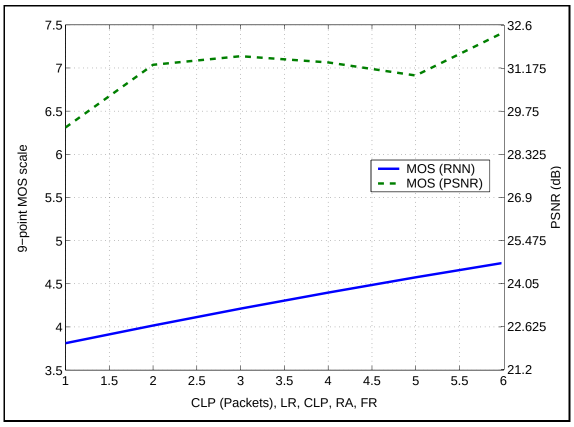
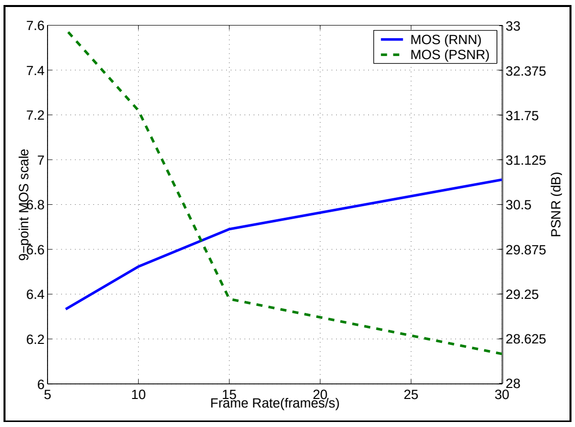


Fig. 12. Keeping all the parameters fixed during each playback, but changing only CLP for each playback (Children: LR=5%, BR=1000KB/s, FR=15 frames/s, RA=0.18).

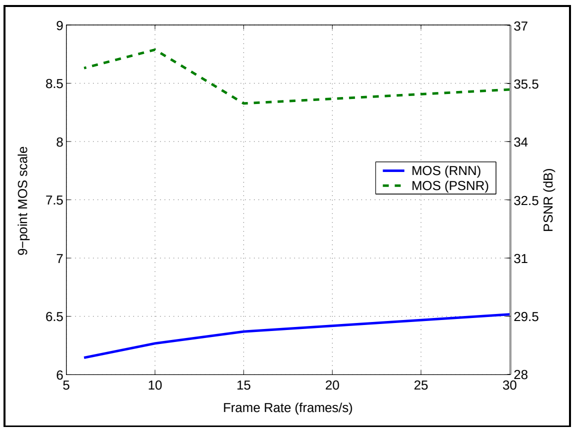
C. Changing FR

In our tool, we do not change the source parameters during the playback of the sequences. To analyze the impact of FR, we set the values of all the five parameters before each playback, repeating the experiment many times while changing the configurations. At the end of each playback, we take the average of MOS(PSNR) and MOS(RNN).

For the FR parameter, we varied it from 6 to 30 frames/s for Stefan and Foreman sequences; the values of the other parameters are as shown in Fig. 13(a) and Fig. 13(b). As we can see from Fig. 13(a), the PSNR measure tells us that the quality decreases when we increase the frame rate. The PSQA metric, in agreement with human observers (as recorded in our subjective tests), gives the right answer, that is, quality increases with the frame rate. A possible explanation to the behavior of PSNR is as follows. Reducing the frame rate for the same value of BR, decreases the quantization parameter of the encoder. Thus, pixels in the processed sequences are more similar to the original ones and the PSNR gives then higher scores when the frame rate is lower. If we look at Fig. 13(b) now, we see a different behavior of PSNR, with a difference of about 2 points with respect to PSQA. Note that LR have different values for the two figures (which explains the lower values of PSQA in Fig. 13(b)).



(a) Stefan: LR=0.5%, CLP=1 and BR=1120KB/s



(b) Foreman: LR=1%, CLP=4 and BR=960KB/s

Fig. 13. Keeping all the parameters fixed during each playback, but changing only FR for each playback.

D. Changing BR

We did the same for BR as for FR. We varied BR from the smallest to the full value for Stefan sequence. As we can see from Fig. 14, PSNR gives completely different measures than that obtained by PSQA. In addition, as in CLP case, the monotonicity behavior is not obtained. When increasing BR, PSNR's measure increases then decreases again.

E. Changing RA

Regarding the last parameter (RA), the same observations given previously can be shown in Fig. 15(a) and Fig. 15(b) for Stefan and Children sequences.

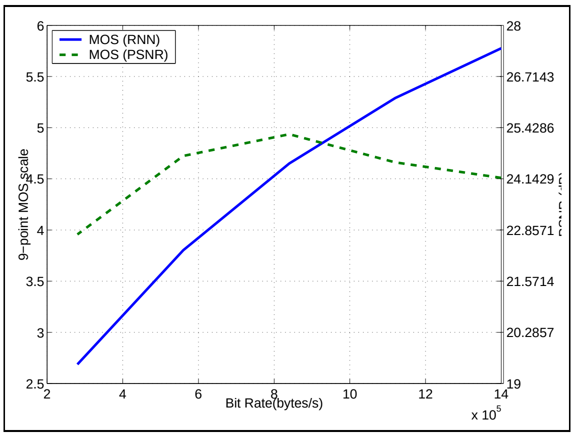
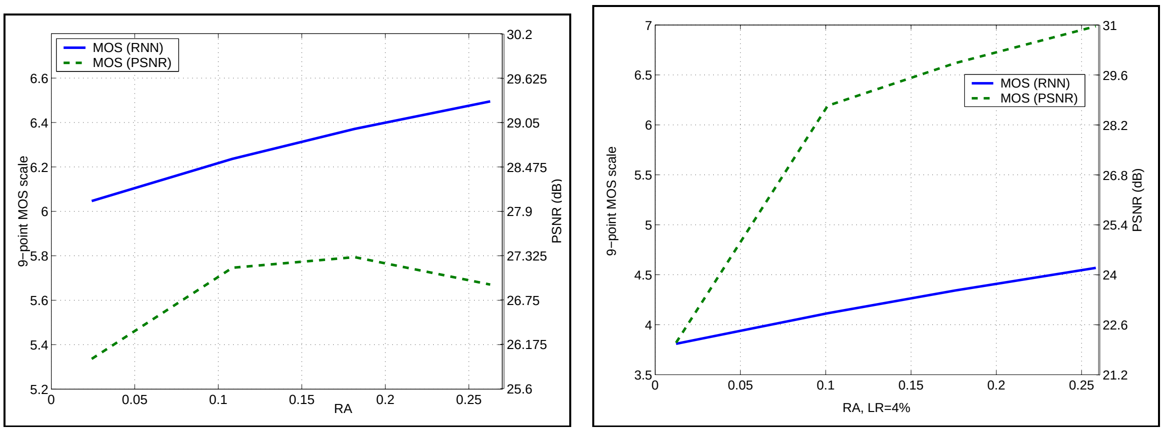


Fig. 14. Keeping all the parameters fixed during each playback, but changing only BR for each playback (Stefan: LR=2%, CLP=2, FR=30 frames/s, RA=0.01).



(a) Stefan: LR=1%, CLP=1, BR=1120KB/s, FR=30 frames/s

(b) Children: LR=4%, CLP=1, BR=1000KB/s, FR=30 frames/s

Fig. 15. Keeping all the parameters fixed during each playback, but changing only RA for each playback.

F. Changing Sliding Window Size

Here, we would like to show the effect of the Sliding Window Size (SWS), which is a direct parameter of PSNR evaluation. As explained before, we fixed the SWS to 10 frames for all the previous experiments. We conducted several experiments to show whether or not the SWS has a dominant effect on the overall value obtained by PSNR. We have varied SWS from 1 to 300 frames. In each experiment (a complete playback of the 10-sec sequence), a value is selected for SWS and the other parameters are kept constant. We show in Fig. 16 the variation of the results obtained by PSNR when changing only SWS. As we can see that the absolute difference is really very small (about 0.25 point), which is negligible on the ITU 9-point quality scale. For different combination of the parameters the results decrease instead of increase as shown (but the absolute difference is really negligible). This curve is obtained for Stefan sequence.

(We omitted several figures for the lake of place).

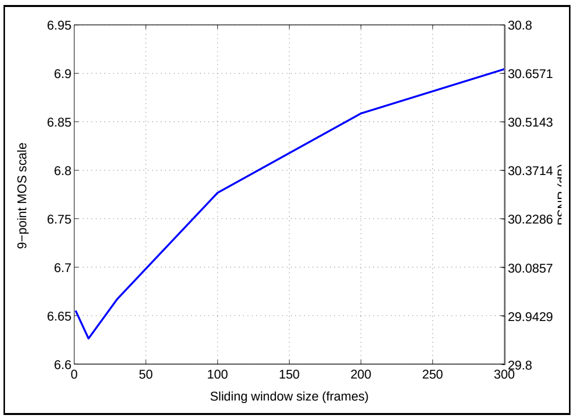


Fig. 16. Variation of MOS(PSNR) when changing sliding window size (Stefan).

IV. Performance of other Video Quality Measures

Among the available objective video quality measures in the literature, there are the ITS and EPFL metrics (the most known metrics). The first one is developed in the "Institute for Telecommunication Sciences", from which comes the acronym ITS. The other ones are developed in "École Polytechnique Fédérale de Lausanne (EPFL)". All of these measures require a prefect synchronization between original and processed sequences (i.e. the same frame rate). From that we can see that they cannot take into account the effect of frame rate degradation (the effect when the processed sequence has lower number of frames than the original one).

ITS Model [12, 13, 15] maps image imperfections onto measurable mathematical parameters. It is based only on the luminance values, the chrominance values are not considered. From that, we can see that the correlation with subjective tests cannot be very good, as the chrominance values are important to the HVS, even if its importance is not as high as that of luminance values. The correlation coefficient obtained when testing it for some video scenes (see Figure 17) without network impairments is about 94% as reported in [13]. However, when this metric is used to evaluate the quality when varying only the bit rate, the performance becomes very bad namely for low bit rate. This can be seen from Figure 19.

The other ones are the EPFL metrics, namely, the Moving Pictures Quality Metric (MPQM), the Color MPQM (CMPQM), and the Normalization Video Fidelity Metric (NVFM) [10, 11]. All these metrics are designed for high quality video broadcasting evaluation as stated in [10, p. 90]. As shown in Figure 18, MPQM cannot evaluate the quality in the presence of loss. Logically, it is expected that the quality becomes better when the bit rate increases (reducing the encoding artifacts). But once, a very small amount of loss is applied, the metric's output does the contrary, the quality decreases with the increase of the bit rate.

Another drawback for these metrics is that the computational complexity is too high. This may limit their use on real-time multimedia evaluation in the Internet. This is conformed by what is here quoted from [10, p. 88]: *"Due to computational complexity and memory management, the MPQM could only be applied to 32 frames of the sequences. It has been chosen to always use the first 32 frames of the video stream."*. Note that for a video sequence encoded at 30

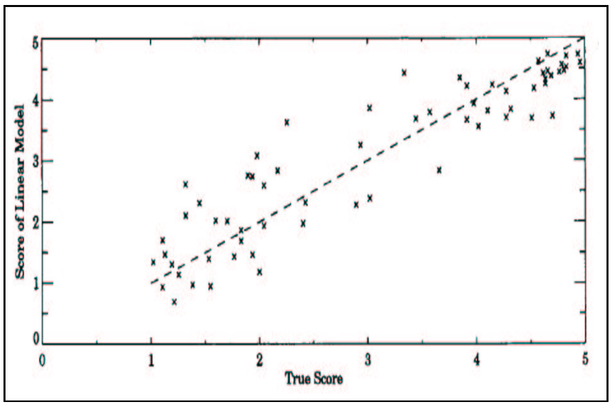


Fig. 17. The performance of the ITS metric to evaluate video quality. This Figure is taken from [13, p. 508].

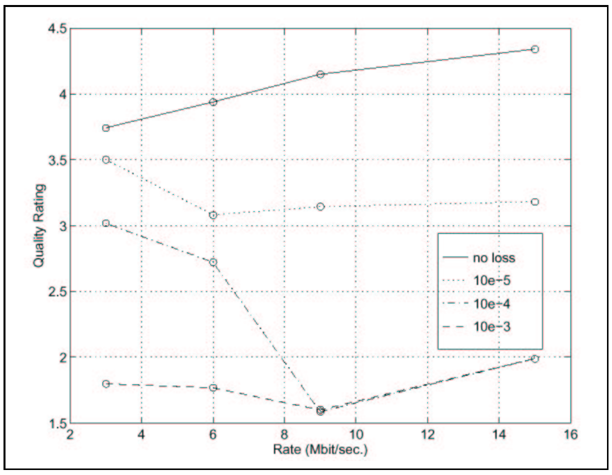


Fig. 18. Quality assessment by MPQM as a function of the bitrate and the loss rate. This Figure is taken from [10, p. 88].

frames/sec., there are 300 frames in only 10 sec. Note that MPQM metric is simpler and less accurate than CMPQM metric and that 32 frames correspond to 1 sec. of video.

Similar to ITS metric, MPQM and CMPQM cannot operate well for low bit rates, this is shown in Fig. 19. As we can see, MPQM and CMPQM give a score of 3 instead of 1 on the 5-point quality scale (as it should be at very low bit rate).

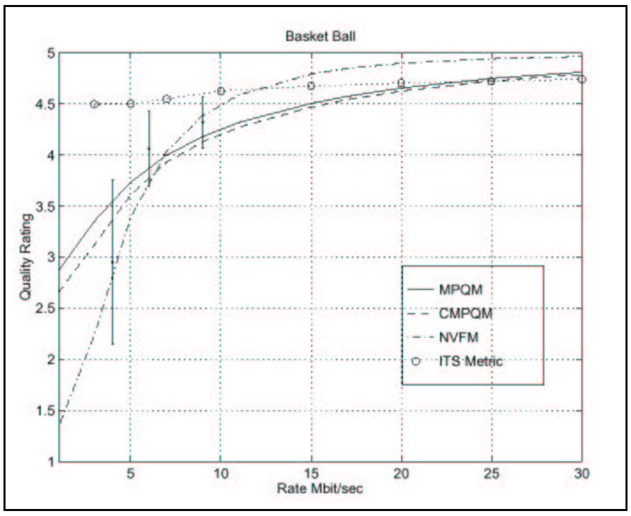


Fig. 19. Comparison of the subjective data (vertical solid lines) against MPQM, CMPQM, NVFM and ITS metrics for the video sequence “Basket Ball”. This Figure is taken from [10, p. 84].

V. Conclusions

In this paper, we compared our recently proposed methodology to measure video quality and some existing metrics, namely PSNR, ITS, MPQM, CMPQM and NVFM (with focus on the former). We showed that our method outperforms these measures in different aspects, especially with respect to the correlation between the obtained results and subjective measures. We presented also an application we built based on our method to measure in real time video quality taking into account the effect of encoding impairments and the distortion due to the transmission of the stream over packet networks. The considered parameters are loss rate, loss burst size, bit rate, frame rate and intra-to-inter block ratio. Given its good results, we are currently exploring the use of PSQA for control purposes.

References

[1] E. Gelenbe, "Random neural netowrks with negative and positive signals and product form solution," *Neural Computation*, vol. 1, no. 5, pp. 501–511, 1989.

[2] ——, "Learning in the recurrent random neural network," *Neural Computation*, vol. 5, no. 2, pp. 154–164, 1993.

[3] ITU-R Recommendation BT.500-10, "Methodology for the subjective assessment of the quality of television pictures," `http://www.itu.int/`.

[4] ITU-T Recommendation H.263, "Video coding for low bit rate communication," `http://www.itu.int/`.

[5] J. Klaue, B. Rathke, and A. Wolisz, "EvalVid - a framework for video transaction and quality evaluation," in *13th Intl. Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, Illinois, USA, Sept 2003.

[6] F. Le Leannec and C. Guillemot, "Packet loss resilient H.263+ compliant video coding," in *Proceedings of International Conference on Image Processing*, 2000.

[7] S. Mohamed and G. Rubino, "A study of real-time packet video quality using random neural networks," *IEEE Transactions On Circuits and Systems for Video Technology*, vol. 12, no. 12, December 2002.

[8] S. Mohamed, G. Rubino, F. Cervantes, and H. Afifi, "Real-time video quality assessment in packet networks: A neural network model," in *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, Las Vegas, Nevada, USA, June 2001.

[9] S. Mohamed, G. Rubino, and M. Varela, "Performance evaluation of real-time speech through a packet network: a random neural networks-based approach," *Performance Evaluation*, vol. 57, no. 2, pp. 141–161, 2003.

[10] C. van den Branden Lambrecht, "Perceptual models and architectures for video coding applications," Ph.D. dissertation, EPFL, Lausanne, Swiss, 1996.

[11] C. van den Branden Lambrecht, D. Costantini, G. Sicuranza, and M. Kunt, "Quality assessment of motion rendition in video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 5, pp. 766–782, 1999.

[12] S. Voran and S. Wolf, "The development and correlation of objective and subjective video quality measures," in *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, 1991, pp. 483–485.

[13] ——, "The development and evaluation of an objective video quality assessment system that emulates human viewing panels," in *International Broadcasting Convention - IBC*, 1992, pp. 504–509.

[14] VQEG, "The video quality experts group," http://www.its.bldrdoc.gov/vqeg.

[15] S. Wolf, M. Pinson, S. Voran, and A. Webster, "Objective quality assessment of digitally transmitted video," in *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, 1991, pp. 477–483.