Mean Time Between Visible Artifacts in Visual Communications

A Thesis
Presented to
The Academic Faculty

by

Nitin Suresh

In Partial Fulfillment
Of the Requirements of the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2007

Mean Time Between Visible Artifacts in Visual Communications

Approved by:

Professor Nikil Jayant
Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Yucel Altunbasak
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Russell Mersereau
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Allen Tannenbaum
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Constantine Dovrolis
College of Computing
Georgia Institute of Technology

Date Approved: 23 May 2007

To my parents and my sister

ACKNOWLEDGEMENTS

I thank my advisor, Dr. Nikil Jayant for teaching me how to be successful in the technological world, and for having complete faith in me right from the start to the finish of my PhD, and beyond. I would like to take the opportunity to thank my committee members, Dr. Yucel Altunbasak, Dr. Russell Mersereau, Dr. Allen Tannenbaum and Dr. Constantine Dovrolis for being an integral part of my doctoral degree.

I wish to thank Dr. Jayant again, for being an expert at choosing his co-workers. I am grateful to Barbara for setting an excellent example by being highly efficient and organized, and for being the gregarious person that helped me in my subjective experiments and scheduling issues. I am also thankful to Rex for helping in just about every technical aspect of my projects. I would like to thank Tina for organizing all the monetary issues related to my studies and company work. I had quite a bit of travel during my studies, and I am thankful to JoAnna for making sure that my trips were smooth. I would like to thank Pravin for being a keystone player in the development of the technology in the company that our team started up. I am grateful to Roberto and the other people from Venture lab for having faith in our capabilities. I would like to thank all my friends in the GCATT building - Arumugam, Babak, Jeannie, Nicolas, Rafi and Souvik, for being excellent peers. I would like to make a special mention of Parashuram and Yogesh, who have stuck with me since long before coming to Georgia Tech, enough to have made a significant impact on my personal and professional life.

Last, but definitely not the least, I thank my family - my parents and my sister for always having believed in me and providing a constant source of encouragement. They made all this effort worth it.

TABLE OF CONTENTS

Acknowledgements i
List of Tables v
List of Figures vi
Summary viii

1. Introduction 1

2. Visual Artifacts 4
2.1 Blocking artifacts 4
2.2 Ringing artifacts 4
2.3 Clipping 5
2.4 Noise 5
2.5 Contrast 6
2.6 Sharpness 7
2.7 Jerkiness 7
2.8 Mosquito Noise 7
2.9 Shimmering 7
2.10 Network errors 8
2.11 Post-processing errors 8

3. Mean Time Between Failures (*MTBF*) 9
3.1 *MTBF* as a standard quality measure 9
3.2 *MTBF* as a measure of video quality 10
3.3 Relationship between *MTBF* and consumer cost 12

4. Standardized Video Testing 15
4.1 Double Stimulus Continuous Quality Scale (DSCQS) 15
4.2 Test material 16
4.3 Objective measurements 16

5. Current Metrics and Approaches to Objective Video Testing 17
5.1 Peak Signal to Noise Ratio (*PSNR*) 17
5.2 Just Noticeable Difference metric (*JND*) 19
5.3 Spatial-Temporal 'Join' Metric (*STJM*) 19
5.4 Blockiness metric (*BLK*) 20
5.5 MSU toolbox 20
5.6 Vlachos blockiness metric 21
5.7 Harmonics Phase Blockiness metric 22
5.8 Picture Appraisal Rating (PAR) 22
5.9 Philips Electronics 23
5.10 NROQM 23
5.11 Structural Similarity Index Metric (*SSIM*) 24
5.12 AT&T Labs Research 25
5.13 NR_VQM Blocking Strength 26
5.14 NR_VQM 26
5.15 Distortion Assessment Model (DF_{IMAGE}) 27
5.16 Continuous Video Quality Evaluation (CVQE) 27
5.17 Tektronix 27
5.18 Digital Video Quality Analyzer (DVQ) 28
5.19 IneoQuest 28
5.20 V-factor 29
5.21 Absolute Temporal Information (ATI) Metric 30
5.22 Relative Peak Signal to Noise Ratio 30
5.23 Streaming Video Quality Parameters 31

6. Subjective Experiments to Measure *MTBF* and *MOS* 32

7. Relationship between *MOS* and *MTBF* 39
7.1 Correlation between *MOS* and *MTBF* 41
7.2 Variation in scores 41

8. Comparison of Current Objective Metrics with *MTBF* 42

8.1 Explanation of objective metrics that are compared with *MTBF* 42
8.2 Error pooling 43
8.3 Relating objective scores to *MTBF* 44
8.4 Estimation of *MTBF* from objective metrics 46

9. The Automatic Video Quality Metric (AVQ) 48
9.1 Components of the AVQ 49
9.1.1 Quantization Step Size 49
9.1.2 Number of DCT Coefficients 52
9.1.3 Bit Rate 53
9.1.4 New Blockiness Metric 53
9.1.5 New Network Error Streak Detector 59
9.1.6 New Blurriness Metric 63
9.1.7 Delta-Autocorrelation-Method 66
9.2 AVQ meter: Combination of different components 71
9.3 Performance of the AVQ metric 74

10. Future Work 77
10.1 Extensions to H.264 and other compression standards 77
10.2 Implementation of newer modules in the real time AVQ metric 78

11. Conclusion 79

Appendix A 80

References 86

Vita 87

LIST OF TABLES

1. The different test conditions in the test database 34

LIST OF FIGURES

1. *MTBF* calculation from failure characteristics 10
2. Verizon scatter plot: artifacts tolerated by consumers for a given discount 13
3. Consumer survey: most common artifacts 13
4. Consumer survey: relative tolerance of different types of artifacts 14
5. Consumer survey: cost versus *MTBF* report 14
6. DSCQS method 15
7. Effectiveness of *JND* over *PSNR* 18
8. Spatial-Temporal blocks of the *STJM* metric 19
9. Vlachos blockiness metric 21
10. SSIM difference maps for a pair of images 25
11. Absolute Temporal Information metric 30
12. VQEG clips concatenated to form the ~140 seconds test sequence 32
13. A two-state Gilbert model 34
14. Subjective testing scheme 35
15. Network errors introduced in the test database 36
16. Artifact triggers obtained from viewers (test database1) 37
17. Artifact triggers obtained from viewers (test database2) 37
18. Failure rate characteristics of the test sequence 38
19. Failure rate for different smoothing windows 38
20. Scatter plot between *MTBF* and *MOS* 40
21. Average relationship between *MTBF* and *MOS* 40
22. Objective metrics vs. Bit rate 42
23. Objective metrics vs. Time 43
24. Scatter plots of Objective metrics with *MTBF* 44
25. Scatter plots of Objective metrics with log(*MTBF*) 45
26. Estimation of *MTBF* from objective metrics 47
27. GT_AVQ flowchart (chart 1) 49
28. GT_AVQ flowchart (chart-main) 50

29. Estimation of locations of intra frames 51
30. Estimation of locations of network errors 51
31. Pixel-based compression artifact measure 51
32. Stream-based compression artifact measure 52
33. Estimation of network artifact measure 53
34. Pixel-based blockiness measure 57
35. Screenshots of MPEG2 video at different qualities 58
36. Spatial masking used in metric calculation 59
37. Sample temporal masking approach 60
38. Pixel based network error artifact metric 62
39. Sample screenshot of the AVQ meter 63
40. Concept behind a pixel-based blurriness measure 65
41. Pixel-based blurriness measure 66
42. Sample input video to a H.264 encoder 68
43. Schematic diagram of the Delta-Autocorrelation method 69
44. Delta-Autocorrelation method as a *full-reference* metric 69
45. Delta-Autocorrelation method as a *no-reference* metric 70
46. Evaluation of Delta-Autocorrelation as a *full-*, *reduced-* and *no-*reference 70
47. Correlation of *MTBF* with AVQ composed of different components 73
48. Accurate detection of network error artifacts by the AVQ meter 74
49. *MTBF* spread plots for different objective metrics 74
50. *MTBF* time plots for the AVQ meter 75
51. A screenshot of the AVQ meter implemented in real-time 76
52. AVQ architectural component overview 80
53. AVQ in media player 84

SUMMARY

As digital communication of television content becomes more pervasive, and as networks supporting such communication become increasingly diverse, the long-standing problem of assessing video quality by objective measurements becomes particularly important. Content owners as well as content distributors stand to benefit from rapid objective measurements that correlate well with subjective assessments, and further, do not depend on the availability of the original *reference* video.

This thesis investigates different techniques of subjective and objective video evaluation. Our research recommends a functional quality metric called Mean Time Between Failures (*MTBF*) [1] where failure refers to video artifacts deemed to be perceptually noticeable, and investigates objective measurements that correlate well with subjective evaluations of *MTBF*. In this work, the subjective tests for evaluating *MTBF* involve different video clips from the Video Quality Experts Group (VQEG [2]) encoded in MPEG-2 format at bit rates in the range of 1.5 - 5 Mbps, and subject to packet losses in the range of 0.1 – 2.0 %. Each of the test clips is 140 seconds in length, and a diverse viewer pool of 30 subjects was used.

Work has been done for determining the usefulness of some existing objective metric by noting their correlation with *MTBF*. The metrics studied include *full-reference*, *reduced-reference* and *no-reference* objective metrics: PSNR, Just Noticeable Difference metric (JND) [3], Spatial Temporal Join Metric (STJM) [4] and Blockiness metric (BLK) [5]. The research also includes experimentation with network-induced artifacts, and a study on statistical methods for correlating candidate objective measurements with the subjective metric [6]. The statistical significance and spread properties for the correlations are studied, and a comparison of subjective *MTBF* with the existing subjective measure of *MOS* is performed. These results suggest that *MTBF* has a direct and predictable relationship with *MOS*, and that they have similar variations across different viewers, when computed over any clip

The research is particularly concerned with the development of new *no-reference* objective metrics that are easy to compute in real time, as well as correlate better than current metrics with the intuitively appealing *MTBF* measure. The approach to obtaining greater subjective relevance has included the study of better spatial-temporal models for noise-masking and test data pooling in video perception.

A new objective metric, 'Automatic Video Quality' metric (AVQ) [6] is described and shown to be implemented in real time with a high degree of correlation with actual subjective *MTBF* scores, with the correlation values approaching the correlations of metrics that use full or partial reference. This is metric does not need any reference to the original video, and when used to display MPEG2 streams, calculates and indicates the video quality in terms of *MTBF*. Certain diagnostics like the amount of compression and network artifacts are also shown.

CHAPTER I

Introduction

Video quality evaluation is an important problem in audiovisual communications. The need for perceptually meaningful objective metrics is broadly recognized, and such measures have the dual role of (a) understanding signal quality in completed algorithm designs and (b) providing an in-the-loop metric for real-time algorithm steering. On both counts, significant progress has been made in the areas of speech and audio coding, including the sub-domains of wireless telephony and satellite audio, but the video metrics problem continues to be generally evasive, with no definitive objective solution in sight. So much so, the traditional metric of PSNR (Peak Signal to RMS Noise Ratio) remains as the single most used objective measure, with opportunistic supplementing by approaches like the JND [3] and VQM [7] metrics.

For video, subjective testing is the ideal approach, since it involves real viewers evaluating the end output. Objective testing for video is more practical, since subjective testing takes up a lot of time and effort. Our research involves the estimation of subjective scores from users in an intuitive fashion for a set of test clips and evaluating the correlation of objective scores of different *full-reference*, *reduced-reference* and *no-reference* metrics with the subjective scores. In current subjective testing methodology, the discrete-point scales of *MOS* (Mean Opinion Score) and MIS (Mean Impairment Score) are well understood and provide useful quality measurements under conditions in which there is adequate training of subjects, and if the mean scores are appropriately qualified by a standard deviation score reflecting inter-viewer differences. There are quite a few established methods for subjective video quality evaluation [8]. In the DSCQS (double stimulus continuous quality scale) method, viewers make two absolute ratings on a continuous scale at discrete times; in the DSCS (double stimulus comparison scale) method, viewers make one difference rating on a discrete scale at discrete times; and in the SSCQE (single stimulus continuous quality evaluation) method, viewers make absolute ratings on a continuous scale continuously over time. The Double Stimulus methods are claimed to be less sensitive to context, whereas the SSCQE method is claimed to yield more representative quality estimates for quality monitoring.

Some guidelines for subjective testing are described in [9], and they involve the viewers watching different video clips and giving each clip a score, or giving a continuous score using a user feedback device

like a slider or throttle. Some of the desired characteristics of a testing scheme involve ease, intuitiveness, effectiveness, and giving the user real-time feedback about the current score. One of the problems with the existing methods may be a score drift over the course of the test. For e.g., viewers might concentrate on moving the slider in the proper direction to track changes in quality and hence lose track of the absolute slider position on the rating scale, adversely impacting the method's reliability. In both the Single and Double Stimulus experiments, asking the viewer to give a score based on a continuous / discrete scale might have some problems with the viewer being consistent in deciding the scores over the range of testing time. In this work, the subjective testing methodology featured in [1] is used to get values of *MTBF* for some test clips from the VQEG (Video Quality Experts Group) encoded in MPEG-2 format at bit rates in the range of 1.5 - 5 Mbps. Objective measurements that belong to the so-called *full-reference*, *reduced-reference* and *no-reference* categories are investigated, and their correlations with subjective *MTBF* numbers are evaluated [1]. The *full-*, *reduced-* and the *no- reference* categories are based on whether information about the original video is fully available, partially available, or not available at all, respectively. The current test data relates to the type of video content used in entertainment television. The effect of artifacts introduced by network-induced errors is studied as well.

The statistical significance and spread properties for the correlations are studied in detail, and a comparison of subjective *MTBF* with the existing subjective measure of *MOS* is performed. This work also involves the development of a new no-reference objective metric, termed the 'Automatic Video Quality' metric (AVQ) [6]. The metric consists of different modules that function on the bit-stream and / or the actual pixel values. The various modules are incorporated into the metric algorithm depending on the availability of the bit-stream and output pixels. Certain diagnostics like the amount of compression and network artifacts are also shown. This metric has been implemented in real time as an AVQ meter with a high degree of correlation with actual subjective scores. The AVQ meter needs no reference to the original video, and when used to display MPEG2 streams, calculates and indicates the video quality in terms of *MTBF*.

This document is organized as follows: Section 2 describes some of the artifacts commonly experienced while watching video and section 3 explains the role of 'Mean Time Between Failures' as a quality metric and extends this concept to video quality. The relationship between *MTBF* and cost issues is

addressed as well, in section 3. Section 4 gives a brief introduction about the activities of the Video Quality Experts Group, and section 5 explains the experiments carried out to determine the subjective scores of degraded video. Section 6 evaluates the relationship between *MTBF* and the current *MOS* methodology. Sections 7 and 8 list some of the existing objective metrics and how they can be used to predict the *MTBF* of a given video based on sample subjective scores. Our Automatic Video Quality (AVQ) metric is described in section 9, and section 10 list some future work. Section 11 summarizes the work.

CHAPTER II

Visual Artifacts

This section describes the types of artifacts encountered in general in processed images and video. The presence of different kinds of artifacts in different proportions gives us an overall perception of the quality of video.

2.1 Blocking artifacts

Blockiness is the most common artifact found in compressed images and video. Blocking artifacts are the result of coarse quantization of DCT coefficients in a coded block. The amount and visibility of blocking artifacts increases with increased compression, i.e. lower bit-rate of the processed video. Perceived quality is strongly affected by blocking artifacts. Blocking artifacts can be measured as the number of 8-pixel edges found on the 8x8 grid overlaying the image. To differentiate block edges from natural edges it is assumed that natural edges are strong (i.e. very steep) transitions while block edges are weak and regularly spaced. One method to measure blocks is by the size of the discontinuity for pixels at block boundaries. In actual images, the transitions are gradual, and the edge is at the center of such transition (usually detected as the point where the second derivative is zero). Another simple method is to calculate the extrapolated discontinuity between 8x8 blocks [10].

2.2 Ringing artifacts

Another well-known kind of MPEG artifact is called *ringing*. Ringing is a shimmering effect around high contrast edges; depending on the orientation it manifests itself as edge doubling. Ringing is not necessarily correlated with blocking as the amount of ringing depends on the amount and strength of edges in the image, while blocking depends on the presence of uniform or smoothly changing regions. An algorithm to detect and measure ringing includes the following steps [10]:

1. Detect strong edges using a high threshold for the edge transitions.
2. Detect low activity regions, or adjacent to strong edges where most local variances are very low.
3. Detect ringing pixels in the low activity regions as pixels where the local variance is large compared to the other variances. E.g. if the local variance for most pixels in a low activity region less or equal 3,

then the local variance for a ringing pixel must be at least four times that value. The sum of all ringing pixels on the image is the ringing value.

An alternative method to measure ringing artifacts is called the visible ringing measure (*VRM*) [11]. The VRM is based on the average local variance calculated on a small-size window in the vicinity of major edges. Those regions, which exclude the edges, are detected through morphological operations. In general, blocking artifacts are found in lower bit rates, and ringing artifacts are observed at higher bit rates.

2.3 Clipping

Clipping refers to the truncation in the number of bits of the image values (luminance and chrominance components) imposed by the arithmetic precision of the process being used. It results in abrupt cutting of peak values at the top and bottom of the dynamic range, which leads to aliasing artifacts caused by the high frequencies created at those discontinuities. The sharpness enhancement technique known as peaking can cause clipping [12]. Peaking works by adding positive and negative overshoots to the edges. However, if the extreme values are beyond the limits of the dynamic range, saturation occurs and the pixels are clipped (i.e. pixels take maximum/minimum values of 255 or 0 for 8 bit precision). The simplest clipping measurement is a function of the number of clipped pixels found in the image. The clipping metric is defined as 0.0 when no pixels are clipped and 1.0 when 1% or more of the pixels are clipped. A fixed margin can be applied to the image on the left, right, top and bottom to avoid counting any blanking or black bars, as well as to speed up the measurement. A clipping measurement algorithm includes:

1. Test every pixel on the image except the margin on top, bottom, left and right.
2. If the pixel is 0 or Max (e.g. 255 if the precision is 8 bits) increase the count.
3. Calculate the percentage of clipped pixels for the tested image.
    a. If it is 0%, clipping is 0. If it is 1% or more, clipping is 1.0
    b. Other values are simply the percentage.

2.4 Noise

Noise is a random variation in the spatial or temporal dimension, which appears in video images as a result of random processes linked to transmission and generation techniques. It is most noticeable in smooth regions or regions with smooth transitions. It gives the subjective impression that the image is not clean, or

that something unintended is superimposed on the image. In some cases, small amounts of high-frequency noise add to the "naturalness" of textures (in contrast with a plastic or synthetic appearance) and have been found to increase perceived quality. Most noise, however, obscures details and reduces quality of the visual information. To measure noise, most algorithms assume that any image contains small areas of constant brightness (i.e. no details are present). Hence, the variation in these areas is nothing but noise. A typical noise measurement algorithm would consist of the following steps:

1. Divide the image into small blocks.
2. Measure the intensity variations for every block.
3. Assuming that the intensity of the noise is much smaller in magnitude than the signal, the block with least variation (or the average of the blocks with the smallest variation) should correspond to a constant brightness region.
4. Use a set of high-pass filters or a band-pass filter to filter out the DC component. The sum of the outputs of the filters, clipped using perceptual thresholds proposed in [13], is used to compute the variance or noise.

2.5 Contrast

In simple terms, contrast is related to the difference between the luminance of pixels of interest and the background, and largely depends on the dynamic range of the luminance signal. Contrast sensitivity is the ability to distinguish objects from the background. The perception of contrast depends on several factors including a mental reference image of the object in question, overall luminance (although not in all cases), background, and color. Some contrast perception issues are described in [14, [15]].

A very basic algorithm to measure contrast includes the following steps:

1. Compute the luminance histogram of the image, excluding a fixed margin on the left, right, top and bottom.
2. Separate the upper and lower parts of the histogram that contain each a certain percentage of the total energy.
3. Calculate the difference between the luminance of the upper and lower parts of the histogram and normalize by the average luminance.

This metric is relevant because for a given image contrast increases are associated with improved quality. Thus it is useful as a first approach. Many interactions with other features can be expected. However, this rudimentary definition of a contrast metric is content dependent and will probably require further research to reduce or eliminate that dependency.

2.6 Sharpness

Image sharpness is the informal, subjective evaluation of the clarity of detail and contours of an image. Objectively, it can be measured by the definition of edges in the spatial domain, or by the characteristics of the high frequencies in the transformed domain. Perceived sharpness is highly dependent on content, and also of spatial resolution, contrast, and noise as reported in [16]. For a no-reference sharpness metric a reduced content dependency is required, i.e. minimum baseline, and with the same dynamic range for any image. A sharpness metric based on the local edge kurtosis is described in [17]. This metric takes into account spatial and local frequency information and uses the weighted kurtosis of 8x8 blocks that contain the edge pixels. The algorithm consists of the following steps:

1. Create the edge image using an edge detection method.
2. Assign edge pixels to 8x8 blocks (can use the MPEG grid blocks).
3. Compute 2D kurtosis for each block and weight by number of edge pixels.
4. Sharpness is the 2D kurtosis averaged over all blocks.

This metric by itself shows high correlation with perceived sharpness, i.e. quality changes affected only by sharpness. Therefore it is used as a perceptual sharpness metric.

2.7 Jerkiness

When there is a frame rate conversion procedure, the resulting video might not be smooth. This unnatural effect is termed as jerkiness.

2.8 Mosquito Noise

Mosquito Noise is the noise observed around stationary edges. It can be calculated as the sum or absolute frame differences among pixels in a small neighborhood around an edge pixel.

2.9 Shimmering

Video sequences are usually coded in a group of pictures (GOP). Periodic refresh or key frames are called I-frames, and intermediate frames are the P- and B- frames. Due to the nature of coding, an IBP drift occurs

at times: There is a sudden difference between an I- frame and the previous P- or B- frame, caused by motion estimation errors which increase in the B- and P- frames between two I-frames. In general, the quality of an I-frame is different from the P- and B- frames, and this results in a shimmering effect.

2.10 Network errors

The errors described so far are coding errors, or those introduced by the coding scheme as such. When video streams are transmitted over a network, a variety of network errors can cause different types of distortions in the received video. The streams are transmitted in the form of packets, and network errors usually follow a bursty pattern. This can result in many consecutive packets being lost. Due to the nature of the coding algorithm, an error in one frame also gets propagated till the next refresh frame is sent. In [18], a method for detecting network errors is described. It involves comparing a specific signature calculated on the original and compared with the same signature as calculated on the processed video. The signature is calculated by measuring the distortion between consecutive frames.

2.11 Post-processing errors

When the video is received at the output, the end system does some post-processing on the signal. This may include smoothing, noise removal, intentional noise addition, gamma correction, brightness control, etc. Some times, these algorithms themselves cause visual artifacts. For example, too much smoothing can cause plastic / cartoon-like appearances of natural images.

A good video quality metric should technically evaluate the different kinds of visual artifacts present in the video, and pool these results to find the overall quality as perceived by the human visual system.

CHAPTER III

Mean Time Between Failures (*MTBF*)

This section defines the subjective metric called *MTBF* and describes an experimental methodology for evaluating it. In contrast with subjective testing techniques that involve viewers giving a score on a specific scale, say 1-5, or 0-100, a more intuitive approach is followed in our research. *MTBF* is a common term used in the measurement of quality of service. An example of evaluating *MTBF* is explained below:

3.1 *MTBF* as a standard quality measure

For example, in the case of evaluating the *MTBF* of a factory component, the failure statistics for a sample of the product can be collected for this purpose. Say, ten samples are run for 1000 hours each, and 4 of them fail after certain hours in that duration (fig. 1): sample 2 fails after 700 hours, sample 5 fails after 200 hours, sample 6 fails after 800 hours, and sample 9 fails after 300 hours. This experiment corresponds to a total of 4 failures for 8000 hours of operation. That means the failure rate is 1 in 2000 hours, which results in a calculation of *MTBF* = 2000 hours. *MTBF* is calculated as:

$$MTBF = 1 / (\textit{average failure rate}) \tag{1}$$

The reason for showing this factory components example is to illustrate that with just 1000 hours of experimentation time, it is possible to measure *MTBF* values of much larger than 1000 hours. On a similar note, it is possible to measure the quality of video that has a *MTBF* of few minutes by just showing an 8 second clip to a few viewers.

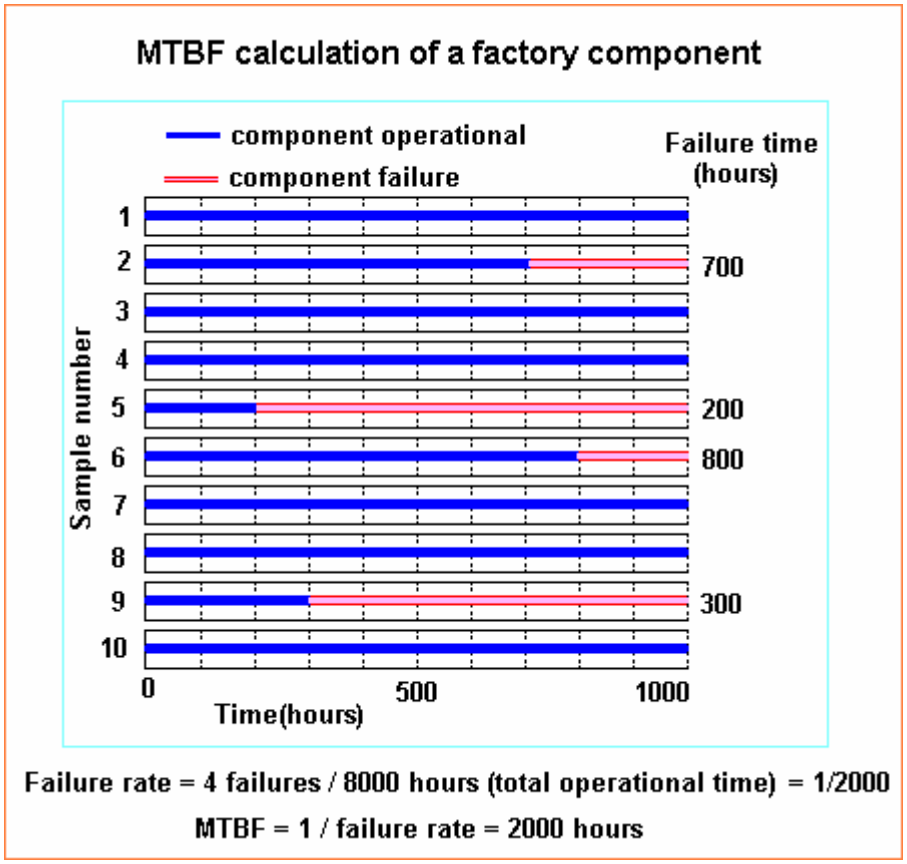


Fig. 1: *MTBF* calculation from failure characteristics

3.2 *MTBF* as a measure of video quality

The concept of *MTBF* is applied to subjective video quality evaluation as a global metric in [1]. *Failure rate* is a related instantaneous metric based on failure statistics, where failure corresponds to the occurrences of visual artifacts. As seen later in section 8, failure rate is useful in computing *MTBF* specifics, such as *MTBF* as function of viewer or stimulus. Whenever a perceptual artifact is observed, the viewer is asked to indicate this, say, using a buzzer. Common artifacts such as noise, blurriness, blockiness, etc. are explained and shown to the viewer prior to the testing. The viewer is allowed to keep the buzzer pressed if an entire stretch of video sequence looks bad.

The idea behind this methodology is that the viewer intuitively tends to give feedback intermittently, with a frequency correlating with how bad the video looks. Though the locations of the user responses are arbitrary for a particular viewer during a particular experiment, the results for a modest number of experiments with a sufficient number of viewers can be averaged out to generate a continuous

score versus time, which is nothing but the probability that the average viewer would observe a visual artifact while watching the video. The user responses can be pooled over a period of time to determine the *MTBF* of the video. There are many advantages of this metric: it is highly intuitive, time invariant and the user need not have real-time feedback about the current score. This is intuitive, because the viewer just has to decide whether the video has any artifact or not. The metric is functional, being directly related to how consumers evaluate otherwise high-quality video. *MTBF* is not concerned with the physical categorization of an artifact, only that it is deemed visible. In this sense, it is non-diagnostic, but simple and universal.

There are certain portions of the video that a viewer finds acceptable during some trials, and finds it to be error-prone during other trials. For e.g., if there is a small jerkiness in the video, it might probably get noticed by a viewer 20% of the time. If the artifact is extremely noticeable, the viewer would probably notice it all the time. There are also some portions which one viewer finds to be bad, while others feel that it is okay. This does not just mean that some viewers are more forgiving than the rest. There are cases where portion 'a' is found to be acceptable by viewer 'X' and found to be unacceptable by viewer 'Y'; and portion 'b' is found to be unacceptable by viewer 'X' and found to be acceptable by viewer 'Y'. Also, when a viewer presses a buzzer, it is natural to assume that the viewer could not have been extremely accurate at pinpointing the occurrence of the visual error. It makes more meaning to estimate the probability of a video portion looking bad, rather than estimating if the portion looks absolutely good / bad for the above-mentioned reasons.

Keeping these points in mind, a simple rule for getting the failure rate characteristic (vs. time, for a certain bit-rate) is devised: The failure vs. time graph for all viewers is averaged, and smoothed out with respect to time. With this failure rate characteristic, we can easily estimate the *MTBF* of the system. For example, if the probability of failure is 1/60 (per frame) on an average, then an observable failure occurs roughly once every two seconds, assuming a frame rate of 30 fps. *MTBF* functions as a reliable and scalable subjective quality metric. It should be noted though, that observing the artifacts also depends on various factors such as the display type/size and viewing distance. *MTBF* represents subjective scores of a typical viewer for a typical output display environment.

3.3 Relationship between *MTBF* and consumer cost

The concept of *MTBF* as a measure of subjective quality was introduced in [19], and it was well received in the video quality evaluation committee. Recently, a study has been performed related to the relationship between *MTBF* and consumer cost [20] by Verizon. This study relates to the price consumers are willing to pay for a certain level of quality in video. They also list statistics concerning the types of artifacts that are found to be the most annoying.

In their study, the customers were not actually shown any video, but were asked to fill out some surveys concerning the cost versus quality tradeoffs. The customers were initially asked about the kinds of artifacts that were generally present in the video service. Understandably, block artifacts and screen freezes were among the most noticed (fig. 2). The customers were asked to choose between perfect quality video at the current cost and a video with some artifacts at a lower cost. The severities of the artifacts were explained intuitively to the customers in terms of Mean Time Between Failures. The data collected for the fraction of customers preferring the lower quality video at a lower cost is displayed in fig. 3. Fig. 4 Shows the relative tolerances of different types of annoyances, and fig. 5 averages this data over different types of annoyances.

Such a survey is useful to determine the price tradeoffs involved in delivering video service with a particular range of *MTBF* values.

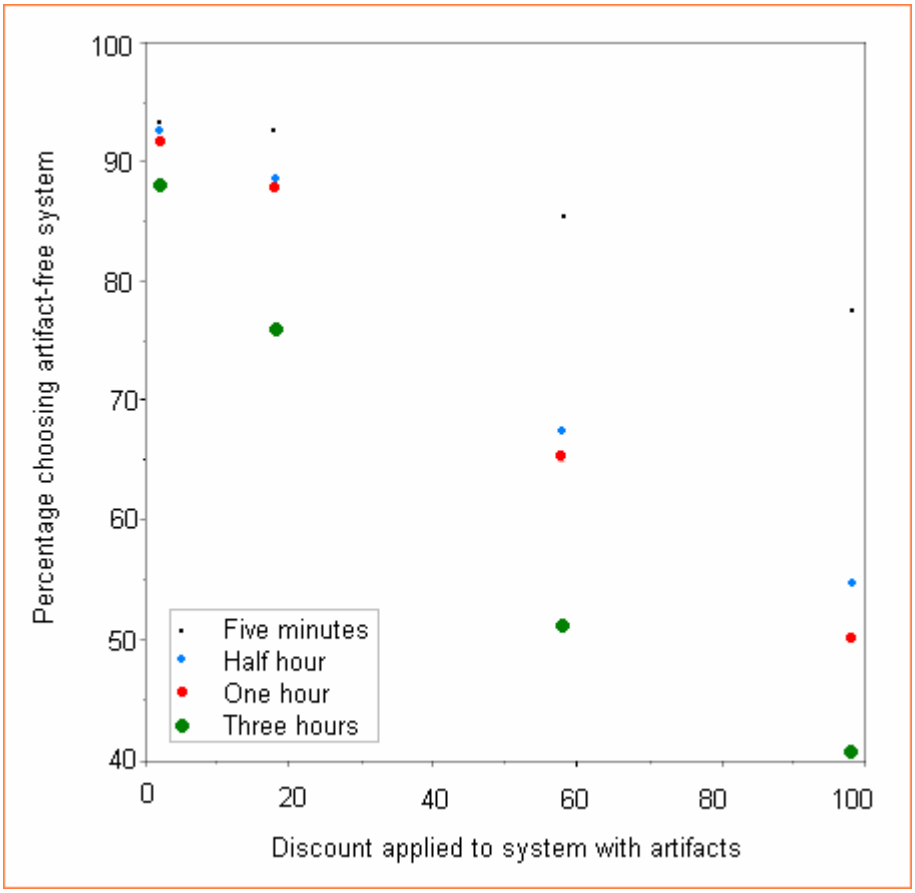


Fig. 2: Percent of respondents choosing the artifact-free system when offered a choice of a system with digital artifacts at four frequencies and four price discounts.

Discount levels in the figure are a linear transformation of the actual numbers used in the survey.

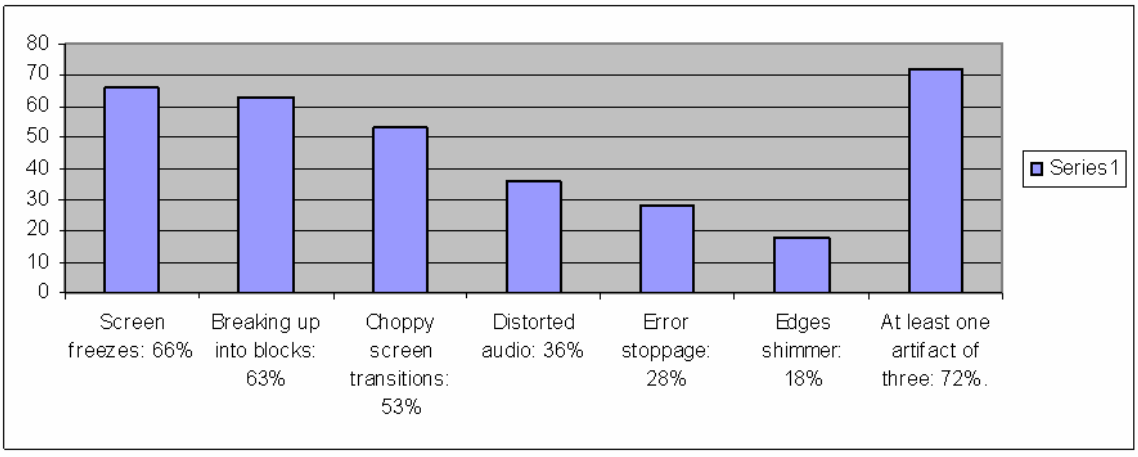


Fig. 3: % of respondents reporting that they had experienced a given artifact at some time

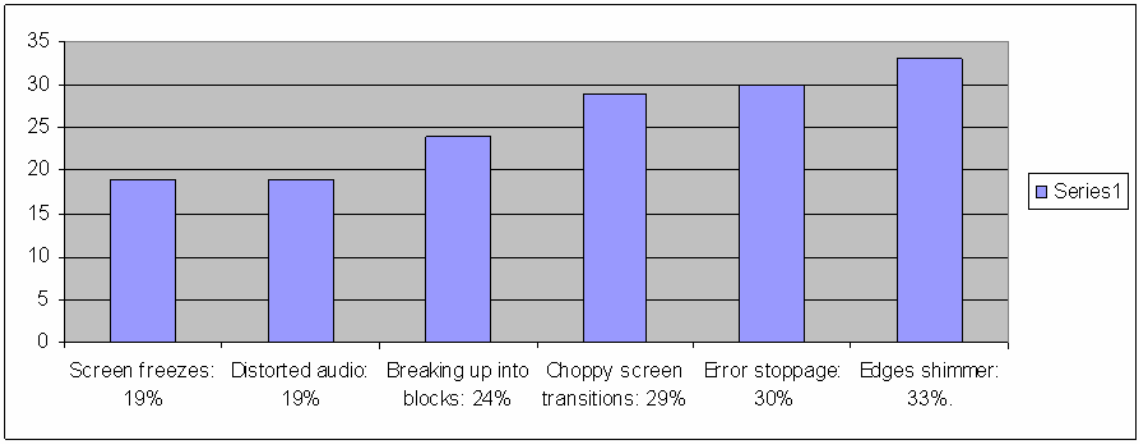


Fig. 4: % of respondents choosing a given artifact type, averaged across frequencies and price discounts

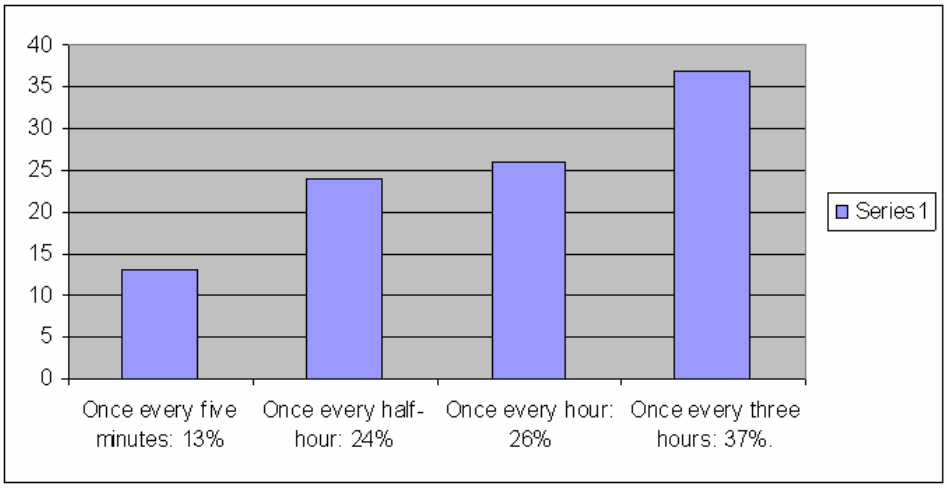


Fig. 5: % of respondents choosing an artifact frequency, averaged across artifact types and price discounts

CHAPTER IV

Standardized Video Testing

The Video Quality Experts Group (VQEG [2]) is a group of experts from various backgrounds and affiliations, including participants from several internationally recognized organizations, working in the field of video quality assessment. Four groups are formed under the VQEG: Independent Labs and Selection Committee, Classes and Definitions, Objective Test Plan, and Subjective Test Plan. The purpose of the subjective test plan is to provide data on the quality of video sequences and to compare the results to the output of proposed objective measurement methods. This test plan provides common criteria and a process to ensure valid results from all participating facilities.

4.1 Double Stimulus Continuous Quality Scale (DSCQS)

The Double Stimulus Continuous Quality Scale (DSCQS) is used by the VQEG. The DSCQS method presents two pictures (twice each) to the assessor, where one is a source sequence and the other is a processed sequence (fig. 6). A source sequence is unimpaired whereas a processed sequence may or may not be impaired. The sequence presentations are randomized on the test tape to avoid the clustering of the same conditions or sequences. Participants evaluate the picture quality of both sequences using a grading scale (DSCQS). They are invited to vote as the second presentation of the second picture begins and are asked to complete the voting before completion of the gray period after that.

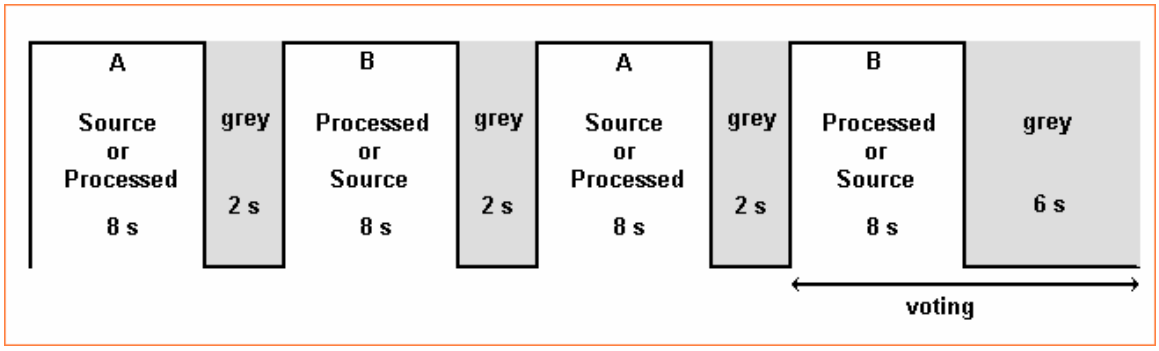


Fig. 6: DSCQS method

The DSCQS method consists of two identical 10 cm graphical scales which are divided into five equal intervals with the following adjectives from top to bottom: Excellent, Good, Fair, Poor and Bad. (Note: adjectives are written in the language of the country performing the tests.) The scales are positioned

in pairs to facilitate the assessment of each sequence, i.e. both the source and processed sequence. The viewer records his/her assessment of the overall picture quality with the use of pen and paper or an electronic device (e.g. a pair of sliders). The 'difference mean opinion score' (DMOS) is the calculated as the difference between the source and the processed scores, averaged over all the users. DMOS is used as the subjective score for the video clip.

4.2 Test material

The VQEG uses tests in the 50 Hz and 60 Hz format. Different factors are taken into account in the selection of the tests. The tests have a variety of color including different skin colors and saturated colors on moving objects. Different levels of brightness are also incorporated. Some of the test clips have moving text (scrolling both in the horizontal and vertical direction), selective zooming, disappearing objects, camera panning and multiple scene cuts. Test conditions span the whole quality range, and scene content either facilitates or masks certain forms of degradation when present (e.g. flat areas, complex patterns, square patterns, water motion, broad range of sequences). Care is also taken to ensure that the tests are culturally neutral and gender unbiased.

4.3 Objective measurements

The VQEG has conducted some evaluations of objective metrics by correlating them with subjective scores [21] measured on test sequences that are subject to different hypothetical reference conditions (HRCs) such as MPEG-2 or H.263 coding at different bit rates. The choice of HRCs ensures that visible picture cropping, chrominance / luminance differential timing, jitter or spatial scaling does not occur. The performance of the objective metrics is characterized by a number of attributes such as prediction accuracy, monotonicity and consistency. The relationship between the objective and subjective scores need not be linear because the subjective testing can have nonlinear quality rating compression at the extremes of the test range. A monotonic non-linear regression is fit between the objective and subjective scores, and the Pearson linear coefficient calculated on this relationship indicates how good or bad the objective metric is. Based on the subjective and objective tests conducted by the organization, the VQEG lists the performances of some objective metrics in [21]. The search for good objective metrics that correlate well with subjective scores is still a topic of interest for many companies and research institutes.

CHAPTER V

Current Metrics and Approaches to Objective Video Testing

A comprehensive understanding of existing objective metrics and the strengths and weaknesses of these metrics has helped in making our Automatic Video Quality metric (AVQ) efficient and subjectively relevant. Objective metrics can be broadly classified based on the amount of information available about the original video. Peak Signal-to-Noise Ratio (*PSNR*) and *JND* [3] are *full-reference* metrics in the sense that they need the complete original signal. *No-reference* and *reduced-reference* metrics are considered to be more practical than *full-reference* metrics since the original video is in general not available at an arbitrary place of quality evaluation such as a network node or the ultimate receiver.

For e.g., some *no-reference* metrics estimate the block distortions introduced in compression algorithms [5, 22, 23], while some metrics estimate the blurring/sharpness in the video. There are a few papers in literature that describe these various metrics [24]. *Reduced-reference* metrics use some prior knowledge about the original video. For e.g., a watermark might be introduced in the original, and its distortion in the processed video could be used to estimate the video quality. Alternatively, a specific signature could be calculated over the original and transmitted along with the processed video. The same signature could be calculated over the processed video and compared with the original signature to determine the quality [4].

Current metrics in literature evaluate different aspects of visual artifacts such as blocking, blurring, ringing, Gaussian noise, etc., and compute the overall metric as a linear combination of the individual components. The ideas in the current patent pool and literature are described below to get an understanding into the working of the AVQ metric.

5.1 Peak Signal to Noise Ratio (*PSNR*)

The *PSNR* metric is based on a simple difference measure. The root mean squared error (*RMSE*) is calculated between the corrupted and the original frame, and the *PSNR* is calculated as:

$$PSNR = 20 \cdot \log_{10}\left[\frac{255}{RMSE}\right] \tag{2}$$

PSNR has been a de-facto industry standard for quite a while, because it is simple yet effective. The drawbacks of the metric are that it does not account for any spatial or temporal masking. Some of the other objective metrics introduced later incorporate some form of visual masking. *PSNR* is in general calculated for the luminance values alone, as is the case for many other objective metrics. The *PSNR* values for all the frames in a test clip are averaged to give a *PSNR* score for the clip.

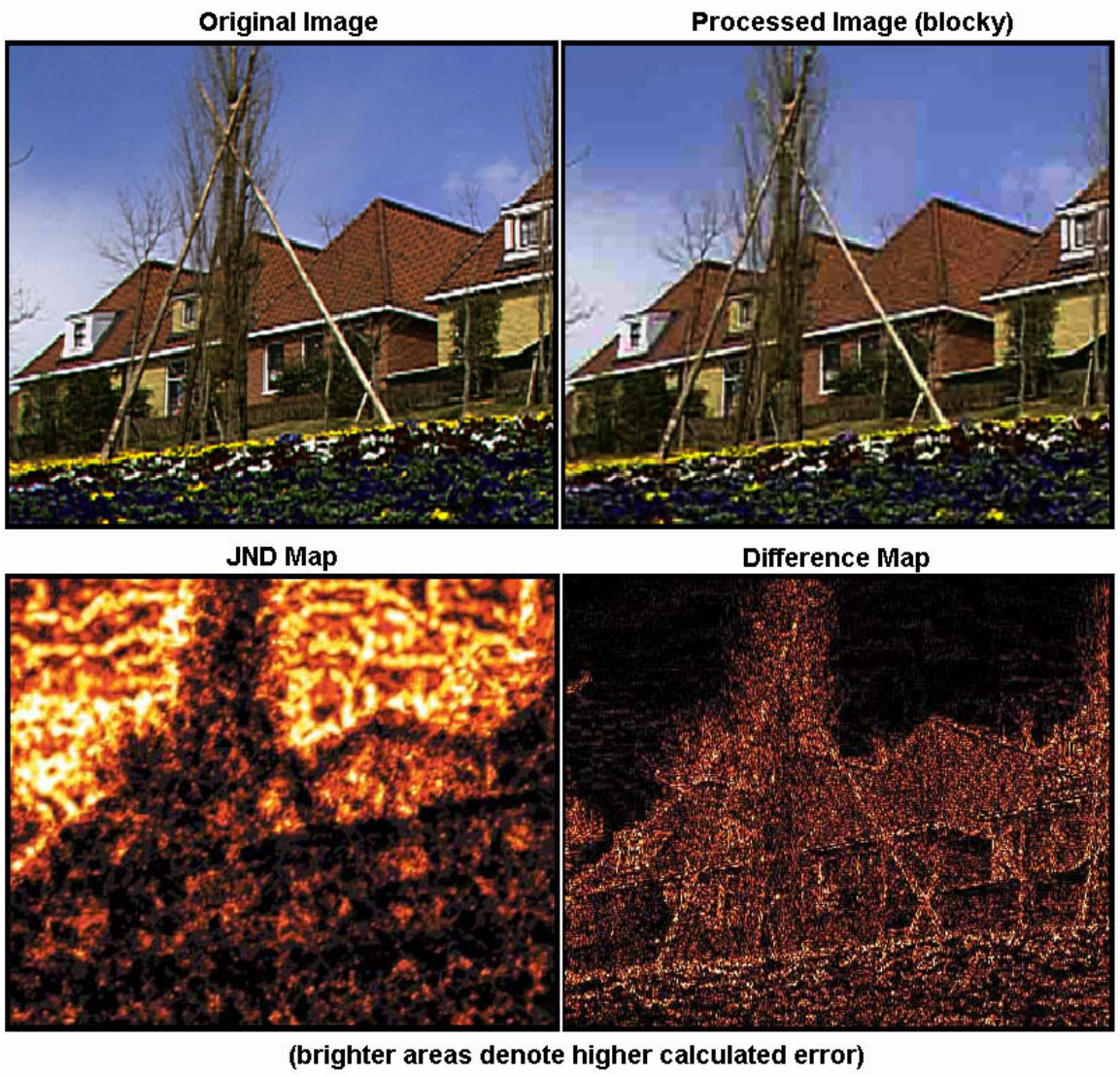


Fig. 7: Effectiveness of *JND* over *PSNR* as depicted in [3]

5.2 Just Noticeable Difference metric (*JND*)

The *JND* metric [3] is based on a weighted difference measure. The JND scale used in the metric is explained in [25]. Two video sequences are said to be 1 JND scale apart if 75% of the viewers observe that these two are of different quality. The effectiveness of *JND* over *PSNR* can be understood from fig. 7. The notion of perceptually important areas in an image is well studied [26]. *JND* gives more importance to the difference measure in the clear blue skies, because blocking artifacts are visually more pronounced in that region. The flower & garden area masks many of the artifacts because the amount of spatial information present. Since *PSNR* and *JND* require full knowledge about the original image, they are termed as *full-reference* metrics.

5.3 Spatial-Temporal 'Join' Metric (*STJM*)

The *STJM* metric [4] is based on the principles from the *VQM* metric [7] designed by the Video Quality Evaluation Group (VQEG). Certain feature are extracted from spatial-temporal blocks over the original video and transmitted to the receiver for comparison with similar features extracted from the processed video. The nature of the spatial-temporal blocks can be understood from fig. 8.

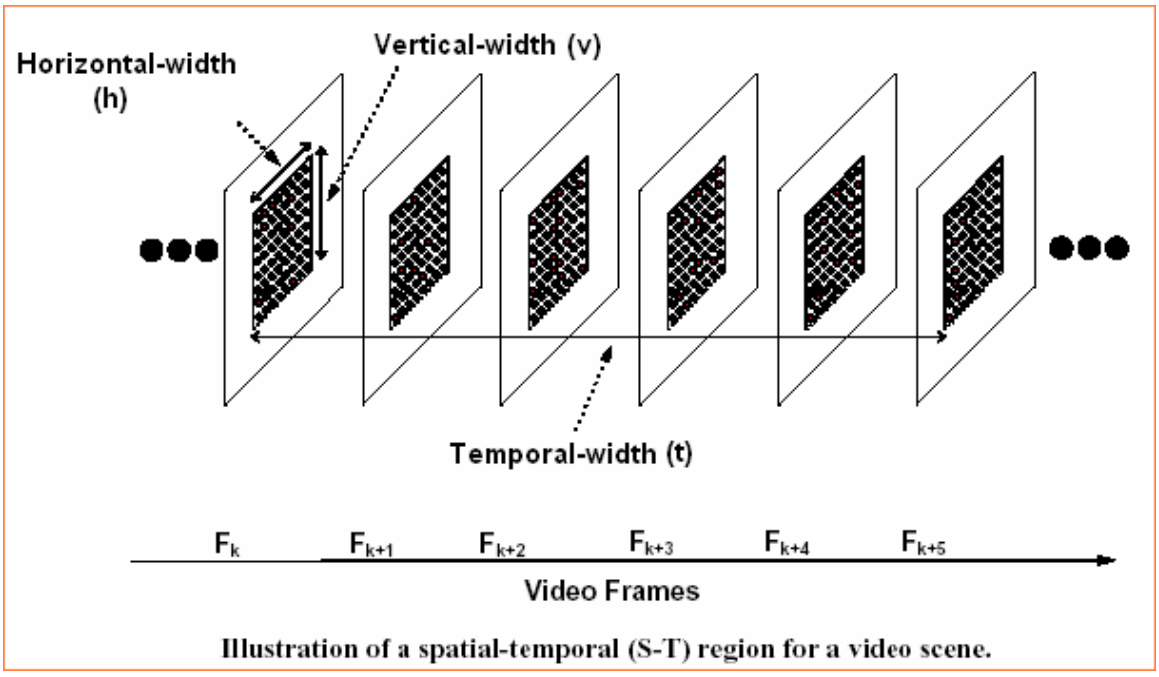


Fig. 8: Spatial-Temporal blocks of the *STJM* metric

The video frames are first treated with horizontal and vertical edge filters and then converted to polar coordinates. Two features are compared in calculating the objective scores. The first feature is the amount of spatial information in the spatial temporal block, which is obtained by calculating the standard deviation of all the pixel values in the block. An increase in this feature value corresponds to addition of noise and a decrease corresponds to smoothing due to the nature of block coding. The second feature calculates the amount of horizontal and vertical edges present. An increase in this feature value corresponds to the presence of block artifacts. When the video is extremely compressed, many adjacent blocks are smoothed out, and this leads to a decrease in this feature value. The various effects of the increase and decrease of features of corresponding spatial temporal blocks are pooled to get the *STJM* values. The different features used in the calculation of such a *reduced-reference* metric are described in [27]

5.4 Blockiness metric (*BLK*)

Some *no-reference* metrics require access to the output bit-stream, since they use values such as bit-rate and quantization values to calculate the metric value. The *BLK* metric [5] is a *no-reference* metric that uses just the output pixel values of the video. Blockiness and ringing artifacts are considered to be the most annoying visual artifacts [28], with blockiness being the most visually annoying for lower bit rates. The BLK metric concentrates on the measure of blockiness. The ratio of the difference across block boundaries to the difference across non-block boundaries is calculated. Instead of using a simple difference measure, a weighted difference based on the spatial and luminance information is used to get the final metric value. The weighted block differences are calculated in both the horizontal and vertical directions, and the two are combined in equal proportions.

5.5 MSU toolbox

Apart from the various metrics and approaches towards objective testing listed in this section, there are also online tools that make their calculation easy. The MSU toolbox [29] is one such useful application. It is a *full-reference* metric tool, and provides metric scores in standard units such as PSNR, MSE and VQM.

5.6 Vlachos blockiness metric

This metric [22] works by comparing the correlation between pixels across block boundaries and pixels within the DCT-coded blocks. The image as such is down-sampled at various offsets (fig. 9), and the correlations between different offset images are calculated for the metric value. For instance, the correlation between the image down-sampled at (7,7) and the other three down-sampled images provides a measure of similarity for picture material across block boundaries. Intra-block similarity can be similarly measured, and the blockiness metric is calculated as the ratio of these two similarities.

Modifications of this metric have also been proposed [30]. This metric works on the assumption, that blocking artifacts have a visible corner. An estimate of blurriness is also estimated based on the detection of edges and the calculation of edge width. It has to be noted, that such an approach would incorrectly detect intended hazy edges as blurriness artifacts. Such video frames do occur frequently, and one such still a video named 'Fries' is shown in fig. 35.

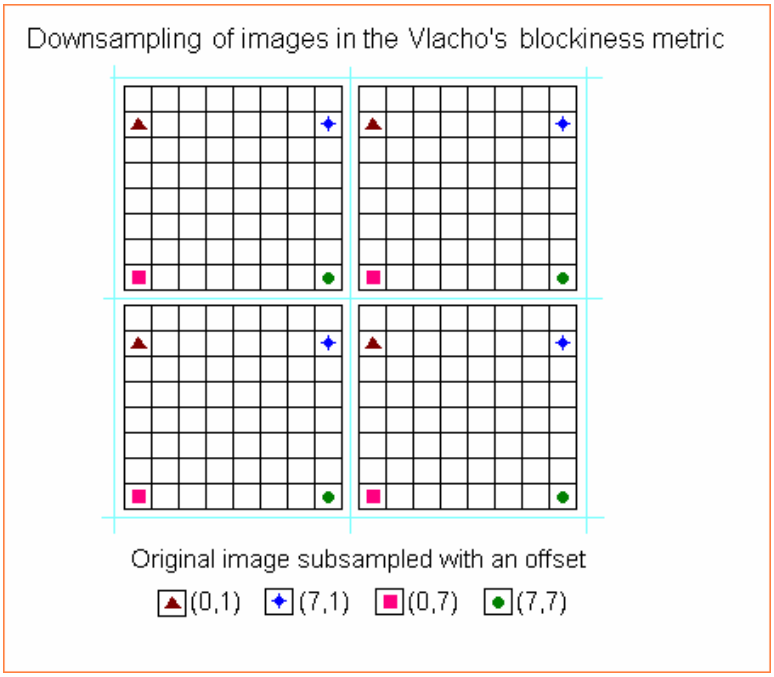


Fig. 9: Different down-sampled images of the original used in metric calculation [22]

5.7 Harmonics Phase Blockiness metric

This metric [31] uses both the amplitude and phase information generated by the block artifacts in MPEG-2 coded video. This is claimed to overcome the problem of misinterpreting contextual detail as blockiness. The same video team also has published literature on a *full-reference* blockiness metric [32]. This metric simulates both DSCQS and SSCQE scores.

5.8 Picture Appraisal Rating (PAR)

Similar to the DVQ product is the PAR metric from Snell & Wilcox [33]. PAR also takes the MPEG stream as input, and estimates the Peak Signal to Noise Ratio (*PSNR*) from available quantization data [34], [35]. The metric makes use of Parseval's theorem, and uses the notion that the quantization error in the transform domain is related to the error in the pixel domain. The formula for the estimated error is derived to be a function of the quantization scale used, the decision threshold offset parameter used in the codec, and the probability distribution of the original signal. A typical value for the decision threshold offset parameter in MPEG-2 is 0.75, and is used in the calculation of the PAR metric. Since the original signal and its probability distribution is unknown, an alternate approach is followed which involves modeling the DCT coefficients as a Laplacian distribution. Having understood that a single parameter may suffice to account for the dependence of quantization noise power on the probability distribution of the source data, it is observed that it is unnecessary to impose a particular shape such as a Laplacian on the data and then attempt to fit the data to the model. Instead, the system could take a representative set of source data and directly observe the dependence of quantization noise power on the quantization scale and a measure of picture activity. It is observed that the quantization scale in use is by itself a good indication of video quality.

The measure of picture activity is got from the number of DCT coefficients that would be used if the picture had been coded with as low a quantization scale as possible, and this number is in turn got as a quadratic function of the current quantization scale and the number of DCT coefficients in use. Some of the modifications to the algorithm mentioned in their work refer to the incorporation of bit rate to improve correlation and the functioning of the metric as a reduced reference measure. There is no mention to the effect of network error artifacts, however. While the PAR makes a note of the Laplacian nature of the

probability distribution function of the DCT coefficients, it does not actually implement it fully. However, there are some other metrics in literature that do this modeling exhaustively [36].

5.9 Philips Electronics

Similar to the work by Snell & Wilcox, Philips Electronics has some intellectual property in video quality evaluation using coding parameters [37]. The coding parameters are estimated from the output pixel values using statistical operators [38]. The quantizer step size for each block of coded video data is estimated, and following this the statistical properties of the DCT coefficients are extracted. It has to be noted that in this and some other *no-reference* metrics in literature, the scores are simply correlated with an established *full-reference* score like PSNR for ease of evaluation purposes.

5.10 NROQM

Another *no-reference* metric by Philips is the no-reference objective quality metric (*NROQM*) [10] that measures video quality as a weighted measure of blocking artifacts, ringing artifacts, luminance and chrominance clipping, noise, contrast and sharpness. The algorithms involved in these calculations are described in [39]. The metric involves an elaborate training method to tune the metric. At first, the six parameters mentioned above are calculated for the entire test set. The test set is partitioned into (i) coded (include coded-enhanced), (ii) noisy (include noise-enhanced), and (iii) original and original-enhanced sequences. The coded and noisy sets are further refined into sets that exhibit variations predominantly in one specific feature, like blocking or ringing. The values of the feature of interest are fit to the subjective data in each of the earlier step through a perceptual function. This function maximizes the correlation between feature values and subjective scores. The dynamic range and spread of feature values are checked in order to identify the weaknesses of the feature metrics or the training set, in order to improve them or introduce temporary workarounds.

Each set is expanded by including sequences whose quality is influenced by more than one feature, and it is checked if they are additive. Interactions such as masking or facilitation are observed as well, and the fitting objective is refined using these observations. The set of (original) enhanced video sequences are considered and the objective data is fit to the subjective data. The effect of sharpness and the

combined effects of contrast and noise, if any, is accounted for. Then, the fitting of objective data to subjective data is performed for coded-enhanced and noise-enhanced sequences. This takes care of scaling factors and interactions not considered in earlier steps. Finally, the entire test set is considered, and the fitting functions found so far are merged. Based on the review of the overall fitness of the model to all subjective data, the entire process is iterated if necessary.

5.11 Structural Similarity Index Metric (*SSIM*)

The (*SSIM*) [40] is a *full-reference* metric that compares the luminance, chrominance and shape information between the input and output. This metric works under the assumption that human visual perception is highly adapted for extracting structural information from a scene, and offers an alternative complementary framework for quality framework based on the degradation of structural information. The metric algorithm can be expressed by the formula in equation 3. For given a pair of the original and degreaded image (x,y),

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (3)$$

where μ and σ are the mean the standard deviation of the image, and C_1 and C_2 are constants used to prevent unstable results. The standard deviation of an image is used as the estimate of its contrast.

Instead of comparing entire frames between the input and output, the algorithm divides each frame into smaller block sizes for comparison. The results are pooled over the frame to generate a metric value. This metric is in development by the University of Texas, Austin, and is evaluated against a test database of JPEG and JPEG2000 processed images. A comparison between the SSIM calculation and a simple Mean Square Error measure can be understood from a difference map (fig. 10), similar to the difference map for the JNDmetrix algorithm (fig. 7).

They also have some work published on an MPEG-2 *no-reference* metric [41] that works on similar principles to that of the Picture Appraisal Rating (PAR) measure [34]. While the latter uses some approximations to the relationship between quantization noise in the pixel and transform domains, the *no-reference* measure from the University of Texas uses the actual probability distribution function of the DCT coefficients to estimate the error. Another *no-reference* metric developed by the University of Texas

involves measuring the blocking artifacts by modeling an image as a non-blocky image interfered with a pure blocky signal [42].

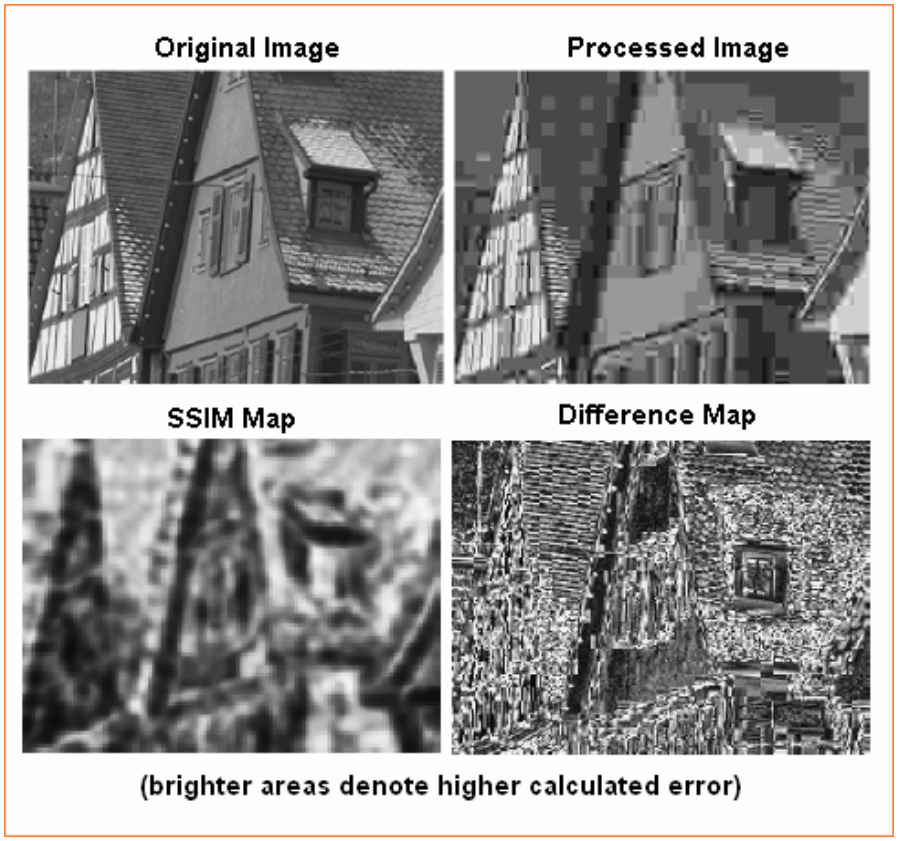


Fig. 10: A comparison between the SSIM and difference maps in [40],
to illustrate the usefulness of a structural similarity measure

5.12 AT&T Labs Research

AT&T has done some significant research concerning video quality issues as applicable to MPEG-2 video streamed over a lossy network [43], [44], [45], [46] and [47]. The Mean Square Error caused by packet loss is estimated by just looking at the received bit-stream. The errors caused by different packet losses are assumed to be uncorrelated, and a typical bursty packet loss scenario is not simulated. Packet losses are randomly injected into the MPEG2 slices. Based on the visibility of these losses, a classifier algorithm is used to find the visibility of a packet loss based on certain stream parameters. The algorithm is categorized into different profiles depending on the amount of information available. The 'Quickparse' method extracts only picture and slice start codes and the corresponding headers, including the slice location and quantizer.

The ‘Fullparse’ method uses full pixel information and motion vectors to determine error propagation between frames.

The term Visible Packet Loss Rate (VPLR) is suggested to be more useful than a simple Packet Loss Rate (PLR) measure for lossy transmission of video, in order to determine the conditions under which a packet loss is found to result in a visible artifact. Importance is given to the observation of the amount of motion and the accuracy and consistency of motion prediction. It is expected that a high motion residue indicates poor visual quality, and so does a large temporal variance in the amount of motion. The Mean Square Error between an error free bit-stream and a lossy bit-stream is also used as an input to the classifier algorithm for determining the visible packet loss rate.

5.13 NR_VQM Blocking Strength

This metric [48] is similar to the blockiness metric as described in [5], except that it does not assume any knowledge about the block boundaries. The boundaries where the pixel differences are a maximum are considered to be the block boundaries, and an estimate of the blockiness thus calculated is used to adaptively suppress blocking artifacts while preserving the sharpness of existing edges.

In order to express the similarity between the local gradient and its spatial neighbors, a normalized horizontal (/vertical) gradient is introduced as the ratio of the absolute gradient and the average gradient calculated over a certain number of adjacent pixels to the left and to the right (/above and below). The suppression of block artifacts is done by the means of adaptive spatial low pass filtering. To optimally preserve the image sharpness, object edges are distinguished from block discontinuities, such that low pass filtering is applied only to those pixel positions where block artifacts are visible.

5.14 NR_VQM

This *no-reference* metric uses the previous frame or the motion compensated frame as a point of reference for metric calculation [49]. Taking account of the temporal dependency between adjacent images of the videos and characteristics of the human visual system, the spatial distortion of an image is predicted using the differences between the corresponding translational regions of high spatial complexity in two adjacent images, which are weighted according to temporal activities of the video.

Since the human visual system can tolerate the distortions in fast-moving regions to a considerable extent, different weightings are applied to the measured spatial distortions of the image according to temporal activities of the video, which are computed as the mean value of the motion vectors in the image. The overall video quality is measured by pooling the spatial distortions of all images in the video

5.15 Distortion Assessment Model (DF_{IMAGE})

This metric [50] by the Institute for Infocomm Research (I2R), Singapore, works under the assumption that a blocky image has many edges that are close to the horizontal and vertical direction. The histogram of edge angles is observed, and if there are more edges closer to 0 and 90 degrees than there are edges at other angles, then the frame is considered to be blocky. The I2R organization also uses a simple blocking measure to determine the locations of network artifacts [51]. The I2R also has some literature on their *full-reference* implementation [52]. This metric has a comprehensive pooling formula over the various color components and also computes its score in terms of distortion invisibility, block fidelity and content richness.

5.16 Continuous Video Quality Evaluation (CVQE)

The CVQE metric [53] is a *full-reference* metric that emphasizes on an error pooling scheme that gives more importance to recent frames. The error pooling scheme is intended to be an improvement over current error pooling schemes in existing metrics, such as [54].

5.17 Tektronix

The PQA300 from Tektronix [55] and VP200 from Pixelmetrix [56] are full reference metrics, which require the processed video and the original as inputs, and calculate quality scores based on the pixel-per-pixel differences on the perfectly aligned images. Tektronix's approach towards objective testing is explained in [57]. They also have a *no-reference* implementation, the PQM300 [58]

5.18 Digital Video Quality Analyzer (DVQ)

This is a *no-reference* metric [59] developed by Rohde & Schwarz [60]. This device measures quality of encoded video based on a blocking artifact metric corrected for motion and spatial masking (conditions that may hide the impairment). The DVQ does not take into account other MPEG artifacts besides blockiness, does not work on the decoded image (the input is the MPEG transport stream), and is not able to measure picture improvements such as sharpness enhancement or noise reduction. Some of the results of their metric are listed in [61].

5.19 IneoQuest

Internet Protocol TeleVision (IPTV) is an important focus of video quality evaluation. IneoQuest, an IPTV company, uses their metric 'Media Delivery Index' (MDI) [62]. MPEG video transport streams undergo time distortions known as jitter when being transported by packet switched networks such as Ethernet. Identifying and measuring jitter and packet loss in such networks is the key to maintaining high quality video delivery. The Media Delivery Index is a set of measurements used for monitoring and troubleshooting networks carrying any streaming media type. The MDI can be used to warn or alarm on impairments that result in unacceptable video delivery and on conditions that result in unacceptable network margin before video quality is impacted.

The MDI has two components: the Delay Factor (DF) and the Media Loss Rate (MLR). The DF is computed at the arrival time of each packet at the point of measurement and displayed and/or recorded at time intervals, which are typically a second apart. A given virtual buffer level X is measured as the difference between the bytes received and the bytes drained. The DF is calculated as the ratio between the range of X and the media rate in bytes per second. The MLR is computed by subtracting the number of media packets received during an interval from the number of media packets expected during that interval and scaling the value to one second MLR reflects the number of media packets lost per second The use of the MDI claims to provide a network margin indication that warns system operators of impending operational issues with enough advance notice to allow corrective action before observed video is impaired.

5.20 V-factor

Spirent, another video testing company, uses their objective measurement tool named V-factor [63]. Their V-factor score is based on the 'Moving Pictures Quality Metric' (MPQM), and is claimed to better than Ineoquest's MDI score [64].

The MPQM model is based on an objective evaluation method that is capable of filtering network-induced quality degradation by properties of human vision. It also considers visual masking techniques to effectively take into account error concealment techniques in the Quality of Service result and targets moving picture quality rather than image quality. It rates the delivered video quality in a standard five point scale from 5 (The index may be extended with a 0 value to denote a zero communication value). MPQM is an objective measurement system that claims to reproduce the subjective experience of the human observer. It does so without having directly emerged from fitting or weighting any known subjective results. Instead it is based on modeling of defining elements of human vision

In MPQM, five spatial frequencies, four orientations and two temporal frequencies are used to decompose the signal. After the suggested decomposition, the perception model is applied, i.e. contrast sensitivity and masking filters, and a distortion measure is computed. The metric also accounts for a focus of attention in time (persistence of images in retina) and space (visual angle). These computations are performed in three-dimensional (temporal and in two-dimensional space) blocks of the video sequence and are mapped to the quality rating scale mentioned above. The MPQM is claimed to better than the MDI metric, because the latter is supposed to be a network-based tool that does not take into account the relative significance of each packet for the delivered video content. MPQM attempts to asses the video quality by analyzing the temporal degradation effects of the encoding scheme. The Spirent product measures different quality parameters. Some of the key parameters measured are: Transport stream rate, jitter of synchronization stream, number of frames discarded due to jitter, number of mis-ordered frames, network loss probability, maximum number of frames lost per episode, relative ratio of I-frames, and overall user-perceived video quality in terms of a unit termed the V-factor.

5.21 Absolute Temporal Information (ATI) Metric

The International Telecommunication Union (ITU) has designed some *reduced-reference* metrics to evaluate loss in video quality due to network errors [65]. For instance, the differences between consecutive frames can be thought of as a video signature. This signature is calculated over the original and sent along with the video stream to the receiver. The receiver can compare the original and processed signatures to arrive at certain quality measures. A positive shift in the signature would indicate the addition of noise to the video, while a negative shift would indicate a smoothening process in transit. The presence of network errors can be detected from peaks in the received signature that are not found in the original (fig. 11)

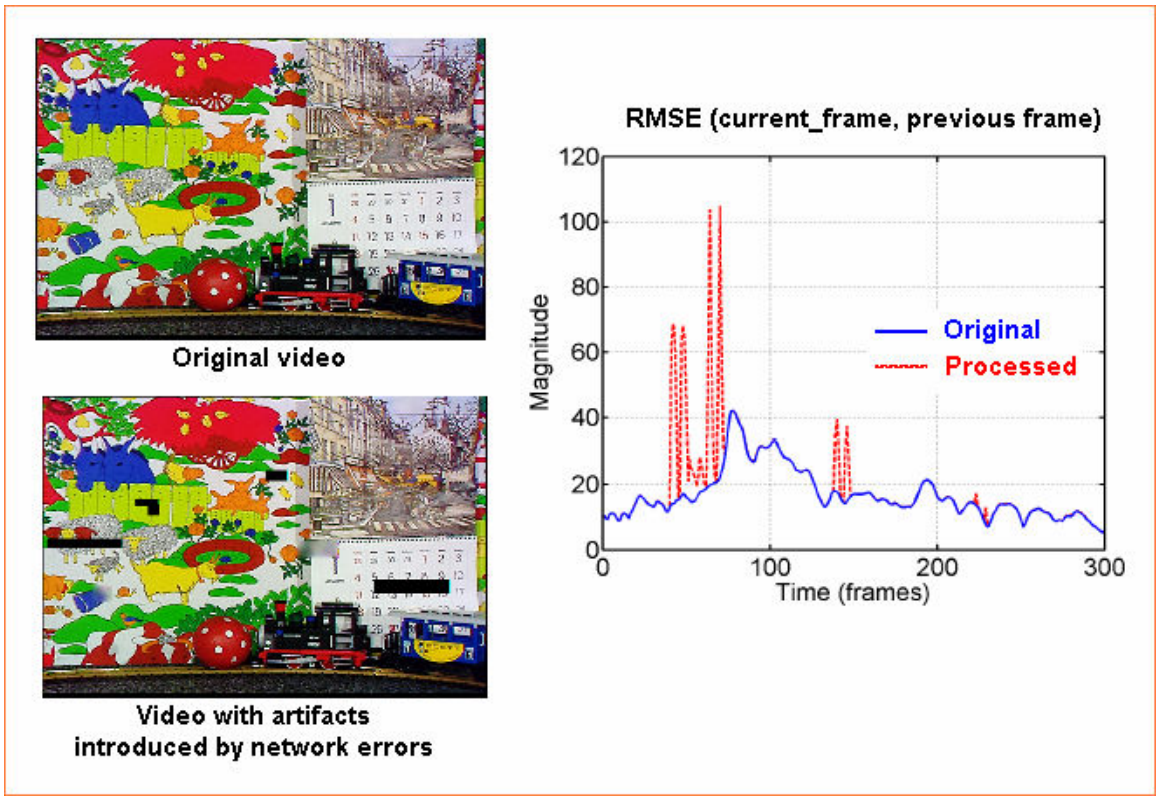


Fig. 11: A *reduced-reference* implementation of detecting network errors

5.22 Relative Peak Signal to Noise Ratio

A study of real-time monitoring of video quality over IP networks is performed in [66]. The effects of loss distortion modeling, impact of video content, packetization and codec selection are studied as well.

The effects of packetization are two-fold. First, the number of slices contained in a packet, together with the number of packets used for transmitting a frame, affects the mapping from packet losses to slice

losses. Second, video streams with different values of sample network paths experience different packet-level loss processes even when transmitted on the same path. For instance, video streams configured with a larger sample network path tend to see longer packet loss bursts than those with a smaller value.

A codec based study is needed due to the different types of error concealment is use. In the MPEG-2 codec, packet losses are handled in relatively simple fashion in their study. If the MPEG2 decoder detects any number of packet losses in a frame, it discards the entire damaged frame and replaces it with the frame previously decoded. The H.264 codec employs more sophisticated error-concealment techniques. All slices in the received packets are decoded and the slices contained in the lost packets are recovered using the corresponding slices in the previous frame and the motion-compensation information of the other slices in the same frame.

Their objective scores are measured in terms of relative PSNR (rPSNR), which is a metric calculated against a quality benchmark that the network is expected to provide.

5.23 Streaming Video Quality Parameters

The study of streaming video quality can also be understood in terms of some useful metrics. Average service rate, number of re-buffering events, average re-buffering time and number of missed packets are listed as metrics that could complement the use of traditional metrics like *PSNR* and *MOS* [67].

The study of these and other objective metrics in literature, along with a good understanding of the various kinds of subjective metrics has helped in the design of enhanced objective video quality metrics at Georgia Institute of Technology.

CHAPTER VI

Subjective Experiments to Measure *MTBF* and *MOS*

The study of the relationship between different subjective metrics, and the correlations between objective and subjective metrics require an extensive test database. The first stage of our experiments included a compression -artifacts alone database, and later, we included sequences that had both compression and network artifacts. Some standard VQEG Sequences were used for our video quality experiments: 18 clips (*Tree, Barcelona, Music, EBU_Test, Rower, Race, Fries, Moving_Text, Rugby, Park, Building, Mobile, Cartoon, Waterfall, Football, Susie, Flower_Garden* and *BBC_Disk*) (fig. 12) were concatenated to form a 140 second sequence.

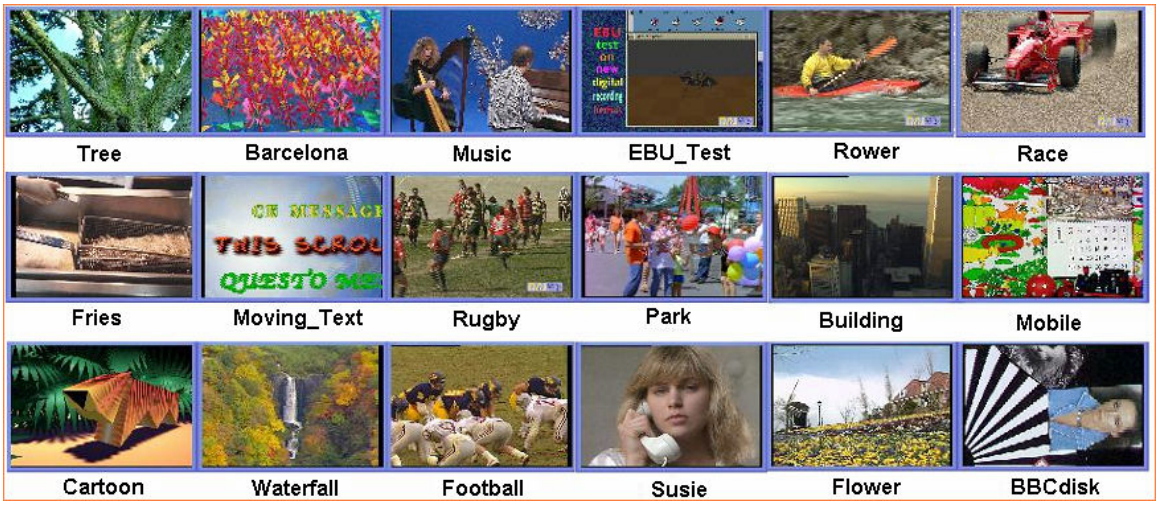


Fig. 12: VQEG clips concatenated to form the ~140 seconds test sequence

The VQEG clips were originally chosen based on the variety of content, spatial and motion information. Some sequences, such as *Cartoon* and *Susie* have low spatial detail in the frames. Sequences like *Flower* and *Mobile* have a lot of spatial variety in them. While sequences like *Rower*, *Rugby* and *Football* have a lot of motion present, sequences like *Tree* and *Waterfall* show near stationary video. The variety in content across the different video clips is also representative of the different types of content encountered in typical video data observed by users. Apart from the video database in the VQEG site, there are other image quality databases as well. For instance, the "LIVE Image Quality Assessment Database" [68] provides different images at different qualities. The VQEG test clips were cropped appropriately, so that all of them

have a uniform size and could be concatenated to form a test sequence. Such concatenated VQEG test clips are regularly used as benchmarks by current video encoder companies, like [69].

This was encoded at different bit rates using a publicly available MPEG-2 encoder [70]. The database having compression artifacts (but no network artifacts) had this sequence encoded at different bit rates (1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5 and 5.0 Mbps) (test database 1). The level of compression used in cable video falls within this range, and is typically in the 2-4 Mbps range. For the second test set (test database 2), the original sequence was subject to different network- and compression- error conditions (Table 1). Compression artifacts were obtained by using a publicly available MPEG2 encoder. Packet losses in an Internet Protocol Television (IPTV) scenario are bursty in nature, and can be modeled using a Gilbert model [71]. Network artifacts were introduced by streaming the MPEG2 streams across a simulated link that incorporated a two-state Gilbert model for corrupting packets (figs. 14, 15). The MPEG2 player used to play these streams incorporated an error concealment algorithm, and depending on the occurrences of contiguous packet losses, frame/slice freezing or block/line errors were observed at the receiver. To ensure that the videos shown to the viewers were consistent, and that the error resilience did not depend on the computing performance during playback, the output frames were first stored and then played back to the viewers.

The sequences were shown on a 30-inch Television screen in a dark room, with the viewer sitting at a distance of 5 times the height. A simple MATLAB script was used to allow the user to indicate the occurrences of visual errors (fig. 16, 17). Pilot experiments were conducted to estimate the delays between the occurrences of artifacts and the viewer's responses. The videos with different quality settings were shown in a random order to the viewers. Using the feedback from the user, one can determine the time locations corresponding to the artifacts that were perceived. This feedback vs. time plot for a viewer looks like an impulse train. This is low-passed filtered with respect to time using a Gaussian curve to account for the viewers' inaccuracy in pinpointing the time of the error. The smoothed feedback vs. time plot for various viewers is averaged to get the failure-rate characteristics (fig. 18). A broad range of viewers were chosen with respect to their video expertise. A total of thirty viewers were tested. Different temporal smoothening filters are used to demonstrate that the failure rate characteristic does not critically depend on the filter's width (fig. 19). It has been estimated in [46] that a viewers typical response time is around half a

second. The *MTBF* of the sequences is calculated as the reciprocal of the average failure rate. The mean of *MTBF* is calculated as the mean of *MTBF* over different clips.

Table 1: Different test conditions which the test sequence is subject to

Condition	Bit rate (Mbps)	Packet loss rate (%)
A	2.0	2.0
B	5.0	0.4
C	2.0	1.0
D	3.5	0.4
E	3.5	0.9
F	5.0	0.1
G	<original>	<none>

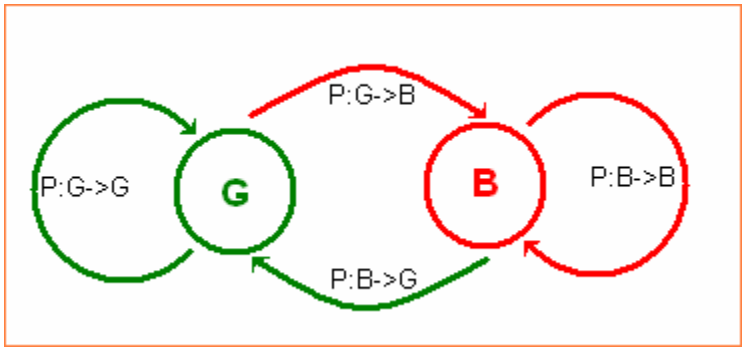


Fig. 13: A two-state Gilbert Model

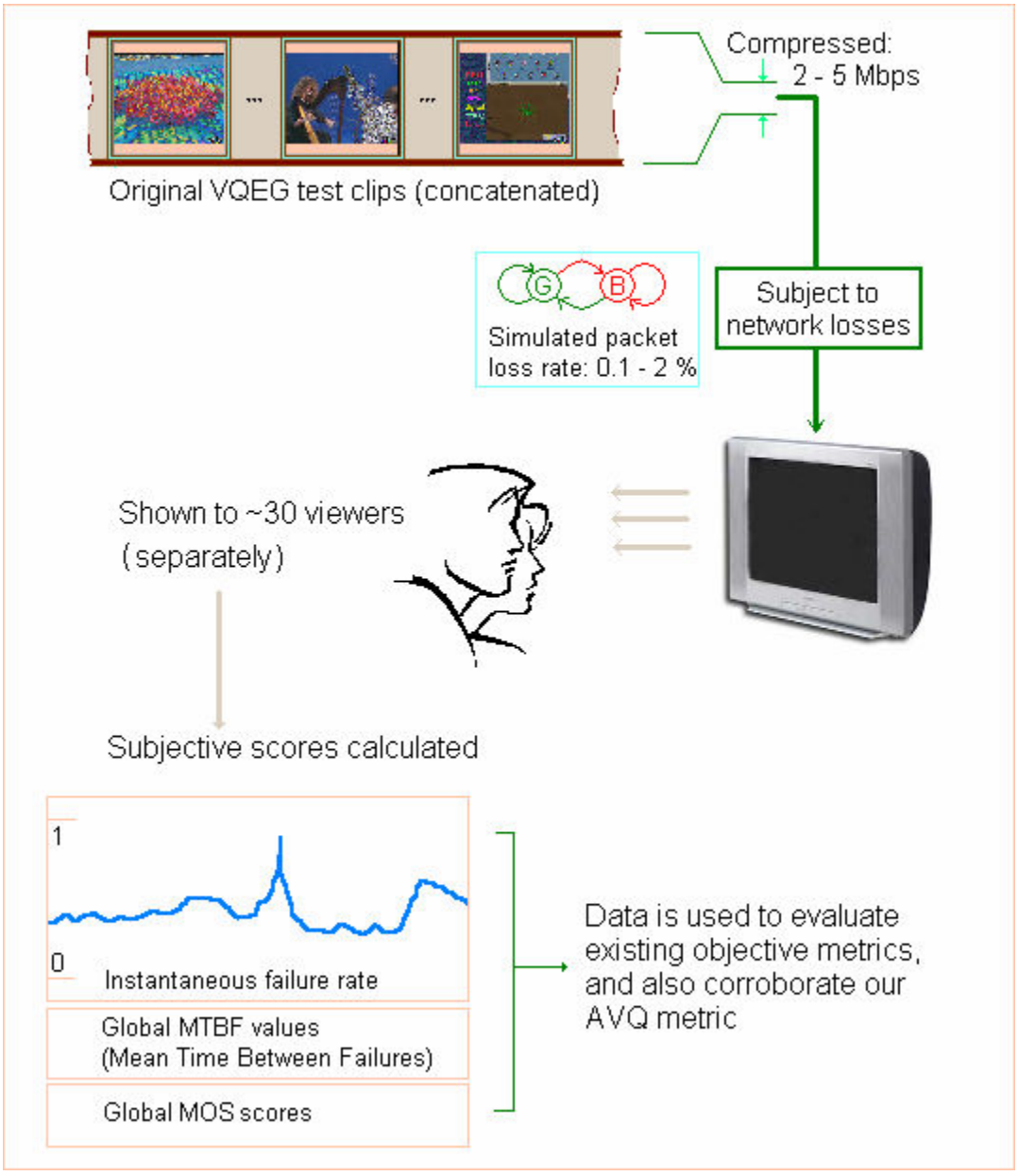


Fig. 14: The testing scheme used for generating subjective scores

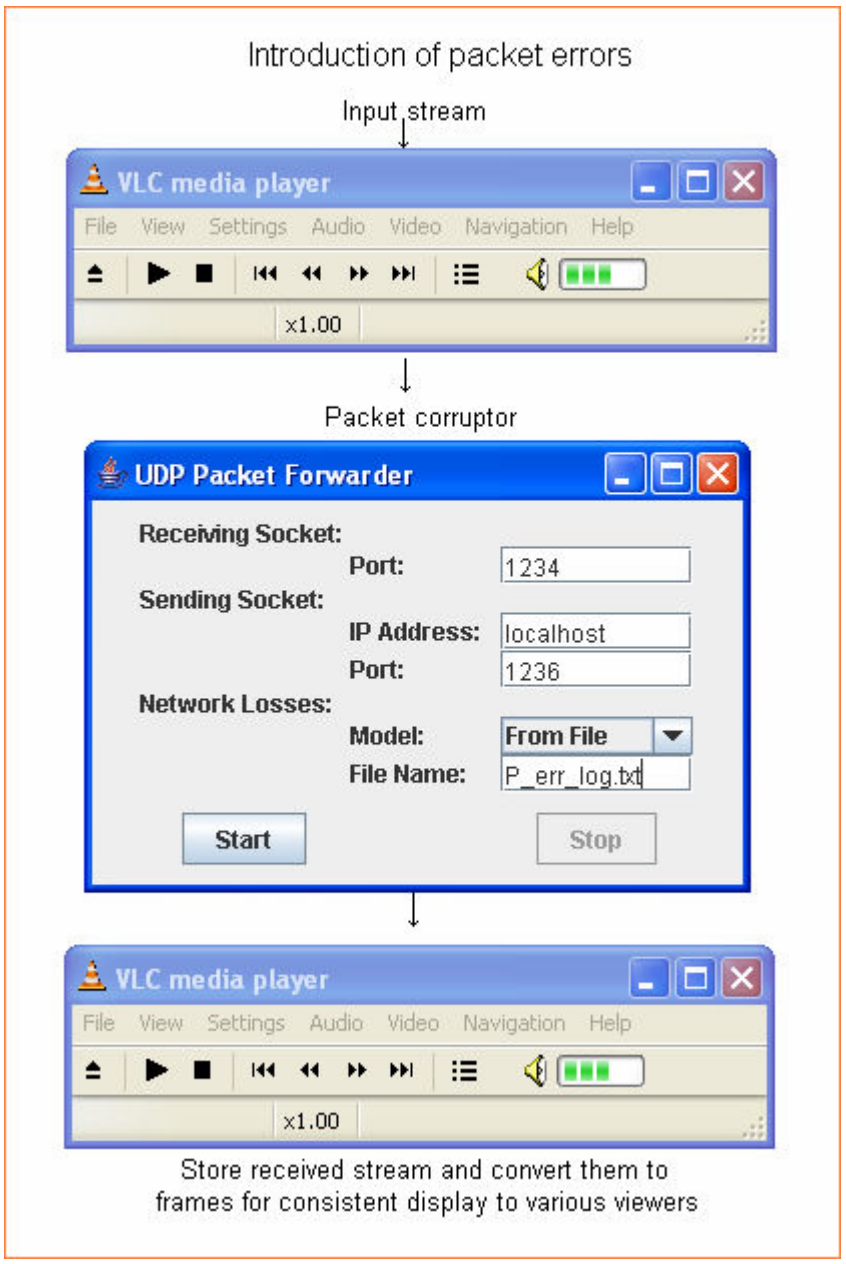


Fig. 15: The introduction of network errors into the test database

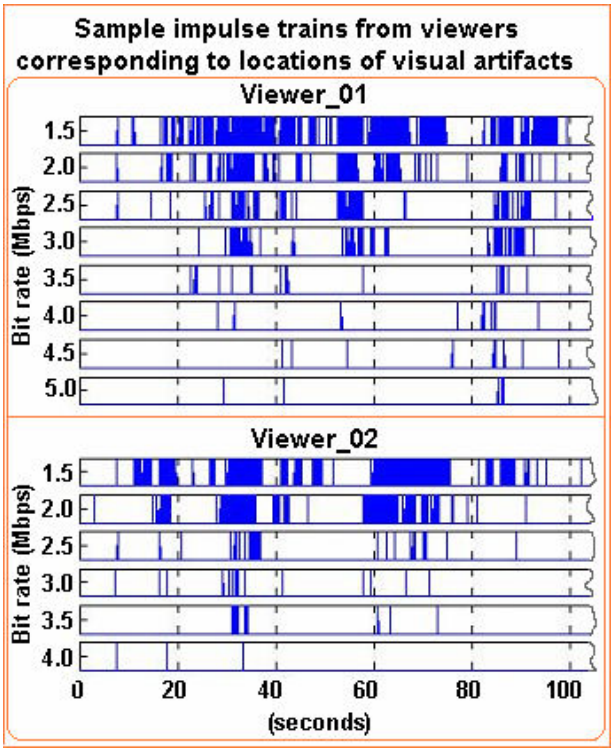


Fig. 16: Artifact triggers obtained from viewers (test database1)

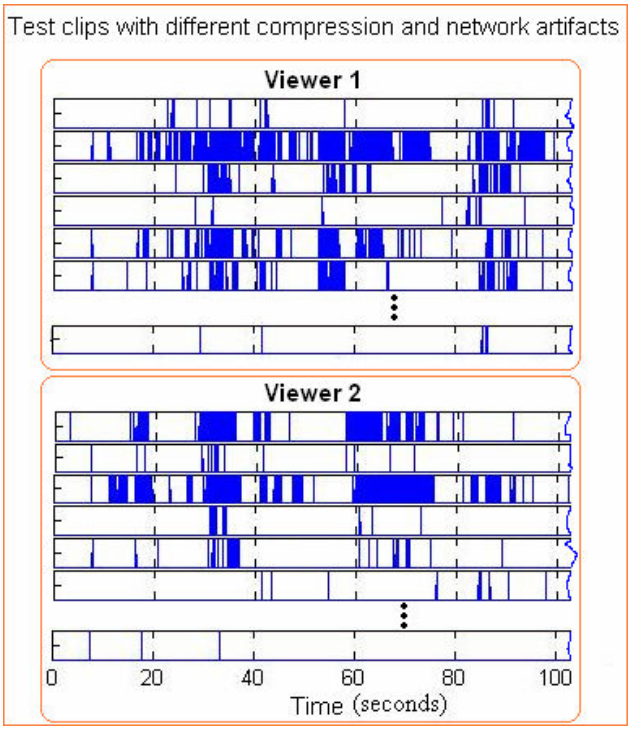


Fig. 17: Artifact triggers obtained from viewers (test database2)

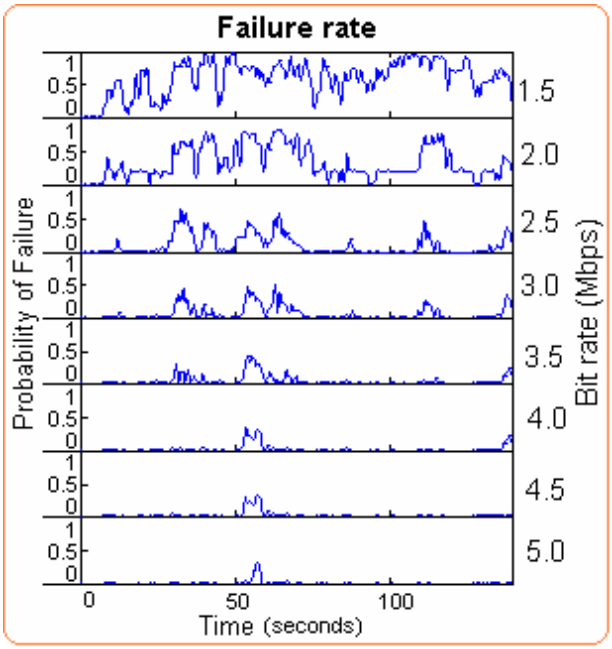


Fig. 18: Failure rate characteristics of the test sequence

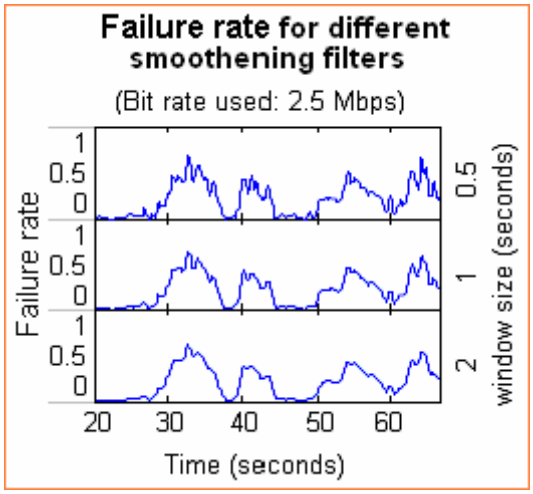


Fig. 19: Failure rate for different smoothing windows

CHAPTER VII

Relationship between *MOS* and *MTBF*

While both *MOS* and *MTBF* are global metrics, *MTBF* experiments generate a richer content of data for the same time spent in getting the viewers' scores. The instantaneous values of *failure rate* can be pooled over a particular subset of the video, and the related global *MTBF* values can be hence generated as necessary. This makes *MTBF* a very useful metric, since the correlations between subjective and objective scores can be measured over a controlled subset of the test database. For instance, our experiment includes seven test conditions on a given clip. For computing the correlation between *MOS* and an objective metric, we would have to pool the objective scores over each test condition and work with just seven numbers. With *MTBF*, we can pool the viewers' artifacts triggers and the objective scores over select sections and measure the correlation as a function of picture activity. Before this can be done, the effectiveness of *MTBF* needs to be verified. Hence, the relationship between *MOS* and *MTBF* needs to be studied.

In our earlier work [6], [1], we had studied the correlations between different objective metrics and *MTBF* by studying the scatter plots with *MTBF* on a logarithmic scale. A linear relationship in these plots was understood as a result of having the presence of the logarithmic function in many of the objective metrics, and the way the human visual system perceives artifacts. The relationship between *MOS* and log(*MTBF*) is also observed to be linear (Fig. 20, 21).

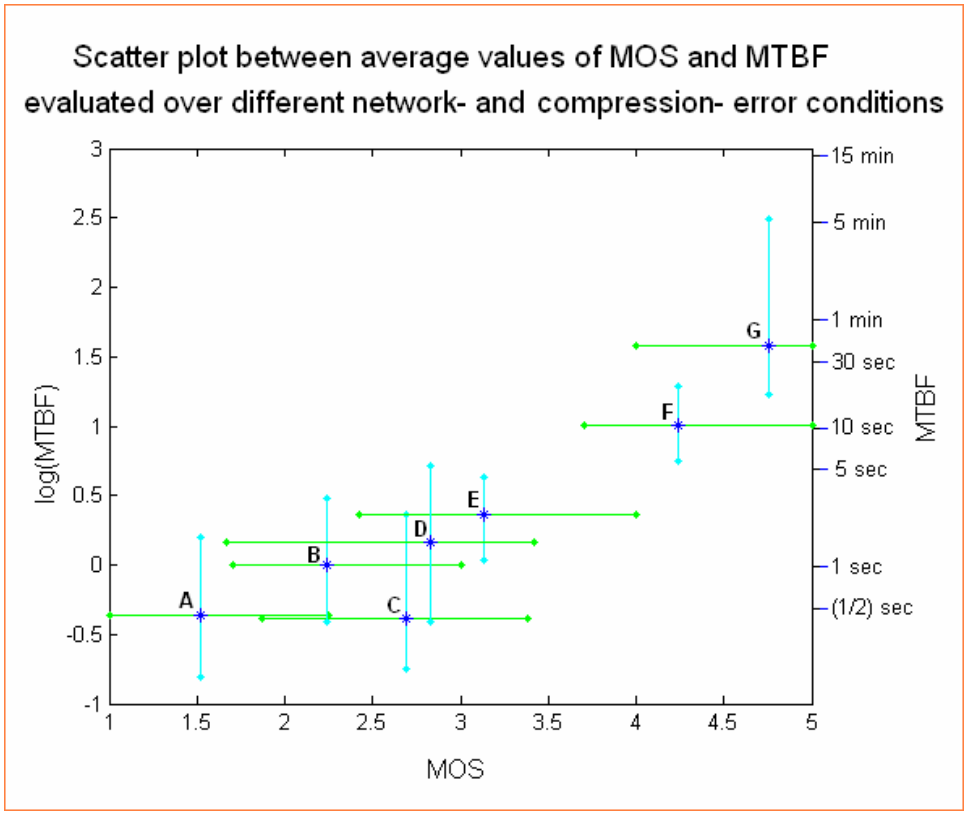


Fig. 20: Relationship between *MOS* and *MTBF*
(Letters A through B represent the different test conditions of compression and network artifacts as described in table 1)

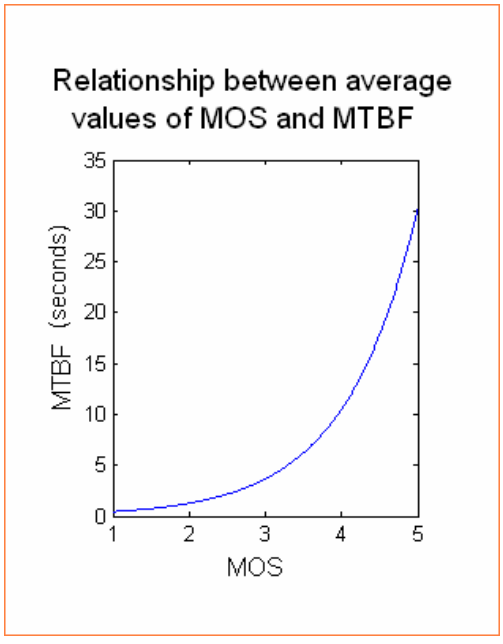


Fig. 21: Linear relationship between *MOS* and log(*MTBF*)

7.1 Correlation between *MOS* and *MTBF*

Given that there are 7 test conditions and 30 viewers, the *MOS*-*MTBF* correlation can be computed in different ways.

- Average correlation (data averaged first over all 30 viewers, and then computed over the 7 conditions) = 0.94
- Average correlation per viewer (computed over the seven conditions for every viewer, then averaged over all viewers) = 0.86
- Overall correlation (computed over the 7 x 30 values each of *MOS* and *MTBF*) = 0.71

It is inefficient to calculate the average correlation per condition due to the small spread of values around each test condition.

7.2 Variation in scores

The standard deviations in both the positive and negative directions for *MOS* and *MTBF* are shown in fig. 20. The reason for showing separate deviations on each side is to ensure that *MOS* values out of the applicable range (1 to 5) are not displayed.

The logarithms of the *MTBF* values were scaled accordingly, so that they spanned the same range as the *MOS* values. It was observed that *MOS* and *MTBF* had similar standard deviations (0.72 for *MOS* and 0.63 for *MTBF*).

These results suggest that *MTBF* has a direct and predictable relationship with *MOS*, and that they have similar variations across different viewers, when computed over any clip.

CHAPTER VIII

Comparison of Current Objective Metrics with *MTBF*

This section relates some of the objective quality metrics in the earlier section to *MTBF*. In the research completed so far, *PSNR*, *JND*, *STJM* and *BLK* are compared with the subjective scores of *MTBF* calculated.

8.1 Explanation of objective metrics that are compared with *MTBF*

PSNR and *JND* [3] are full-reference metrics. The *Spatial-Temporal "Join" Metric* [4] (*STJM*) is a *reduced-reference* metric. It works by comparing some features calculated over the input and output videos, such as the amount of spatial information and the presence of horizontal and vertical edges. It is considered to be an efficient and scalable metric. The *Wu-Yuen no–reference* metric mentioned in [5] is modified using a worst-5% spatial pooling technique and logarithmic conversion to get the *BLK* metric. It works by measuring the block artifacts introduced by the encoding scheme and also incorporates spatial and luminance masking.

Fig. 22 shows the variation of the average value of quality metrics with bit rate. The objective scores as a function of time are shown in Fig. 23, with bit rate as a parameter. There are some advantages and disadvantages to the *no-reference* metric. Since the original video is not available, it is expected to be not as accurate as some of the *full-reference* models.

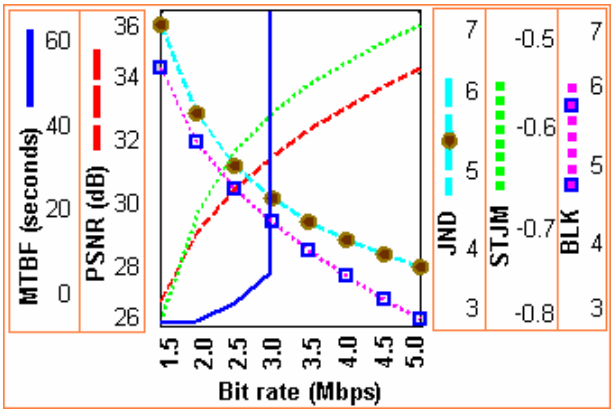


Fig. 22: Objective metrics vs. Bit rate

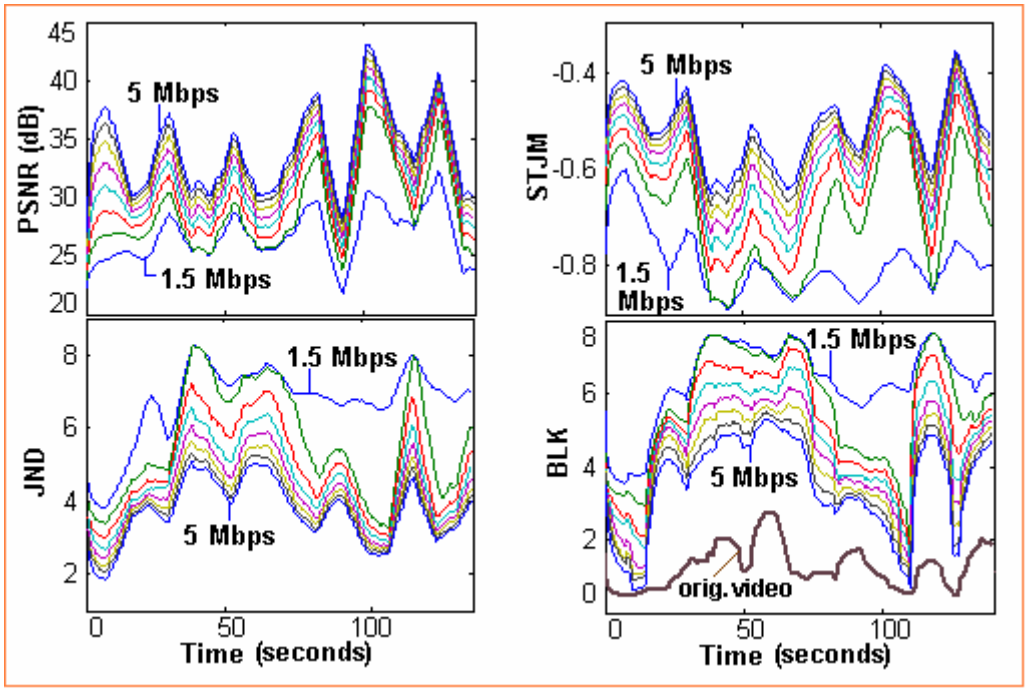


Fig. 23: Objective metrics vs. Time

Post-processing does not affect a *no-reference* metric in this way. In the case of *full-reference* and *reduced-reference* metrics, if the original itself is visually bad, then the metrics have no way of identifying this if the processed video is similar to the original video. For *no-reference* metrics, the residual signature (fig. 23) calculated on the original video can be thought of as the interpretation of inherent artifacts in the original by the human visual system.

8.2 Error pooling

Usually, only the luminance values are used to calculate the objective metric's scores, and so has been the case in the conducted experiments on *PSNR*, *JND*, *STJM* and *BLK*. The objective scores vs. time can be smoothed out with a time window to have a clearer view, as in fig. 23. This smoothing process averages out the metric values over a group of Pictures (GOP), where the scores vary considerably between I-, P- and B-frames of the MPEG-2 video.

Apart from the temporal pooling of error discussed above, the errors within a frame are also spatially pooled. In the case of *STJM* and *BLK* for example, the worst 5% area of the processed frame is used. This is done because the human eye tends to spot visual artifacts easily.

8.3 Relating objective scores to *MTBF*

The subjective *MTBF* measure can be correlated with an objective metric to observe its effectiveness, by pooling the data over different video clips and different bit rates (fig. 24).

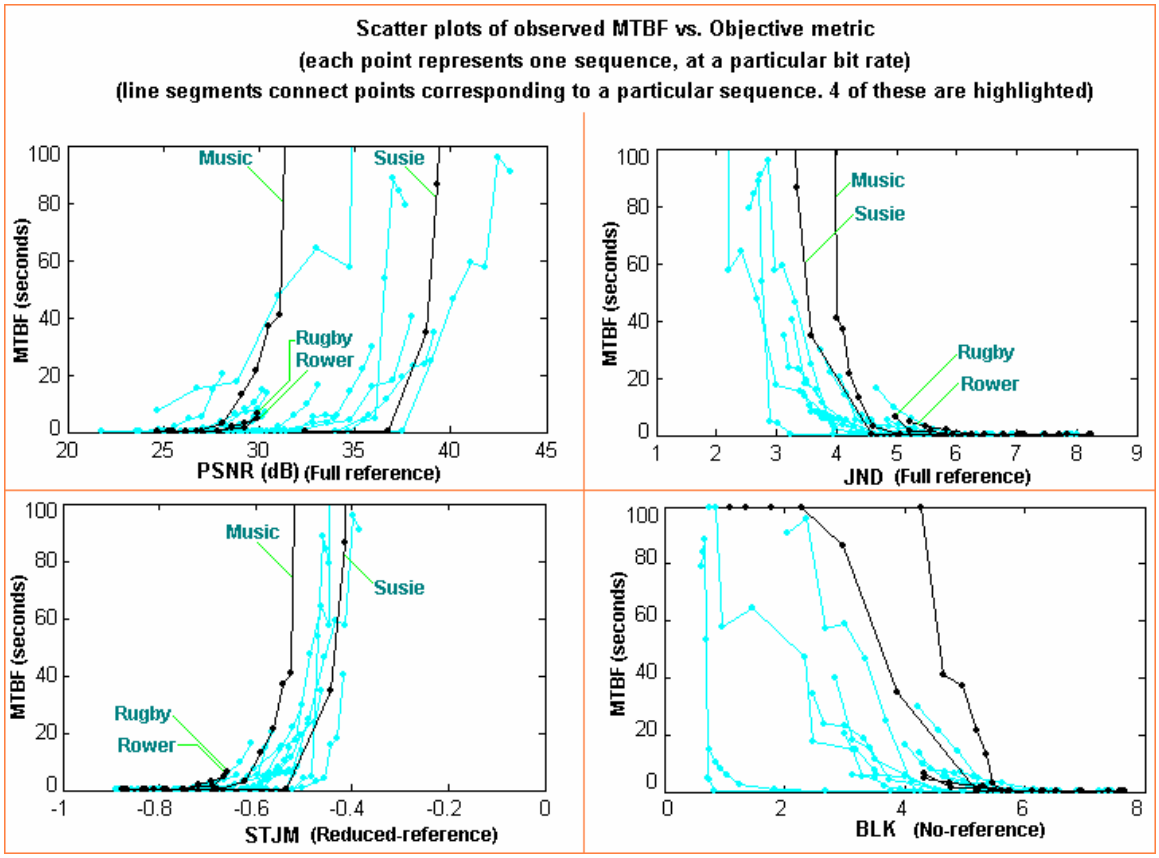


Fig. 24: Scatter plots of Objective metrics with *MTBF*

MTBF characteristics seem to exhibit an exponential type of behavior to a certain extent (fig 24). For example, as the *PSNR* of a sequence increases, the *MTBF* exponentially increases up to a point after which visual artifacts are practically not visible. The exponential behavior can be observed from the linear relationship in the scatter plot of log (*MTBF*) vs. the objective metric (fig. 25). The idea of using log (*MTBF*) also stemmed from the observation that many of the objective metrics use logarithmic expressions in their calculations to account for the nature in which the human eye perceives visual artifacts. The knee of the exponential curve depends on the type of sequence. For example, while the "Music" sequence starts to look good (meaning, high values of *MTBF*) at *PSNR* =28 dB, the "Susie" sequence starts to look good only at around *PSNR* = 38 dB.

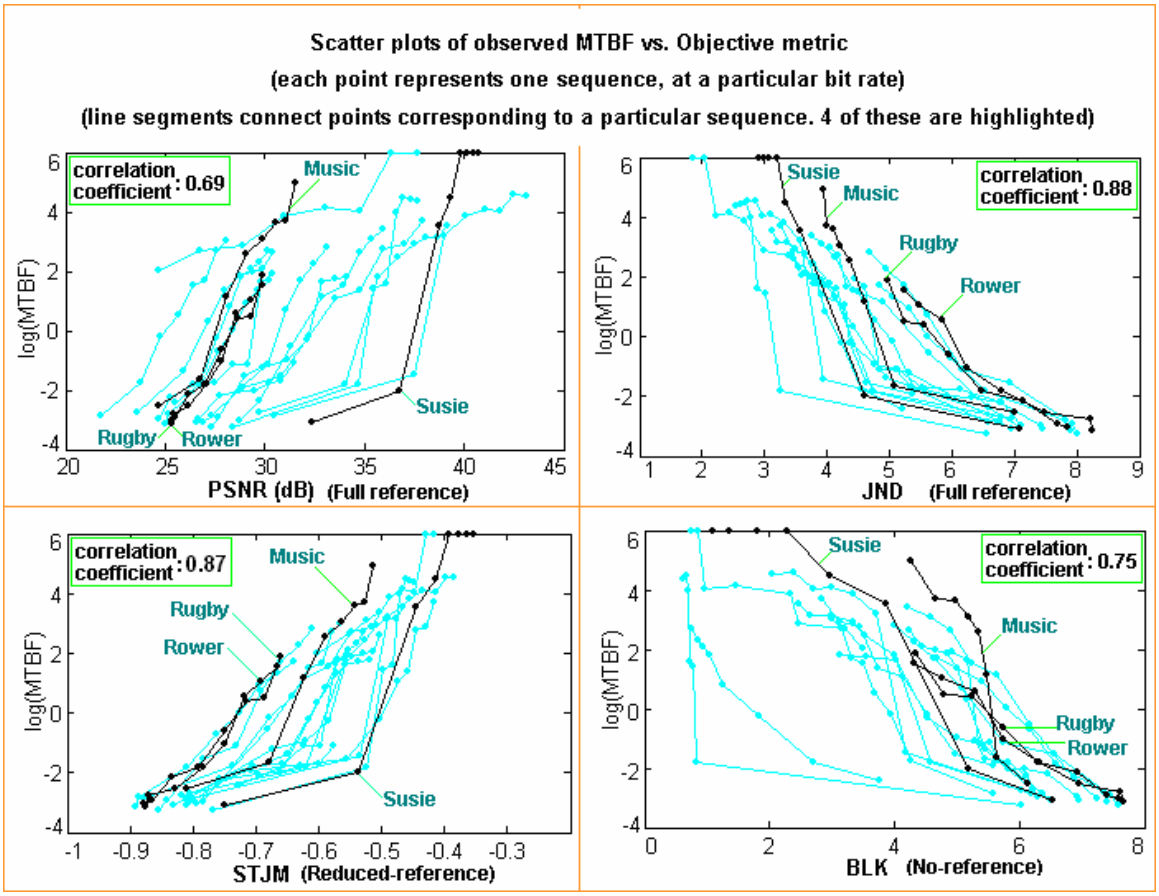


Fig. 25: Scatter plots of Objective metrics with log(*MTBF*)

This happens because of the objective metrics' inefficiency in estimating video quality, and also because of the variations between users that results in errors in the subjective metric itself. The "Music" sequence has a lot of spatial detail in it, which masks many of the artifacts introduced, and this is not captured by *PSNR,* which relies on a simple difference measure. Similarly, sequences like "Rower" and "Rugby" have a lot of inherent motion that seem to mask visual artifacts to an extent. No apparent relationship with spatial detail is found in the scatter plot of *JND* vs. *MTBF*, and this can be attributed to the incorporation of spatial masking in *JND*. However, visual masking in sequences with a lot of motion is observed, possibly due to the absence of temporal masking in *JND*. Observing such characteristics from the scatter plots helps in improving the design of objective metrics.

While *MTBF* is expected to increase monotonically as *PSNR* increases, this does not strictly happen for all sequences because the users are not conveyed the bit rate information of the video

beforehand. Having more exhaustive viewer response information should help reduce this problem. It should be noted however, that the objective metric behaves similarly for similar sequences. For example, the relationships between *MTBF* and *PSNR* for the "Rugby" and "Rower" sequences are very similar, because both the sequences have more or less the same amount of motion and noise information, and *PSNR* behaves similarly for both.

The reliability of an objective metric can be estimated by plotting the line (or in general, polynomial) of best fit in the '*log* (*MTBF*) vs. metric' graph, and noting the correlation coefficient. Higher correlations signify better predictions of *MTBF* from the objective metric. In this sense, the *JND* metric has a tighter fit in the plot as compared to *PSNR*. This is understandable, as *JND* incorporates spatial masking and is expected to be a better metric than *PSNR*. The reduced-reference *STJM* and the no-reference *BLK* metrics incorporate some spatial masking, and perform better than *PSNR*. It should be noted that when the no-reference *BLK* metric is computed over the original signal, its scores are not constant with time as desired (fig. 23).

8.4 Estimation of *MTBF* from objective metrics

The *MTBF* for a set of test stimuli and a pool of subjective viewers can be averaged in different ways for a better understanding. The test clips are of duration 8-10 seconds each, and each clip is available at different bit rates. From the extensive test data, *MTBF* can be averaged over a set of test clips to obtain the spread of *MTBF* with different viewers as a function of bit rate. Alternatively, *MTBF* can be averaged over all the viewers and test clips and displayed as a function of bit rate (fig.22) or it can be averaged over all the viewers for every test clip and bit rate setting (fig. 25). The overall average of *MTBF* calculated over all the parameters involved (different viewers, test clips and bit rate) can also be calculated from the test data.

The relationship between an objective score and the corresponding value of *MTBF* (averaged over all viewers for different test clips at different bit rates) can be interpolated to find the expected *MTBF* of any given video (fig. 26). A reduced-reference metric like *STJM* [4] or *BLK* [6] is recommended in our work for this purpose. By observing the scatter plot of the '*log* (*MTBF*) vs. objective-metric' graph, the exponential of best fit is determined to find the relationship between the objective-metric and *MTBF*. The

relationship between the metric and failure rate is computed using the inverse relationship between *MTBF* and failure rate.

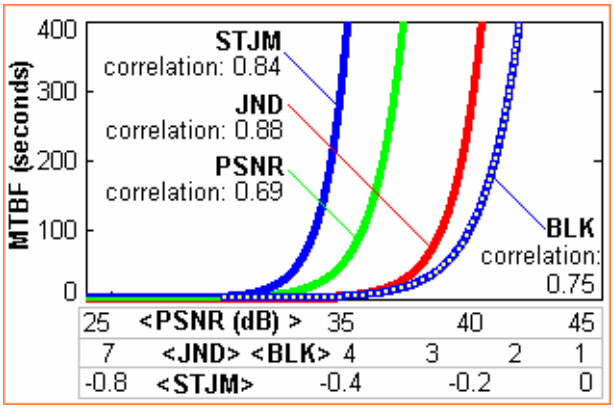


Fig. 26: Estimation of *MTBF* from objective metrics

With this relationship, it is easy to calculate the *MTBF* of any arbitrary video considering that we effectively have a lookup table between average metric values and *MTBF*. For video sequences that have a wide range of objective metric values, the intermediate failure rate characteristic can be used to get an estimate of the video quality: The objective-metric vs. time of the corrupted video is calculated, with this, the failure-rate is estimated using a table lookup, and then *MTBF* is calculated for the overall video sequence as the reciprocal of the average failure rate. It is to be noted though, that the equation for calculating *MTBF* (equation 1) is theoretically valid only for a constant value of failure rate. Nevertheless, it can still be used in the case of near constant values of failure rate. If the failure rate does change a lot with time, then local values of *MTBF* can be computed and the individual values of *MTBF* can be pooled over time as desired.

The *STJM* metric considered in this work uses a block size of 8x8 pixels for every frame. With a frame size of 720x480 and a frame rate of 30 fps, the information overhead for conveying this *reduced-reference* metric works out to a huge 2.6 Mbps. The effectiveness of the metric at lower overheads is a topic of interest. As anticipated in [4], we have observed that by calculating the features over selective time-varying regions of the video, the overhead can be reduced by about a factor of 50 without a significant change in the performance of the metric.

CHAPTER IX

The Automatic Video Quality Metric (AVQ)

AVQ is an acronym for a no-reference metric that is being developed built on the new algorithms described in this section based on the characteristics of the various kinds of artifacts observed in video transmission systems, and the knowledge of the Human Visual System. A comprehensive understanding of existing objective metrics and their strengths and weaknesses of these metrics has also helped in making AVQ efficient and subjectively relevant, with the correlation values approaching the correlations of metrics that use full or partial reference.

The metric works on the pixel values and / or the bit-stream parameters. A schematic flowchart of the meter is described in the GT_AVQ flowchart. AVQ has been implemented as a real-time evaluation tool that displays the video quality in terms of Mean Time Between Failures. The flowchart shows key components of the AVQ meter in the real-time implementation, as well the other modules in progress. The degree of compression artifacts is estimated as a function of different available aspects of the bit-stream and pixel values (GT_AVQ flowchart), such as the quantization step size and the picture activity in the scene. The perceptibility of network artifacts is obtained from the deviation from normal behavior as observed during the error concealment process. Normal behavior may be defined as a function of the current frame and a predicted version of it. Any of the functions mentioned above could be either applied to the entire frame, or on a selective block-by-block basis. The AVQ metric is also diagnostic in nature, in the sense that it indicates the relative annoyance of compression and network artifacts. Current metrics in literature evaluate different aspects of the visual artifact such as blocking, blurring, ringing, Gaussian noise, etc., and compute the overall metric as a linear combination of the individual components. A linear combination is usually not sufficient to describe the relationship between the overall metric and the individual type of artifacts, and non-linear methods are used to make the AVQ increasingly efficient.

Block-Transform based compression schemes like MPEG-2 and H.264 introduce a variety of artifacts in the video. Blockiness and Blurriness are two of the most common artifacts. Block artifacts occur when the DCT-block edges are visible in the picture frames, and Blurriness is caused at times when the edges in the image are subject to excessive compression. Apart from these compression related artifacts,

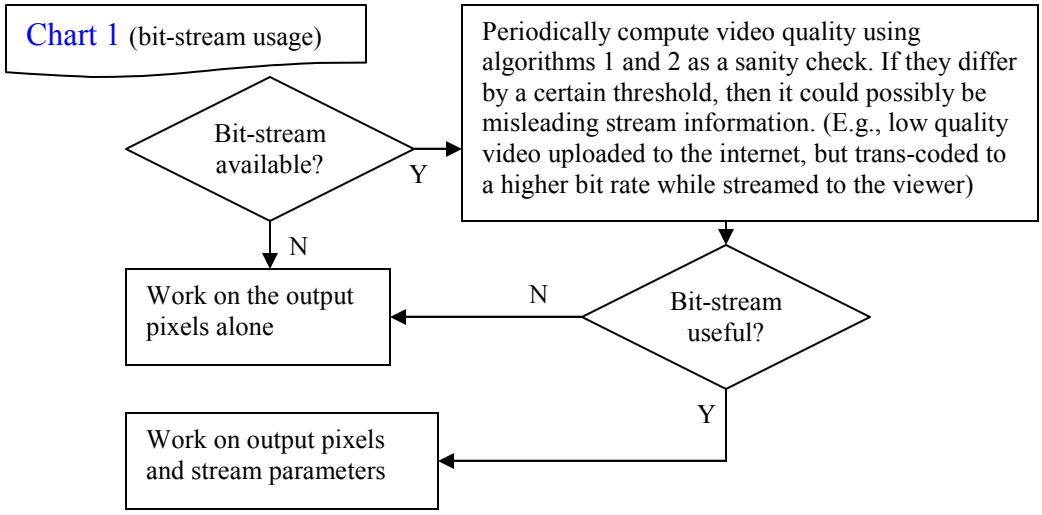
packet losses in the video stream cause network artifacts as well, which manifest themselves as unnatural streaks in the frames or as stuck / reordered frames. This work relates to a class of no-reference objective video quality metrics that strive to evaluate these different artifacts with a unified approach. The modules in the AVQ meter, termed the Spatial-Temporal Coherence metrics, approach the problem of evaluating video artifacts by observing the behavior of specific attributes of the video within a frame and across frames. The current sample implementations of blockiness, blurriness and network-error detectors are described in the following sections.

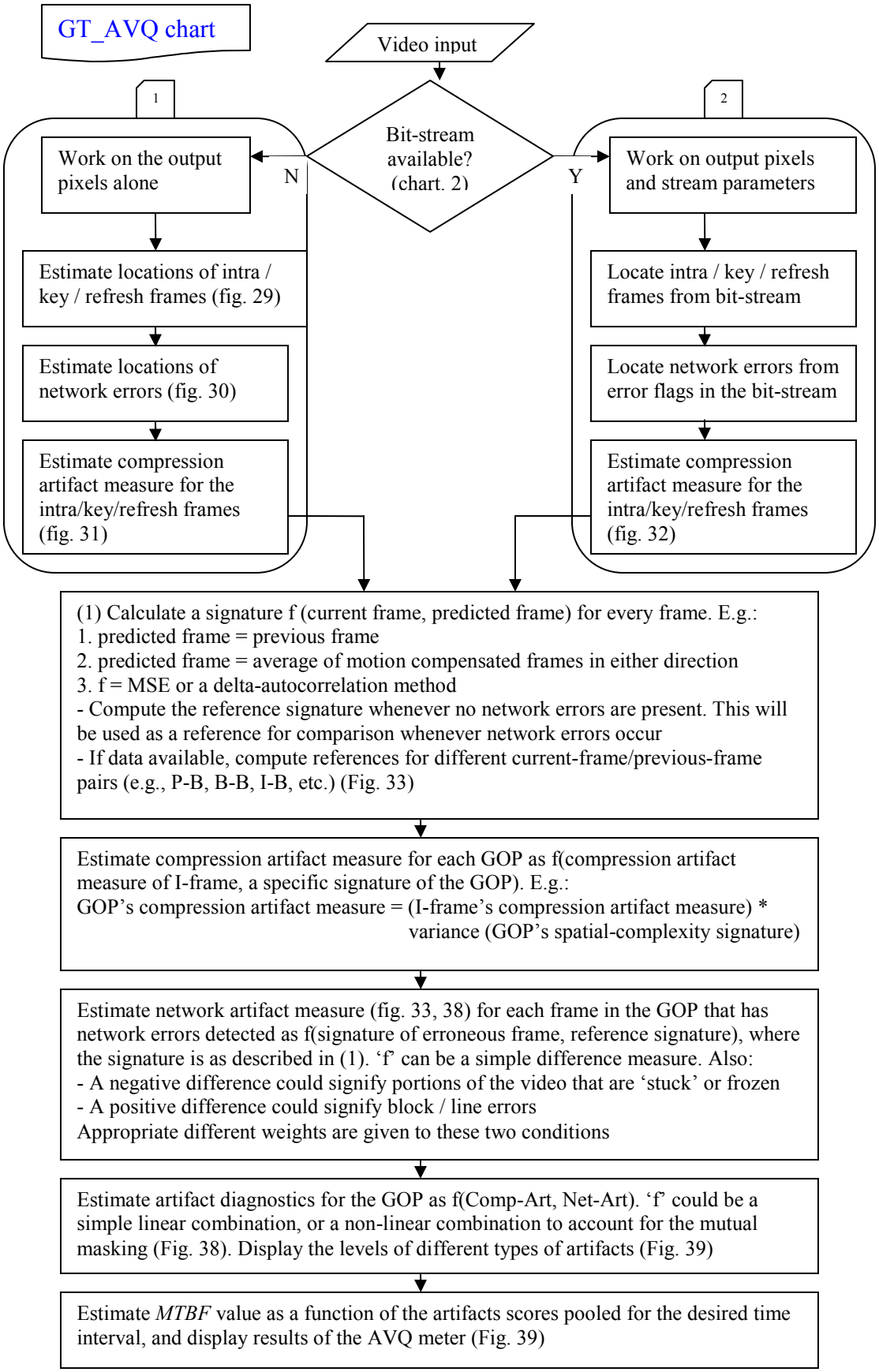
9.1 Components of the AVQ:

9.1.1 Quantization Step Size:

The quantization step size used in the initial compression has been found to be a good indication by itself of the subjective quality or PSNR [35], [37].

While Snell & Wilcox's Picture Appraisal Measure (PAR) [34] focuses on directly using this step size parameter from the bit-stream, the Intel patent concentrates on estimating this and other encoding parameters from the output pixels alone [38]. In the AVQ meter, the quantization step sizes for different macro-blocks in a frame are measured and pooled to get a number per frame. It is observed that the logarithm of the average quantization step size correlates well with most subjective metrics in general.




GT_AVQ chart
Video input
1
2
Bit-stream available? (chart. 2)
N
Y
Work on the output pixels alone
Work on output pixels and stream parameters
Estimate locations of intra / key / refresh frames (fig. 29)
Locate intra / key / refresh frames from bit-stream
Estimate locations of network errors (fig. 30)
Locate network errors from error flags in the bit-stream
Estimate compression artifact measure for the intra/key/refresh frames (fig. 31)
Estimate compression artifact measure for the intra/key/refresh frames (fig. 32)
(1) Calculate a signature f (current frame, predicted frame) for every frame. E.g.:
1. predicted frame = previous frame
2. predicted frame = average of motion compensated frames in either direction
3. f = MSE or a delta-autocorrelation method
- Compute the reference signature whenever no network errors are present. This will be used as a reference for comparison whenever network errors occur
- If data available, compute references for different current-frame/previous-frame pairs (e.g., P-B, B-B, I-B, etc.) (Fig. 33)
Estimate compression artifact measure for each GOP as f(compression artifact measure of I-frame, a specific signature of the GOP). E.g.:
GOP's compression artifact measure = (I-frame's compression artifact measure) * variance (GOP's spatial-complexity signature)
Estimate network artifact measure (fig. 33, 38) for each frame in the GOP that has network errors detected as f(signature of erroneous frame, reference signature), where the signature is as described in (1). 'f' can be a simple difference measure. Also:
- A negative difference could signify portions of the video that are 'stuck' or frozen
- A positive difference could signify block / line errors
Appropriate different weights are given to these two conditions
Estimate artifact diagnostics for the GOP as f(Comp-Art, Net-Art). 'f' could be a simple linear combination, or a non-linear combination to account for the mutual masking (Fig. 38). Display the levels of different types of artifacts (Fig. 39)
Estimate *MTBF* value as a function of the artifacts scores pooled for the desired time interval, and display results of the AVQ meter (Fig. 39)

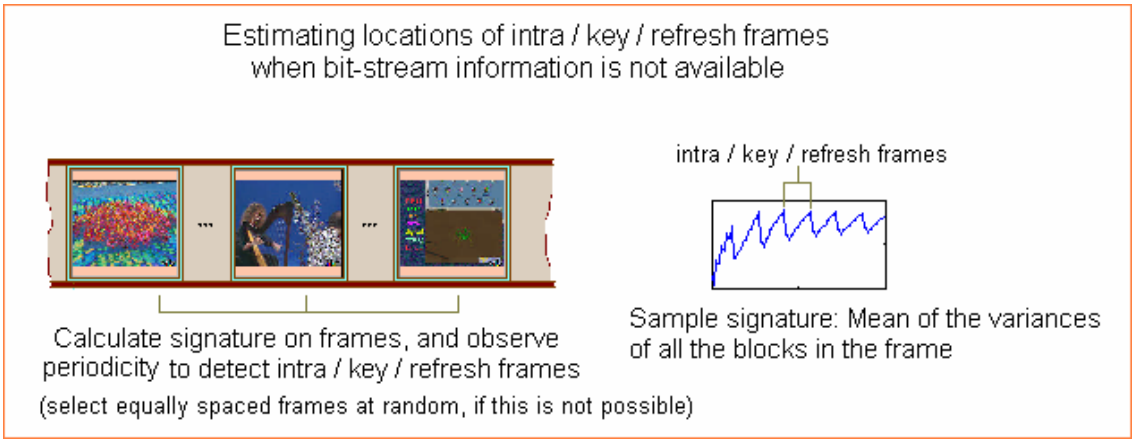


Fig. 29: Sample pixel-based algorithm to locate intra frames

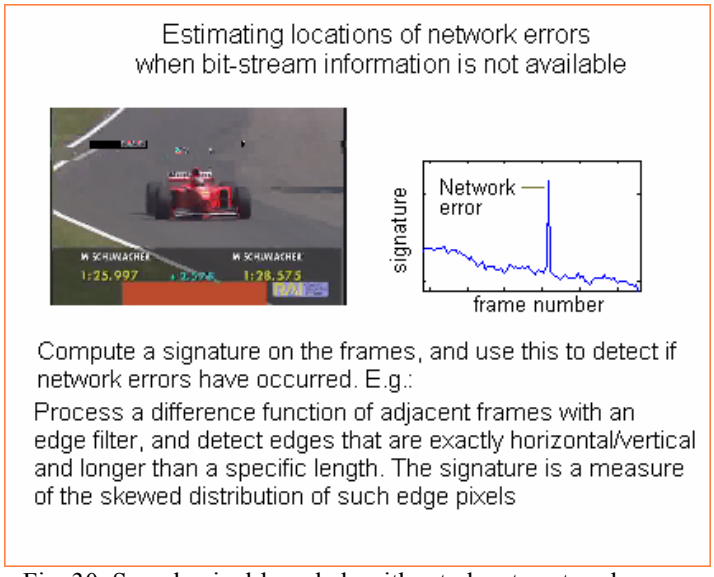


Fig. 30: Sample pixel-based algorithm to locate network errors

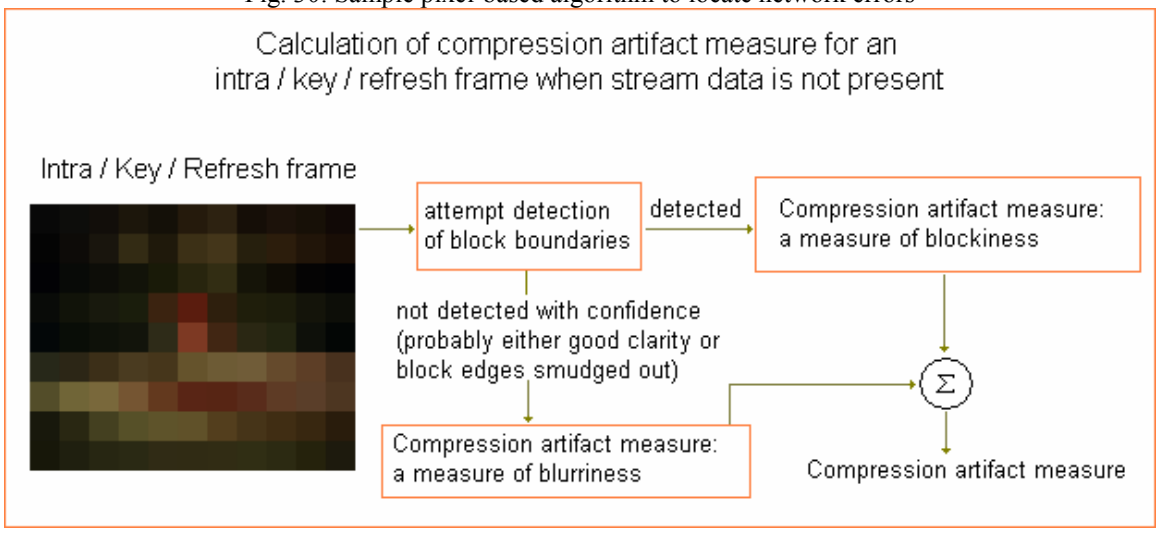


Fig. 31: Sample pixel-based algorithm to evaluate compression artifacts

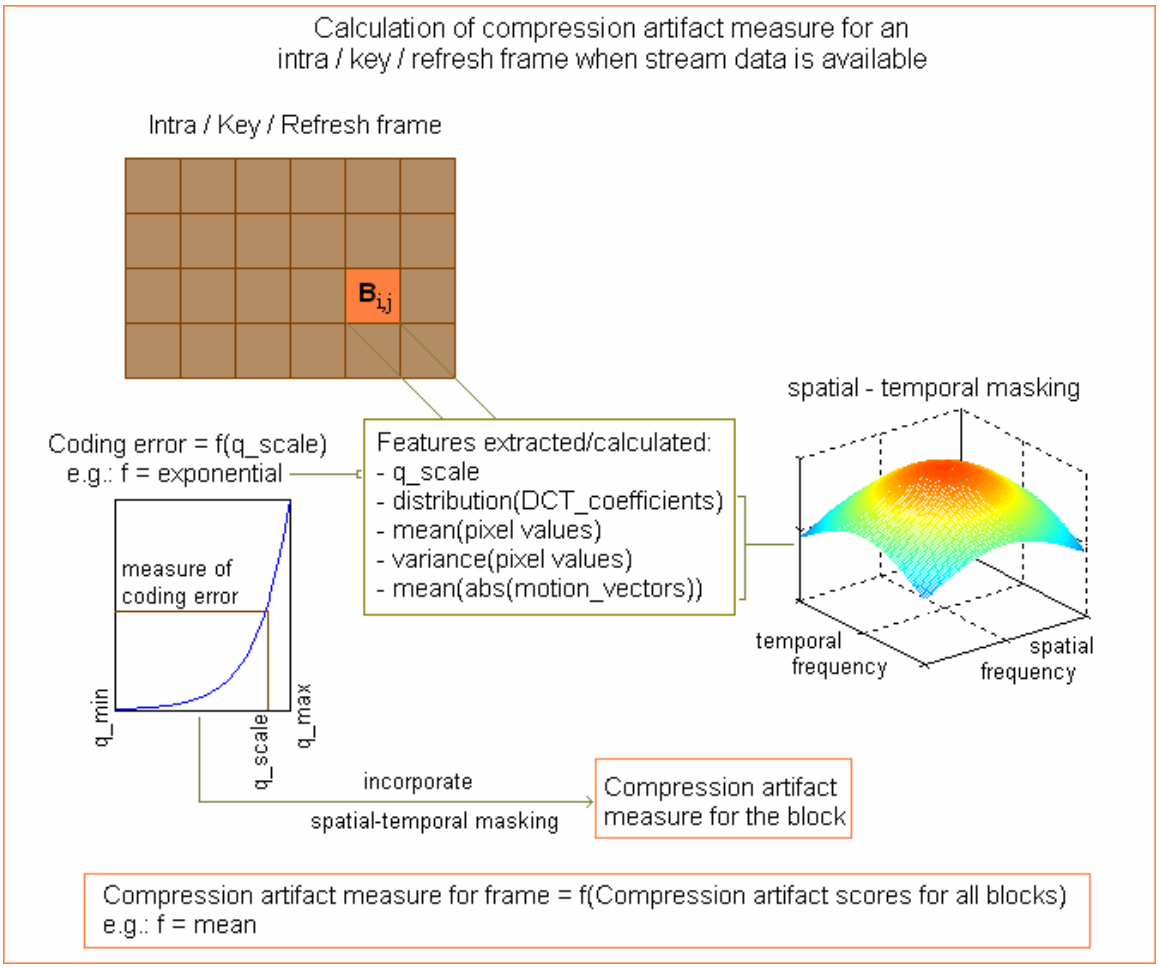


Fig. 32: Sample algorithm to evaluate compression artifacts when bit-stream information is available

9.1.2 Number of DCT coefficients:

The number of DCT coefficients has some reference in literature. Snell & Wilcox's PAR [35] measure estimates PSNR as a function of quantization step size and the activity in the picture. While they specify an exhaustive theoretical method towards finding the activity, for practical purposes, they just calculate picture quality as a function of the number of DCT coefficients used in encoding and the quantization step size.

The number of DCT coefficients used in encoding is an ingredient to some of the components of the AVQ metric.

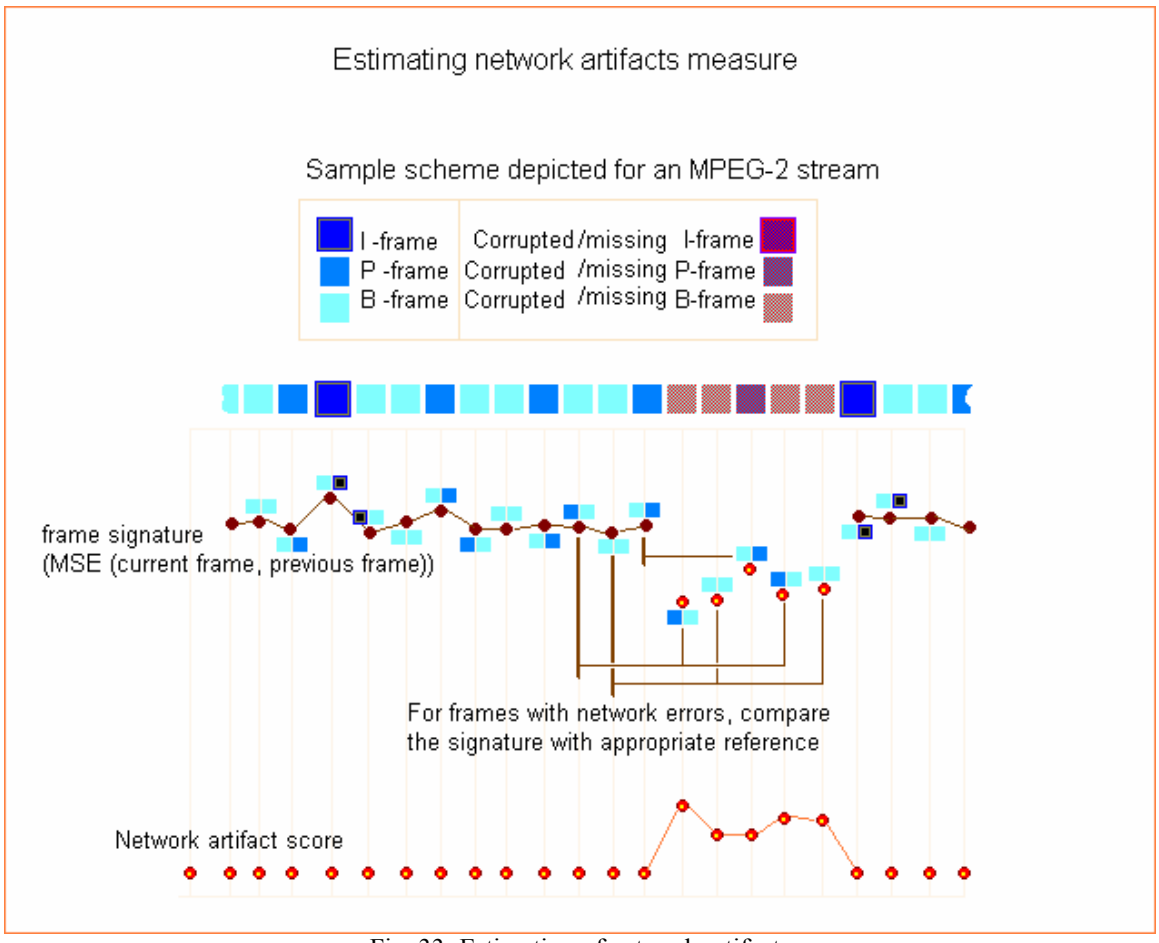


Fig. 33: Estimation of network artifacts

9.1.3 Bit Rate:

The bit rate used in encoding is by itself a good indicator of the resulting quality. The observation of the relationship between the subjective video quality and bit rate for sequences in the test database is used to improve the AVQ algorithm whenever it is accessible.

9.1.4 New Blockiness Metric:

There are a considerable number of blockiness metrics in literature, and exhaustive surveys of those metrics as well [1][24], [2][47]. Most metrics compare the inter-block and intra-block differences to get an estimate of the video quality [3][35], [4][5]. Some metrics compare the differences in correlation between and across block boundaries [5][22]. Some metrics measure blockiness from the histogram of edge angles in the video frames [6][50]. These blockiness metrics in general focus on a video frame, and do not

incorporate temporal masking. The metrics described above are no-reference in nature, meaning that the quality score can be evaluated with just the received video. There are some reduced-reference metrics as well, that evaluate blockiness. For instance, [7][65] evaluates video quality by measuring the degradation of certain features extracted over the frames. One of the features relates to the addition of new edges in the compressed video that are close to horizontal or vertical alignments.

Some of the drawbacks of current metrics are that they can function unexpectedly when the image contains intended edges. This problem is avoided at times by using different thresholds for omitting natural edges [8][10]. The threshold calculation is difficult, however, resulting in a few false decisions. When the metrics are calculated over an original signal with no block artifacts, one would expect a metric signature that indicates an error free signal. In general, this is not the case, and there is in fact a varying signature with time. This problem is particularly encountered when there are scene changes in the video.

Proposed algorithm:

The proposed metric for detecting block artifacts (BLK_ART), works by evaluating the spatial and temporal distribution of edges, especially horizontal and vertical edges in the video. BLK_ART is shown to have a good correlation with subjective scores, and when combined with ideas from existing metrics, the metric outperforms existing blockiness metrics. Aside from the good correlation, this metric is also observed to be computationally efficient. It has been implemented in real-time in our Automatic Video Quality meter (AVQ) as well. The algorithm and different versions of it are described below.

The blockiness algorithm does not need access to the exact location of the individual block boundaries. This makes it possible to function as a pixel based algorithm without any need to access the bit-stream. The algorithm can be evaluated on per frame basis, or evaluated over different regions of a frame and pooled as required. For a given video frame, horizontal and vertical edge filters are applied to get the horizontal and vertical edges in the image.

The vertical edge image can be used to compute the blockiness caused by the vertical edges of the DCT blocks, and the horizontal edge image can be used in a similar fashion for the horizontal artifacts. The vertical edge image is further processed to include only those edge pixels that belong to an edge that is exactly vertical and longer than a stipulated length. For instance, a value of four for this length parameter is

observed to locate the block artifact edges with reasonable accuracy. This processed vertical image is then sliced into different fields. This is done by down-sampling the image in the horizontal direction by a specific number. Down-sampling by eight is observed to reflect the periodicity of the artifacts accurately for typical video frames. This operation results in eight different fields of the edge image. The distribution of the edges in these different fields is observed to get an estimate of the blockiness.

Images that do not have any blocking artifacts would typically have this edge distribution uniform across the different fields. A sharp deviation from uniform behavior would indicate blockiness. Fig. 35 shows typical screenshots of MPEG2 video frames at different qualities. Fig. 34 shows their corresponding distribution of the abovementioned edge pixels across different down-sampled fields of the image. These numbers are arranged in ascending order for an easier understanding. The high quality image has roughly the same number of edge pixels for each field. Hence, its blockiness estimate is minimal. On the other hand, the low quality image has a skewed distribution. The down-sampled version of the image containing the block-DCT boundaries has a disproportionate amount of long vertical edges. This deviation from expected behavior is used to calculate the blockiness estimate.

Algorithm implementation issues:

Edge filtering: The edge detection algorithm can be performed either on the video frame itself, or on inter-frame differences. Working on inter-frame differences, or in general, a function of different frames in a neighborhood is observed to produce good results. When performed on just the video frame, the algorithm has to use appropriate thresholds to make sure that intended edges in the image such as frame borders are not incorrectly detected as artifacts. When the algorithm uses inter-frame differences, this problem is avoided. If the intended edge is stationary, then it does not figure in the inter-frame difference. If the intended edge is moving, then it figures in different down-sampled fields in the frame difference image, and does not interfere with the blockiness calculation. The inter-frame difference makes it easier to observe the blockiness in video.

Masking: The step involved in deciding whether a pixel belongs to a vertical / horizontal edge or not involves spatial-temporal masking. For instance, the masking function as described in [4][5] could be used, and has been found to function well in our work. The X-axis denotes the average luminance value around the pixel of interest. A higher weight in the graph indicates that the effect of the pixel difference is more pronounced. The spatial masking function also takes into account the effect of local standard deviation (equation (4)).

$$W = \begin{Bmatrix} \lambda \ln\left(1 + \dfrac{\sqrt{mean}}{1 + deviation}\right), mean < \varsigma \\ \ln\left(1 + \dfrac{\sqrt{255 - mean}}{1 + deviation}\right), mean \geq \varsigma \end{Bmatrix} \tag{4}$$

where,

$$\lambda = \frac{\ln\left(1 + \sqrt{255 - \varsigma}\right)}{\ln\left(1 + \sqrt{\varsigma}\right)} \tag{5}$$

The edge detection process involves applying the masking function to the pixel differences and then comparing them to a certain threshold. In this work, the same goal is achieved by choosing the threshold as a function of average luminance and standard deviation values. This turns out to be computationally more efficient, given that this new masking function can be approximated as a combination of line segments. In addition, the standard deviation is approximated to the average linear deviation from the mean luminance value. With this approximation, the threshold factor masking function can be represented by the curve in fig 36. This blockiness detection algorithm with the appropriate masking function has been observed to work well in real time in the implementation of the AVQ meter.

Apart from the spatial masking, the use of temporal masking is incorporated as well. Some examples of temporal masking schemes are described in [72], [73]. For instance, when the edge detection algorithm is applied to inter-frame differences, the threshold for edge detection is scaled linearly as a function of the mean pixel value differences between consecutive frames. Other algorithms that could be an

extension of this algorithm include temporal masking schemes that directly or indirectly detect the motion present in the received video.

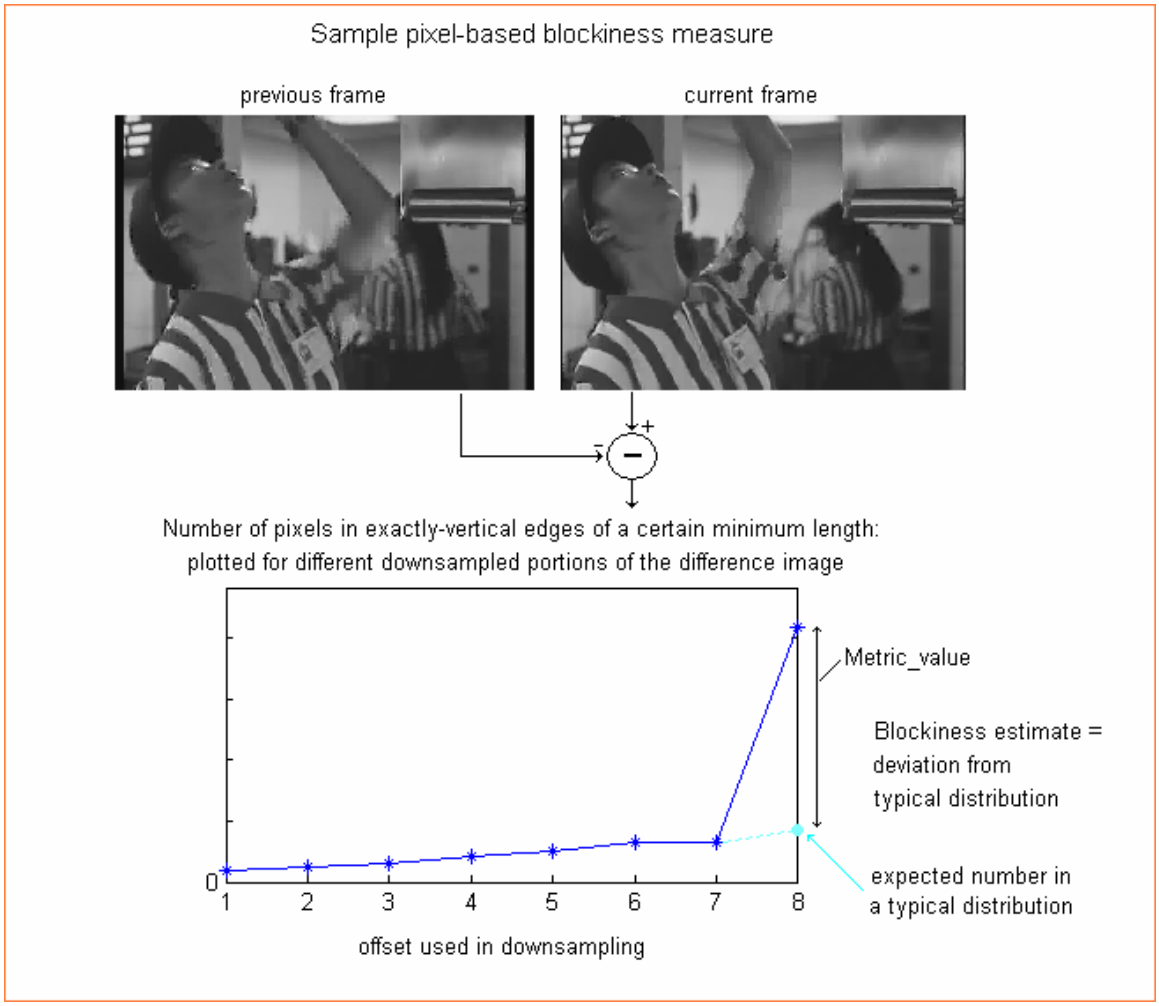


Fig. 34: Sample pixel-based blockiness measure based on the spatial distribution of certain edge pixels in the video frame

The blockiness algorithm could be enhanced in several ways. For instance, different sub-sampling sizes could be used to observe irregular patterns in the video. Block sizes down to a 4x4 pixel area could be used for detection of artifacts in H.264 video. The algorithm could be applied to a block-by-block basis and on a neighborhood of frames with a concept of selective spatial-temporal pooling in mind. The nature of the variation in blockiness scores or any other function of the pixel values with time itself could be used to enhance the metric. For instance, the blockiness measure for a group of frames could be a linear function of

the blockiness of its intra frame, and the variation of the local pixel variance with time over the group of frames. The algorithm could also be made into a reduced-reference implementation by computing the metric scores at the sender, and comparing them with the received signature. The effectiveness of the blockiness metric can be increased by using information from other metrics, such as, quantization step size, number of DCT coefficients or bit-rate.

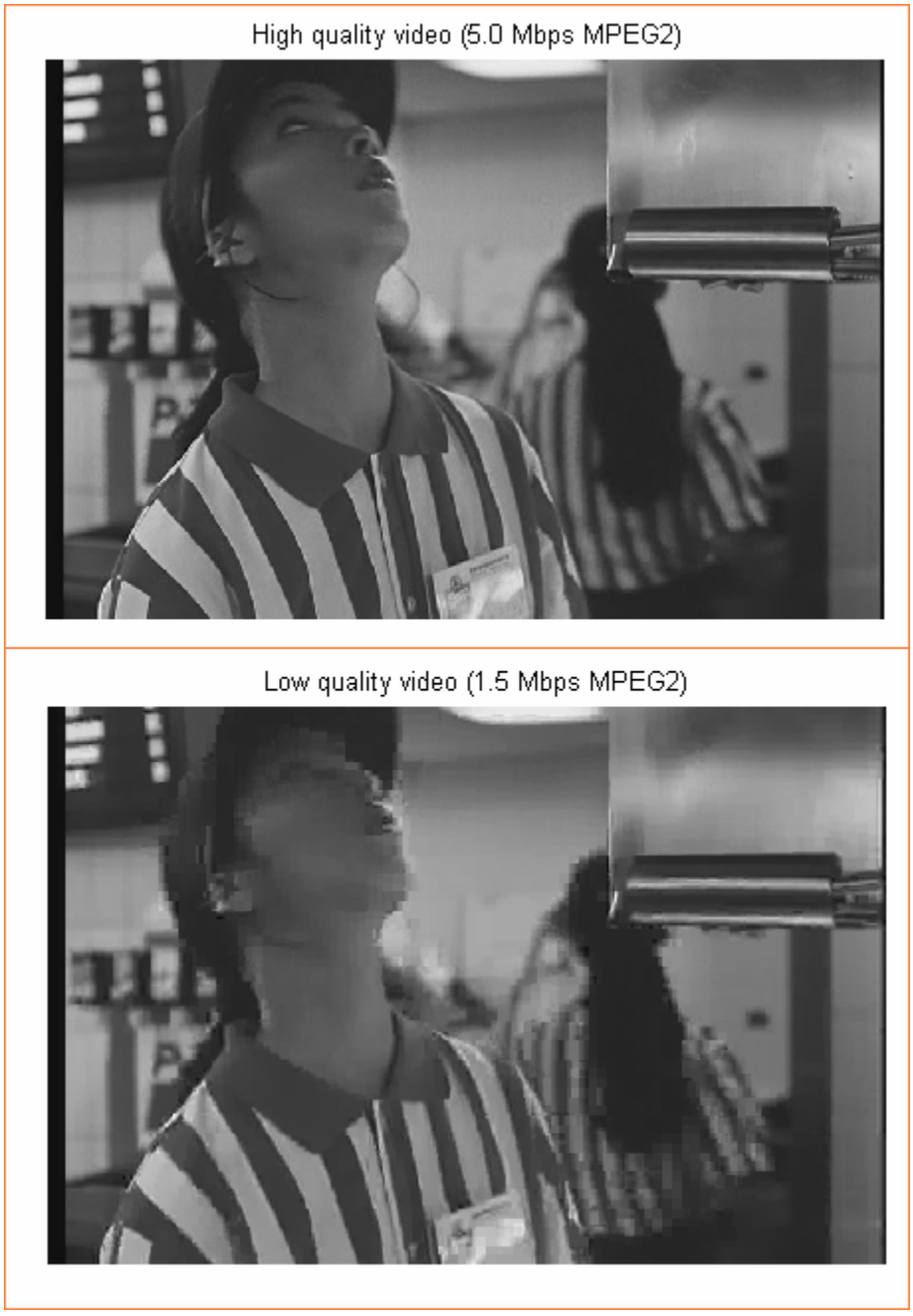


Fig. 35: Screen shots of two different qualities of MPEG2 video

(Note the abundant presence of natural edges that could confuse many of the existing blockiness metrics)

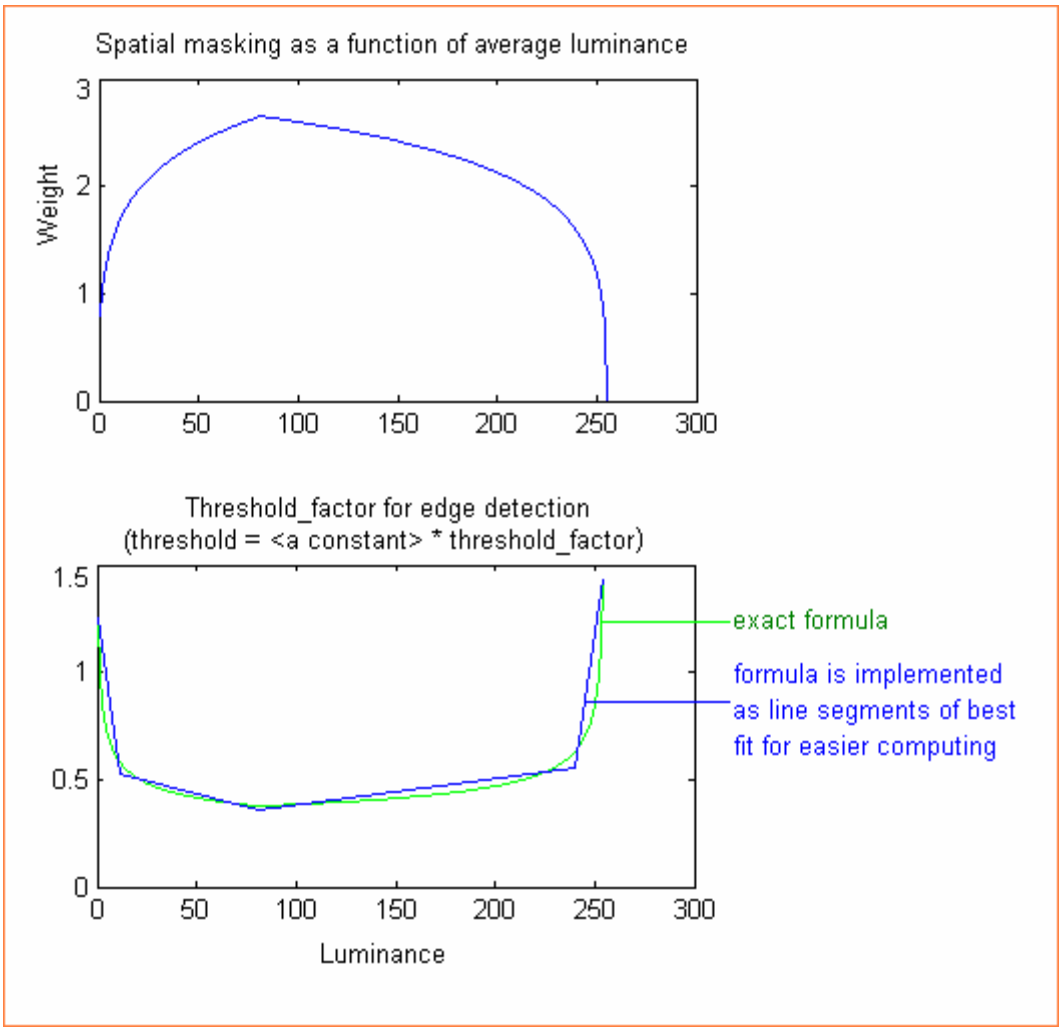


Fig. 36: Existing spatial masking in literature; and modified spatial masking function used in our work for easy computation

9.1.5 New Network-Error Streak Detector:

Significant research has gone into evaluating the effect of packet losses on video. The algorithms used in detecting network errors can be bit-stream based, pixel-based, or a combination of the two. For instance, [46] estimates the mean squared error by just looking at the received bit-stream. A classifier algorithm is used to measure the visibility of a packet loss based on certain stream parameters. The temporal locations of the packet losses, the amount of motion and the accuracy and consistency of motion prediction are some of the parameters considered. Some network-error detectors use blockiness metrics in a modified fashion [51]. The blockiness is measured as a function of time, and any abrupt changes in this signature are used to

indicate a network error. This simple pixel based measure could possibly face problems with video that is varying considerably or has many scene changes.

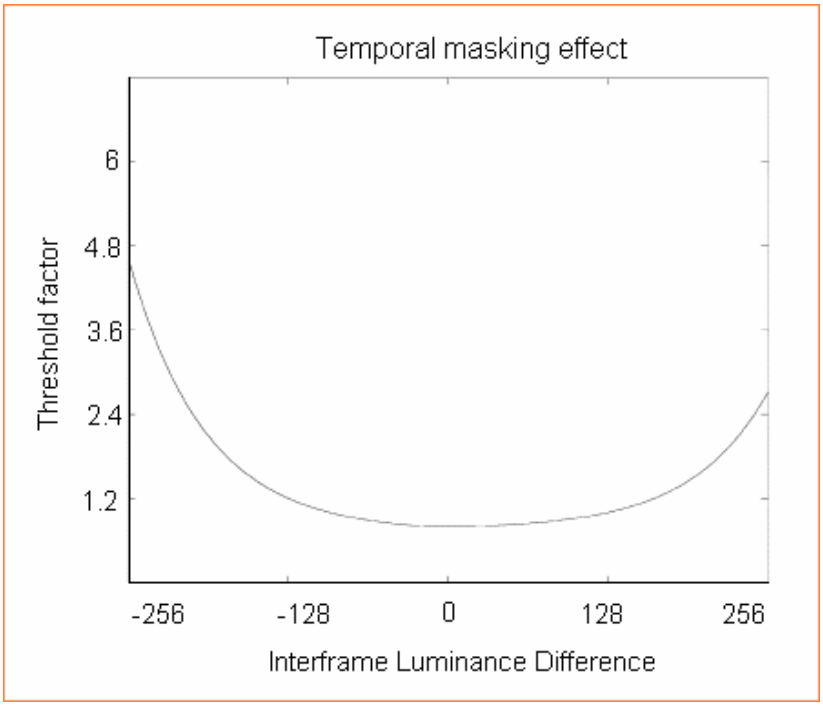


Fig. 37: The relationship between threshold detection and inter-frame differences, as described in [73]

The proposed new algorithms for detecting network artifacts could work on either the pixel values, the bit-stream parameters, or using both:

Proposed algorithm:

Pixel based:

The pixel based network artifact detector (NET_ART_PIX) also works on the spatial and temporal distribution of edges in the video. The occurrences of network artifacts can be evaluated as a modified blockiness measure. The blockiness algorithm described earlier (BLK_ART) is used with a few modifications to function as a network-error detector. For instance, the length of the horizontal edge to be detected is stipulated by a greater threshold, and the visual masking model incorporates the notion that the pixel values across the block boundaries can result in misleading values for the local variance. This can be observed by solid black lines across the video at times. The mean and standard deviation values are

calculated separately on the different sides of the block boundary to ensure that the masking value is registered correctly.

The network error detector can use the modified version of blockiness metric in conjunction with the blockiness metric itself, to account for the cross masking between compression and network errors. It is possible that videos that are extremely compressed have many blocks that could present themselves as a network streak error. To prevent the false detection of these compression artifacts as network error artifacts, a function of the fraction of compression artifacts is appropriately processed out of the network error score to incorporate cross-masking between these two types of artifacts (fig. 38).

Bit-stream based:

When access to the video bit-stream is possible, network errors can be detected by certain flags. For instance, code words with illegal run-lengths indicate network errors. The bit-stream based network error detector (NET_ART_BIT) works on maintaining a record of the spatial temporal behavior between frames in a neighborhood, and evaluating network artifacts as the deviance from normal behavior during packet losses. For instance, the mean difference between consecutive frames are observed for different types of adjacent frame pairs and maintained on record. When a packet loss occurs, the frames in the region between the erroneous frame and the next refresh frame are considered as subject to network artifacts. The artifact measure is evaluated as the deviation from the mean difference value on record for the specific frame-pair under observation (Fig. 34).

Hybrid:

The network error detector could be based on both the pixel and bit-stream values (NET_ART_HYB). These two could be calculated independently and averaged or pooled in a specific fashion. Alternatively, one algorithm could be used as a sanity check for the other. They could be mixed in different ways as well. E.g., the location of packet errors could be identified from the error flags generated in the bit-stream algorithm. Then, the pixel-based algorithm could be evaluated only on the frames between the erroneous frame and the next refresh / Intra / key frame.

The network artifacts detector could be modified in a few ways to improve its functionality and accuracy. Stuck frames are detected either based on pixel differences between consecutive frames or error flags generated from the bit-stream indicating dropped frames. Re-ordered frames are detected either based on inconsistencies observed in inter-frame pixel-difference based signatures, or error flags generated by the bit-stream. The stuck or re-ordered frames are handled by measuring the video quality as a function of the video quality of non-stuck / reordered frame(s) in both temporal direction, and the temporal distance between the frames in comparison. The function could be a linear interpolation between the video qualities of the non-stuck frames in either direction based on the number of frames separating the stuck frame and the non-stuck frames. Cross-validation between pixel based and bi-stream based methods as sanity check improves the functionality of the metric. Any other appropriate modifications borrowed from the blockiness metric could be used for the network streak detector, since they are related.

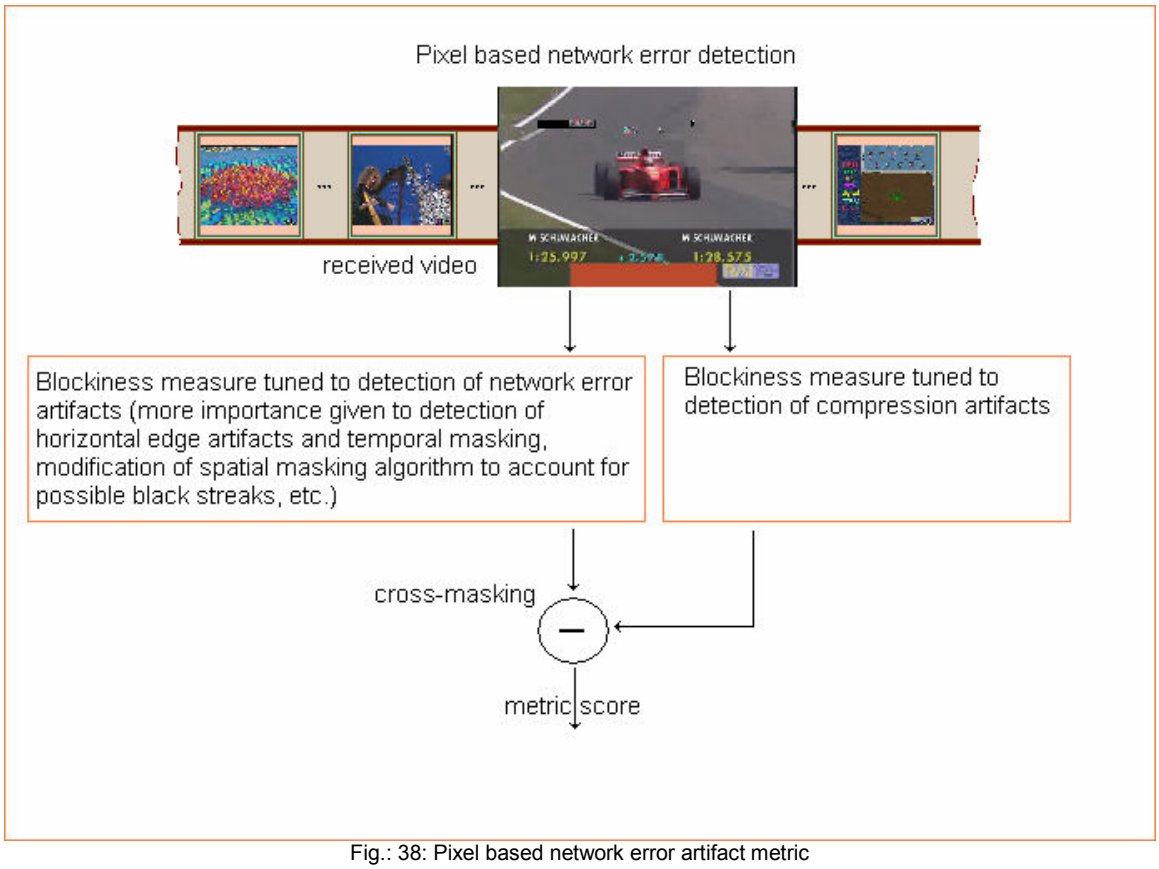


Fig.: 38: Pixel based network error artifact metric

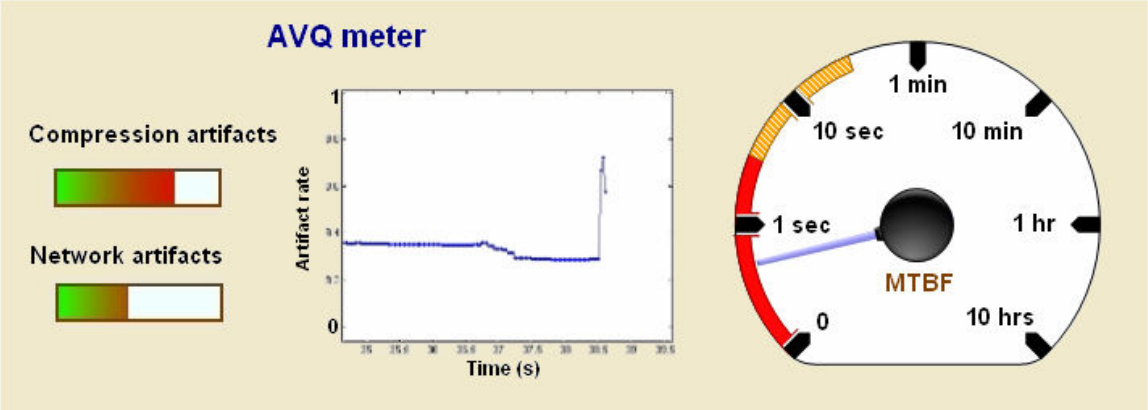


Fig. 39: Sample screenshot of the AVQ meter

9.1.6 New Blurriness Metric:

The blurriness metrics in literature focus on measuring the blurriness either directly or indirectly through a measure of sharpness. A brief literature survey of these metrics can be found in [74]. E.g., [12][30] locates the edges in a given frame and evaluates blurriness as a measure of the average edge spread. A measure of image sharpness is obtained by calculating the local edge kurtosis around edges in [17]. Some metrics compute the blurriness as a function of the histogram of DCT coefficients in the compressed bit-stream [75].

Proposed algorithm:

The proposed blurriness metric, (BLR_ART), works by observing the behavior of video when subject to spatial and temporal enhancement or degradation processes. In one specific implementation, the current frame is subject to a spatial smoothing operation. Then, the difference between the current frame and the smoothened current frame is calculated. This difference is a weighted difference measure, and incorporates spatial masking from the mean and variance of the local pixel values as described in the BLK_ART algorithm, and measures temporal masking from inter-frame pixel differences as described in the NET_ART_PIX algorithm. The idea of this algorithm is that the smoothening process does not have as much effect on blurry images as it has on sharp images (fig. 40, 41). The problem of locating the boundaries of the edge pixels is avoided, and this simplifies the calculation of the metric.

This algorithm has several modifications to make it more effective. A potential problem of this algorithm stems from the notion that smooth original images that are not blurry would produce results

similar to blurry images. This can be avoided by pooling the weighted difference measure between the frame and the smoothened frame in a block-by-block basis, and considering only a portion of the maximum differences. Also, totally smooth video frames, such as those depicting a clear blue sky, are detected by measuring the average local variance so that the BLR_ART algorithm does not detect them as being blurry. The blurriness detection algorithm could be used in conjunction with blockiness estimation algorithms (fig. 31). For instance, if block edges are not detected, then this could either mean that the video is of high quality, or is of extremely compressed low quality that the block edges are themselves smudged out. Using the blurriness estimation algorithm in areas where the blockiness algorithm fails to find any block edges helps improve the detection of video artifacts (fig. 41).

Typical blurriness metrics in literature measure spread of edges, assuming that blunt edges mean low quality. This is not always true, and the original video could be having edges in the background / distance that are intended to be blurry. The BLR_ART could be made to give more importance to regions with high spatial variance to take care of this problem. Alternatively, the consistency of edge angles in the difference between frames could be processed to enhance the BLR_ART blurriness measure. This stems from the notion that intended blurry edges are smooth and continuous in all frames. This can be calculated by the variation in orientation angles between successive edge pixels along the edges detected in the frame or in the difference image between successive frames.

The blurriness metric could be modified in a few ways to improve its functionality and accuracy. Similar to how the degradation process is used to estimate blurriness, enhancement processes can be used to measure sharpness. The degradation process could be used to measure other types of artifacts as well, such as ringing and mosquito noise. The observation of the consistency of edge angles in edges in the frame / difference-frame could be used to distinguish between intended blurry edges and edges blurred due to image/video coding. These blurriness metrics can be applied to JPEG2000 type and non-block oriented compression schemes as well. Any other appropriate modifications borrowed from the blockiness metric could be used for the blurriness metric as well, since they all fall in the class of related spatial temporal coherence metrics.

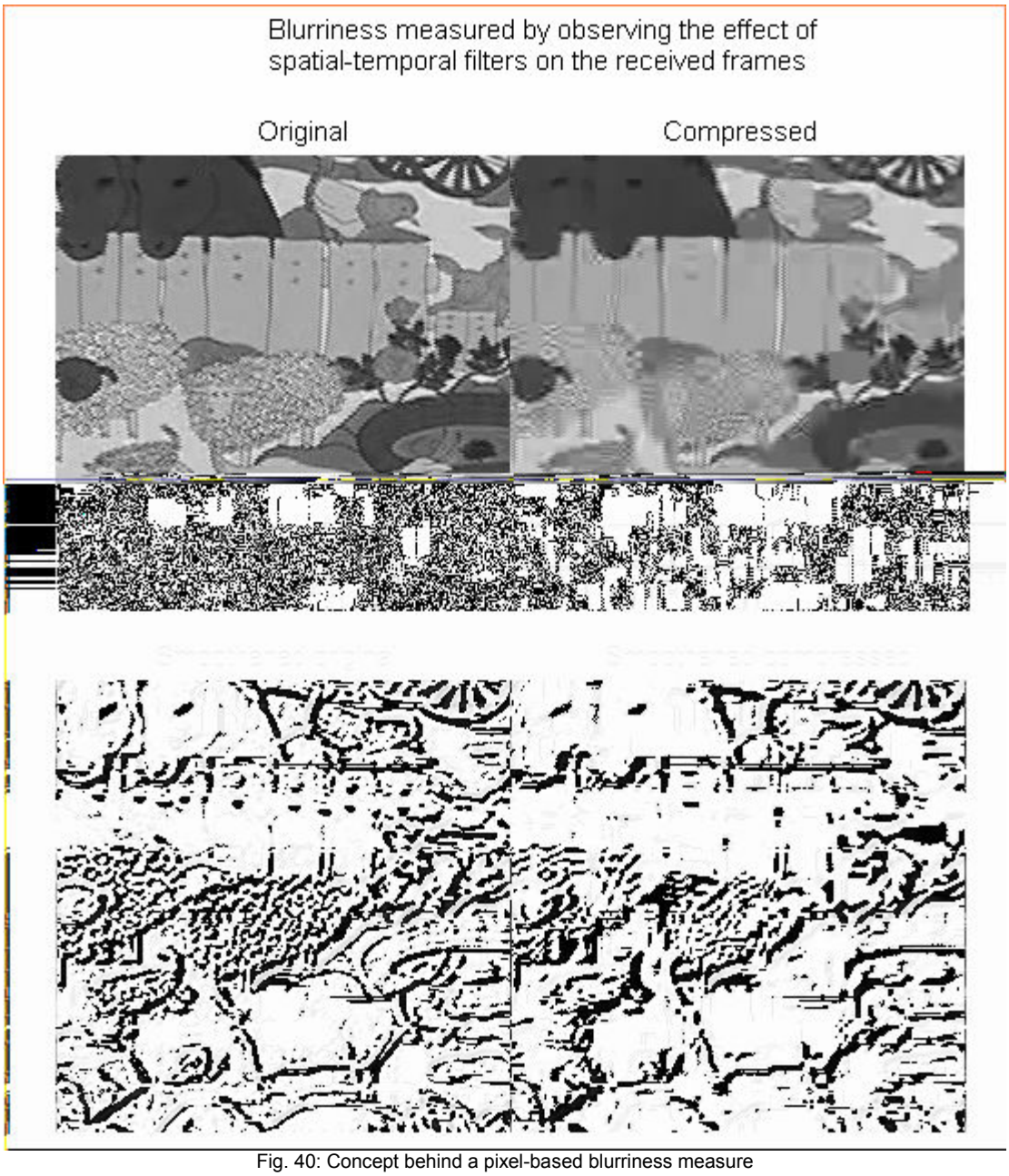


Fig. 40: Concept behind a pixel-based blurriness measure

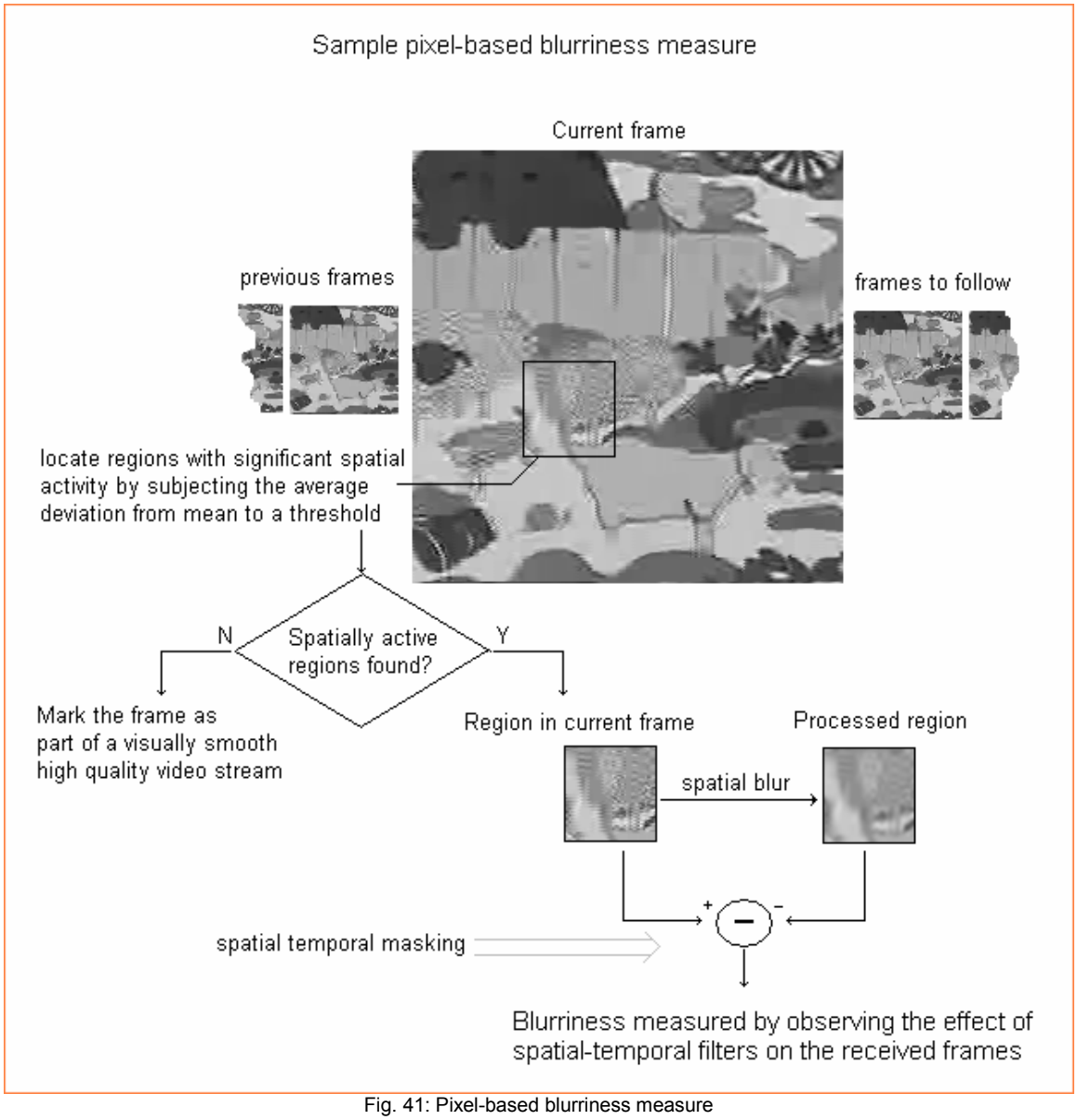


Fig. 41: Pixel-based blurriness measure

9.1.7 Delta-Autocorrelation-Method

The delta-autocorrelation method aims at designing a metric module using functions that characterize signal coherency or dependency. In its general form, the video quality evaluation algorithm works by comparing the received video signal (henceforth denoted by 'Y_n') with a reference signal ('S'). The reference signal can either be the original video frame(s) ('X_n', 'X_{n-1}', … etc.), some information extracted from the original video ('XS_n') or the video frames received and decoded at the output before the current frame ('Y_{n-1}', 'Y_{n-2}', 'Y_{n-3}', … etc.).

One of the methods of comparing Y_n and the reference signal, S, includes comparing smaller sections of both. All portions of the two can be compared or only selective regions deemed as visually important can be considered. The comparing algorithm includes computing the autocorrelation plots of the smaller sections and comparing the autocorrelation functions of corresponding sections. Alternately, the Absolute Mean Difference Function (AMDF) can also be used, instead of the autocorrelation function. Let 'R' denote the function used in the comparing algorithm. In one embodiment, R would be the mean value of the difference in autocorrelation plots of the sections in comparison. In general, the autocorrelation (and cross-correlation) functions can be replaced by other functions that characterize signal coherency or dependency.

In general, the video evaluation algorithm uses at least one feature based on the current and neighboring frame statistics, such as, but not limited to:

- $R(X_n, Y_n)$
- $R(Y_{n-1}, Y_n)$
- $f(R(Y_{n-1}, Y_n), R(Y_{n-2}, Y_{n-1}))$, where one example of f can be a simple absolute difference function
- $f(R(Y_{n-1}, Y_n), R(X_{n-1}, X_n))$
- $R(Y'_{n-1}, Y_n)$
- $f(R(Y'_{n-1}, Y_n), R(Y'_{n-2}, Y'_{n-1}))$, where Y' implies that instead of just using the previously decoded frames as reference, the motion compensation vectors are used to refine the usefulness of the reference signals
- global statistics using combinations of arithmetic mean (AM), geometric mean (GM), and harmonic mean (HM) of autocorrelation differences (operations on local statistics capturing macro-block or frame properties).

One of the useful products of this approach would be *a no-reference* system for monitoring video quality where no reference needs to be made to the original undistorted video information. One of the ways to replicate original video properties is to combine the information from a multiple-frame window of the artifacted original, possibly in a complex non-linear fashion, with correlation to subjective quality being a function of the complexity.

Sample evaluation runs using the delta-autocorrelation measure:

Consider the 'Foreman sequence' (Fig. 42):

Fig.42: Input video frame to a H.264 encoder

This is subject to a variety of processing to get different artifacts. One of the ways to calculate differences between portions of the processed and reference video is to compute the difference in autocorrelation plots as shown (Fig. 43). Based on this difference measure, error maps can be created for the processed video to denote the areas that are visually bad (Figs. 44, 45). The error maps can be pooled as required. A simple average value over the frame can be used to denote an overall error measure. This is calculated for two H.264 outputs of the 'Foreman' sequence at different bit rates (Fig. 46):

As expected, the error measures shown in fig. 46 show a monotonic dependence on the degree of video compression (bit rate), with greater bit rates corresponding to lower errors. Interestingly, this property is true not only for the full-reference and reduced-reference algorithms, but also for the no-reference algorithm. These results are exemplary of a broad class of algorithms represented by the current description. In the different modules of the AVQ meter that work on difference maps between frames, the delta autocorrelation method can be used as a better alternative to the difference map.

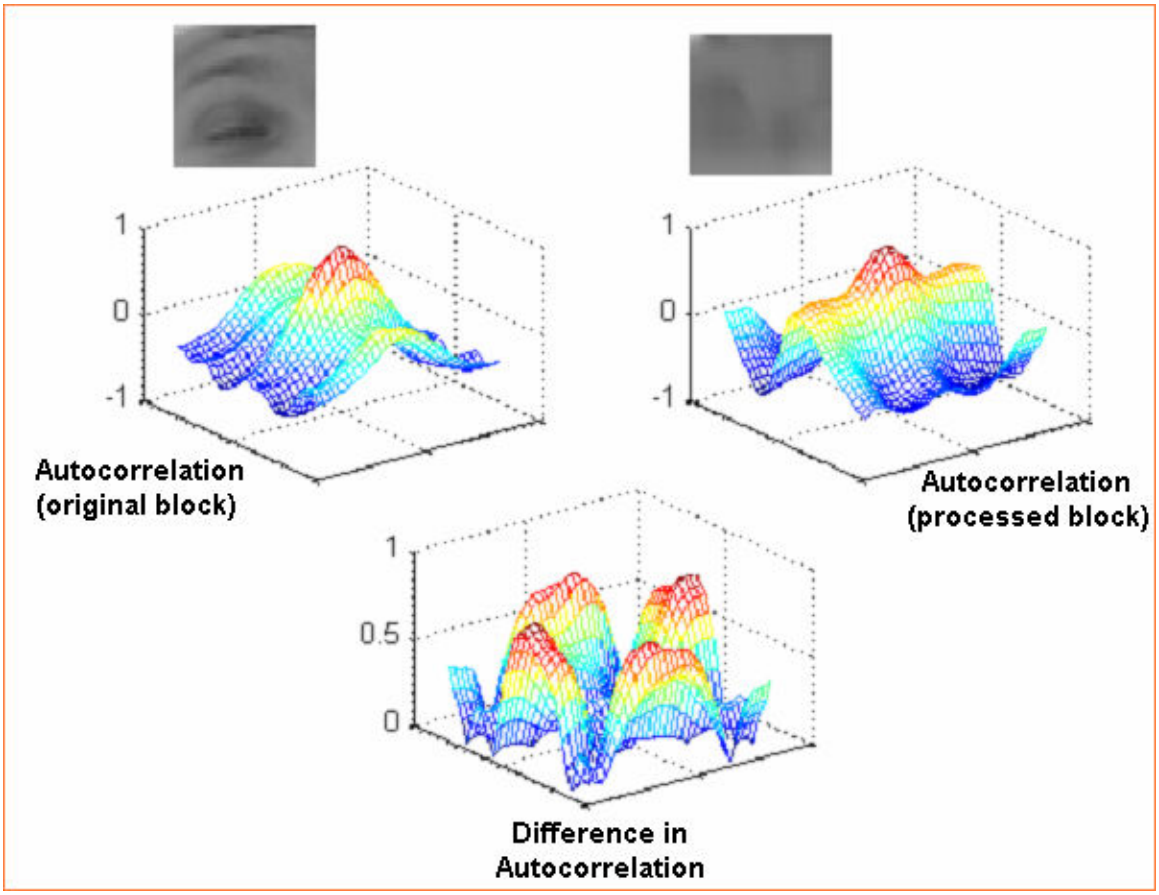


Fig.43: Comparison of sections between processed and reference video (Delta-autocorrelation method)

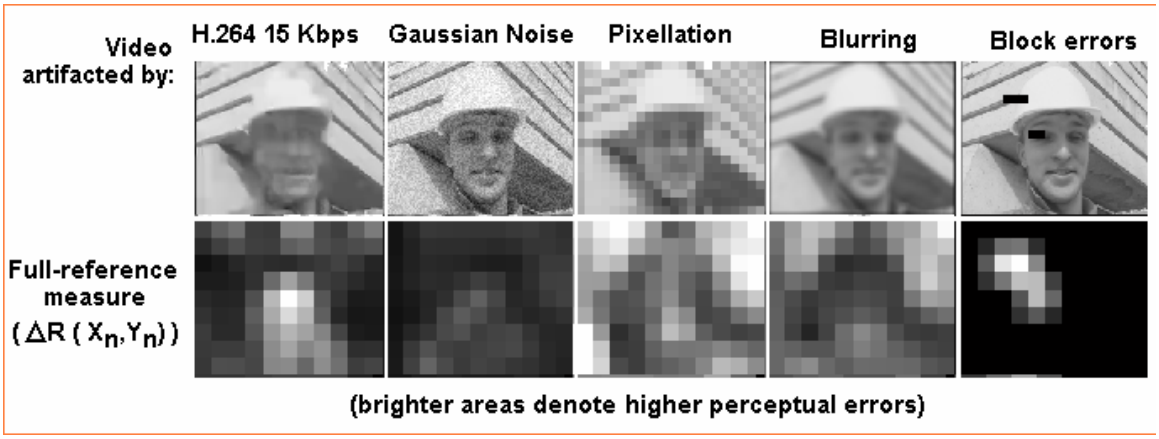


Fig.44: Sample evaluation maps using a delta-autocorrelation full-reference metric

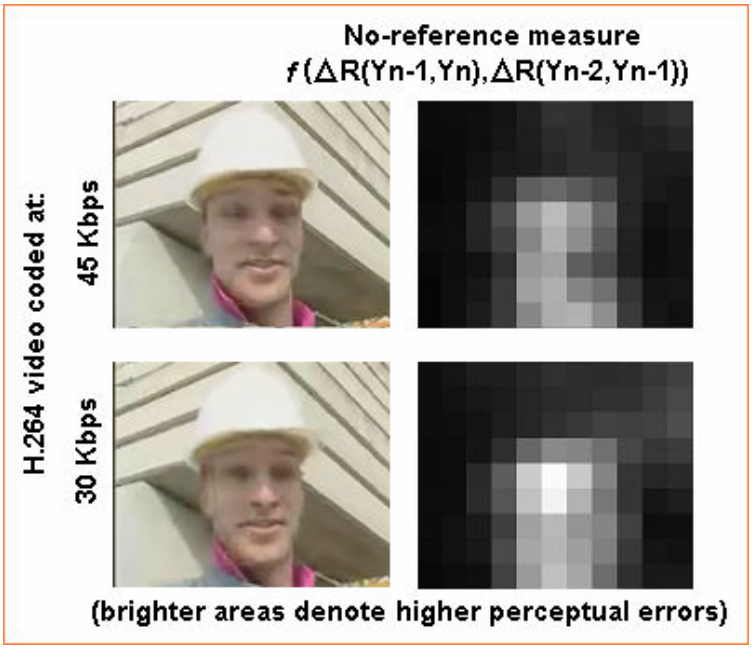


Fig.45: Sample evaluation map using a delta-autocorrelation no-reference metric

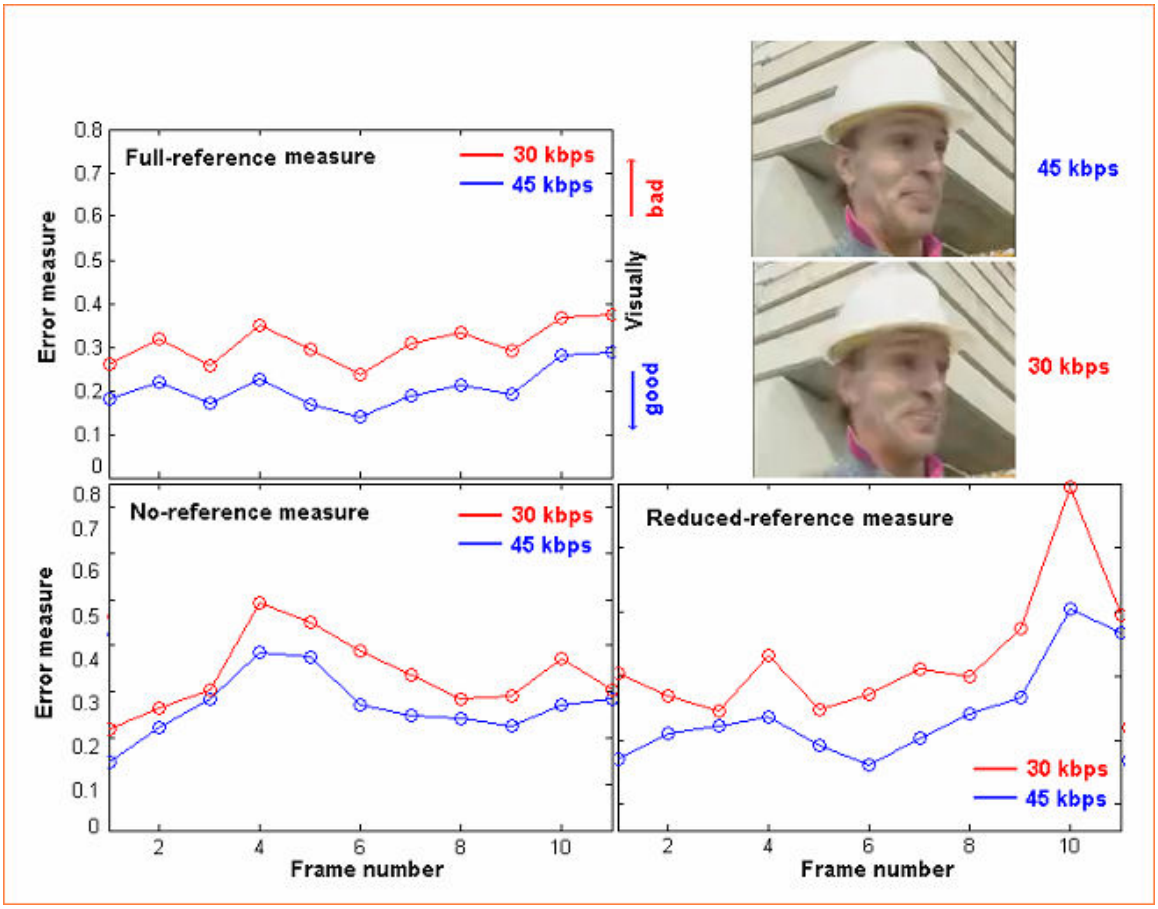


Fig.46: Sample evaluation of full-, reduced-, and no-reference metrics based delta-autocorrelation

9.2 AVQ meter: Combination of different components:

When different components of the AVQ meter are used in conjunction, different values of correlations are obtained (Fig.47). The different modules of the algorithm that are currently implemented in real-time, are described below. The other modules have been implemented in MATLAB, and the process of including them in the real-time C prototype is on going research.

1. $$Quant\_step = \log(\lfloor Q \rfloor) \quad (6)$$

where 'Q' is the average quantization step size averaged over all the macro-blocks in the frames, clipped at a certain threshold.

2. $$N\_dct\_coeff = \log(\lfloor N\_dct \rfloor) \quad (7)$$

where 'N_dct' is the average number of DCT run lengths per macro-block in a frame. This is clipped at a certain threshold as well.

3. $$Blockiness = \log(1 + k_B B) \quad (8)$$

where 'B' is the average number of exactly vertical edge pixels per frame that are deviant from the expected number of edge pixels, as explained in the blockiness algorithm in the previous sections

4. $$Streakiness = k_1 Ne^p - k_2(B) \quad (9)$$

where 'Ne' is the average number of exactly horizontal edge pixels per frame that are deviant from the expected number of edge pixels, as explained in the network error detection algorithm in the previous sections, 'B' is a measure of the blockiness in the video, and k_1, k_2 and p are constants decided based on data fitting with the subjective test database.

The various modules are combined in the following fashion:

$$Combine\_Module = a_1 \times (module1) + a_2 \times (module2) + a_3 \times (module1) \times (module2) \quad (10)$$

where a1, a2 and a3 are constants determined by the best fit with the test databases. Apart from these modules, the bit rate is also explored as a video quality detector. The bit rate is used to determine the relationship between the module in use and the *MTBF* calculated.

The equation for the *MTBF* calculated is:

$$\log(MTBF) = C_1 \times (Combined\_Module) + C_2 \quad (11)$$

where C1 and C2 are linear functions of the bit-rate, if that information is accessible.

Apart from the calculation of *MTBF* values, certain diagnostics such as the level of compression artifacts and level of network artifacts are also displayed in real time. These are again linear functions of certain modules clipped at values based on the observations from the test database. The network streak detector is used as the module to flag the occurrences of network errors, and the module consisting of quantization step size, number of DCT coefficients and blockiness is used to flag the occurrences of compression artifacts.

It should be noted that all the constants that are in use do not depend on the test database. In other words, the constants of the algorithm are set, and the AVQ algorithm functions well for either test database, in terms of the correlations with subjective values of *MTBF*. The algorithm has been observed to work visually well for other test video clips as well.

Some key points of the AVQ algorithm can be observed from Fig. 47. The increase in correlation with subjective scores gets increasingly difficult with additional modules. The correlation numbers are high and impressive, considering that the AVQ meter does not know whether it is dealing with a compression or network error database while performing the tests. The network error detector is quite effective in the sense it adds value to the network error database, while not affecting the compression artifacts database. It is also observed that the inclusion of bit-rate information does not add much value if the quantization step size is already known. Equal values of correlations with subjective *MTBF* might warrant some additional information with respect to where the detection of artifacts is more accurate. For instance, the combination of quantization step size and the number of DCT coefficients produce the same correlation for the network error database, as the combination of the pixel based blockiness and network error detectors. The latter combination detects the network errors more accurately, and does not predict the compression errors that accurately. Its prediction of network errors can be shown in fig. 48. It has to be noted that the temporal effect of the network errors is implemented as a first order filter to represent the mechanism by which network errors impact a human observer.

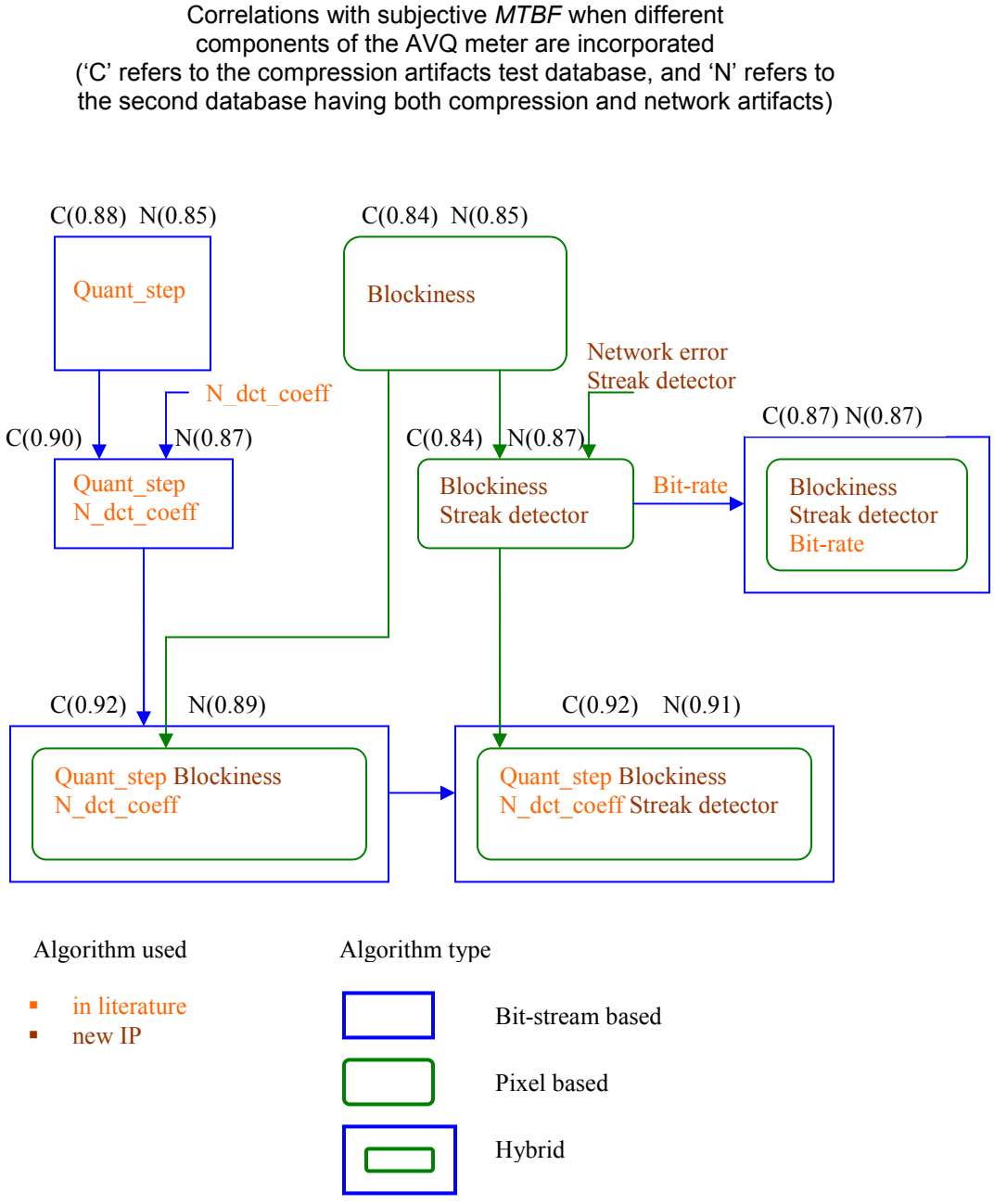


Fig. 47: Correlations with subjective *MTBF* scores when different components of the AVQ meter are used in conjunction: evaluated for different test sequences from both the test databases

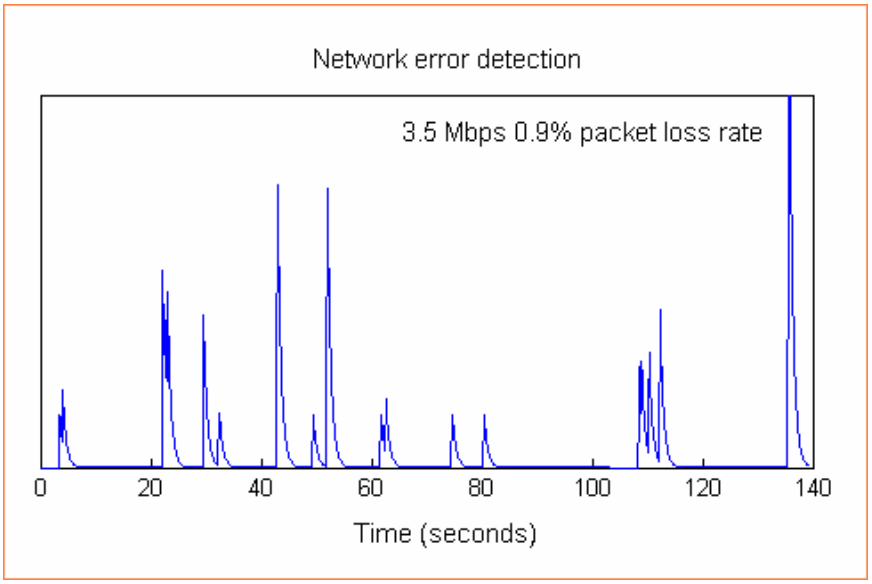


Fig. 48: Accurate detection of network error artifacts by the AVQ meter

9.3 Performance of the AVQ metric:

The accuracy of the estimated *MTBF* values got by the AVQ metric can be measured in terms of the correlation between estimated and actual *MTBF* values.

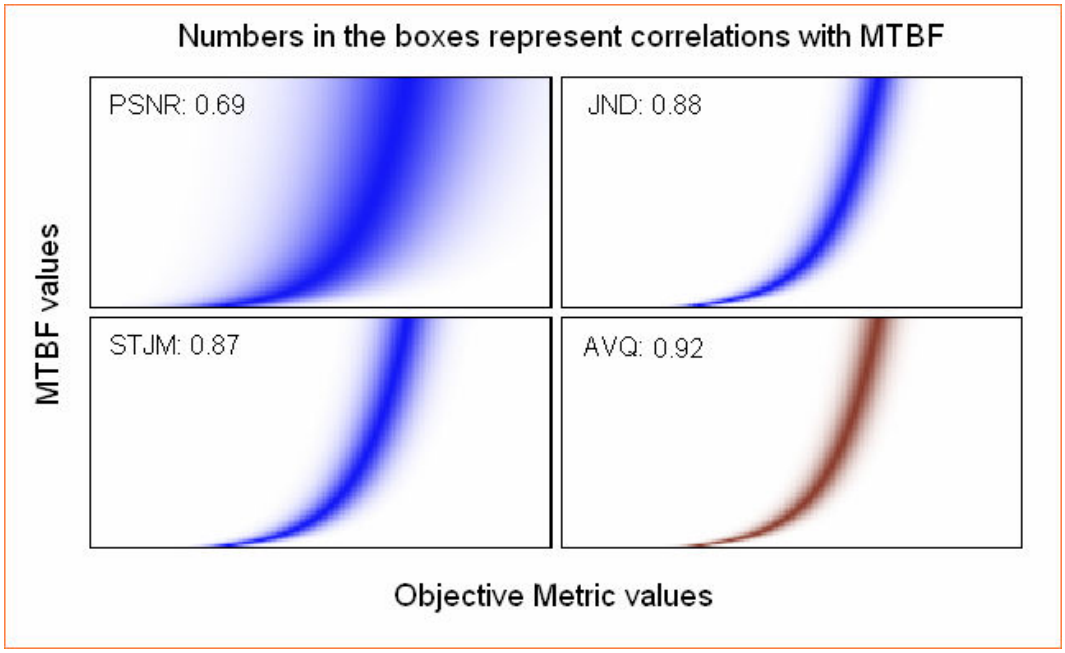


Fig. 49: The accuracy of different objective metric represented using their correlations with the actual *MTBF* values generated for the compression artifacts database.

This correlation number can be represented in terms of a spread plot. Such a spread plot is shown for test database 1, which had video clips with only compression artifacts (fig. 49). A tighter spread reflects the nature of the metric to accurately estimate the *MTBF* scores objectively. The *MTBF* values can also be shown as a running average with time. Such a plot is shown for the test database 2 which had a mixture of compression and network artifacts. (Fig. 50). A screen shot from the AVQ meter is shown in fig. 51.

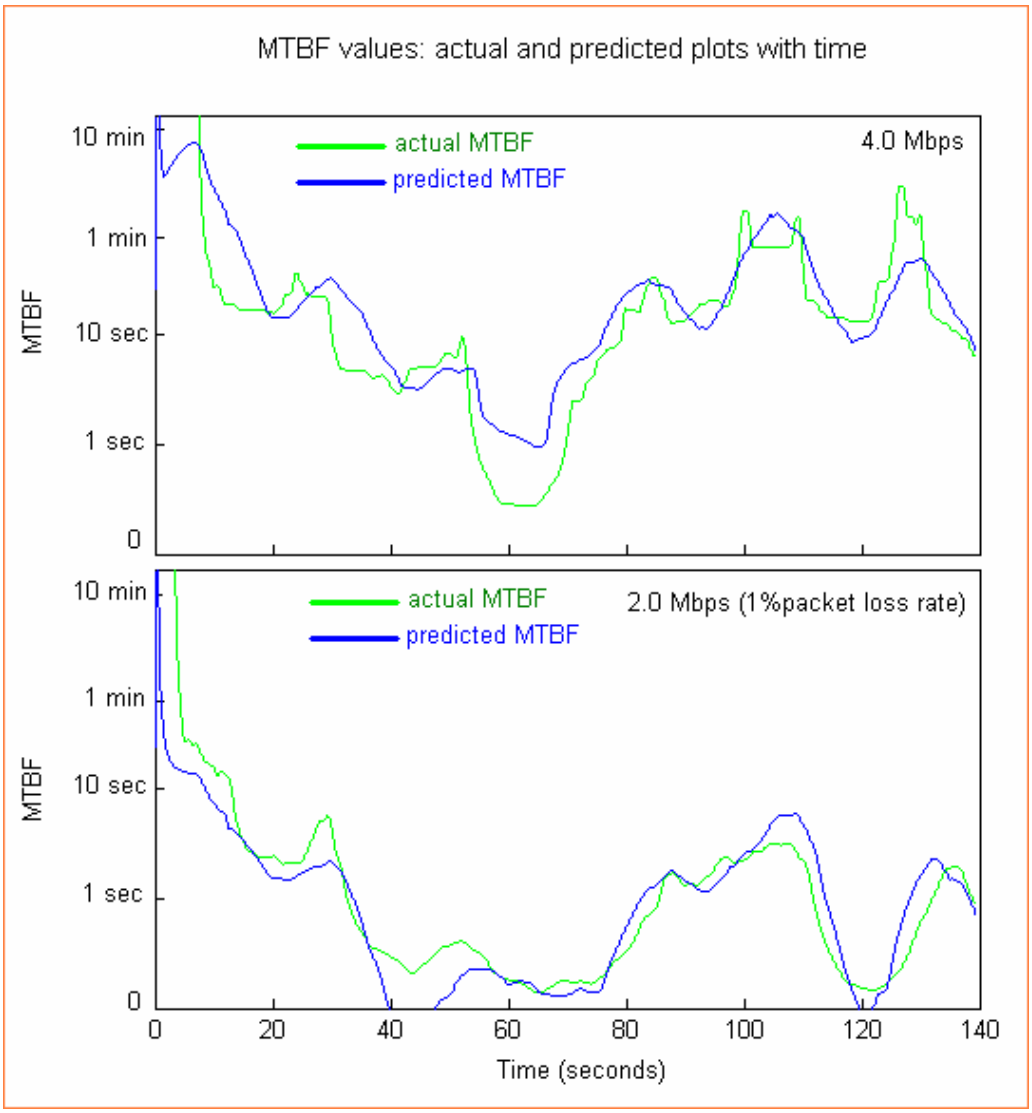


Fig. 50: Sample plots of predicted *MTBF* and actual *MTBF* with time shown for different test sequences comprising of both compression and network artifacts

Fig. 51: A screen shot of the AVQ metric implemented in real-time

The dial shows the *MTBF* value, and the amount of compression and network artifacts are displayed as the sliding values of 'CA' and 'NA' respectively. Compression and network error flags are also displayed on the main dial as a 'C' or an 'N'. The current prototype is implemented in such a way that media streams or play lists can be dragged and dropped onto this dial, and it starts playing the video file and displays the corresponding quality in real-time.

CHAPTER X

Future Work

10.1 Extensions to H.264 and other compression standards

H.264, MPEG-4 Part 10 or the Advanced Video Coding (AVC) is a digital video codec standard that is known to achieve compression rates that are twice as good as MPEG-2. With the advent of newer and better compression standards like H.264 and Windows Media Video 9 [76, 77], it is desirable to have objective video metrics that work over a range of codecs.

A variety of improvements over MPEG-2 have been performed to make H.264 a better codec. One aspect of the codec involves better numerical algorithms to make the computation faster. This is achieved by the use of a simple 4x4 integer transform which is easy to compute using simple bit shifting operations, and has the added advantage of being able to produce an exact inverse transform as opposed to the occurrences of rounding errors in MPEG-2's inverse DCT. Another aspect of the H.264 involves the usage of context adaptive entropy schemes that enhance the compression. The improvement in video quality comes from the usage of variable block sizes that can go to as small as 4x4, quarter pixel estimation, and the usage of in-loop de-blocking filters. The last two aspects of H.264 discussed above affect the way objective metrics ought to handle their algorithms on H.264 encoded video. Some blockiness estimation metrics can function as a detector for the de-blocking operation as well. For example [78] proposes a de-blocking algorithm based on the number of connected blocks in a relatively homogeneous region, the magnitude of abrupt changes between neighboring blocks, and the quantization step size of DCT coefficients. [79] and [80] discuss issues related to quality evaluation of H.264 video, and methods to map current metric scores like *PSNR* to a useful *MOS* score.

If the objective metric uses bit-rate information, then the effect of context adaptive entropy coding on bit-rate savings ought to be studied in detail to fine tune the metric. As for the de-blocking operation, the appearance of block edges is reduced by selective filtering. If the objective metric is a *full-reference* metric, then such metrics should understand that smoothened out block edges might result in a poorer metric score than a video which was not de-blocked. If the metric is of a *no-reference* nature and uses a measure of blockiness to calculate its score, then care has to be taken to check for video frames that have not much

blocky artifacts, but are yet unnaturally smoothened out. Surveys of existing objective metrics on H.264 encoded video have been performed, and they primarily verify if the metrics satisfy some basic sanity checks [47]. For instance, the metric score has to appropriately scale with bit-rate. It has been noted in that survey that the blockiness metrics in general do not have any temporal masking included, or are parameterized with respect to the viewing distance.

The future work related to this thesis includes a comprehensive study of objective metrics tuned to various video codecs, and the development of a *no-reference* metric that would generate objective scores that has a correlation with subjective scores of *MTBF* for different codecs such as H.264 and MPEG-4.

10.2 Implementation of newer modules in the real time AVQ metric

The real-time implementation of the AVQ metric currently involves modules consisting of the quantization step size, the number of encoded DCT coefficients, a new blockiness metric and pixel based streak detector algorithm, and bit rate tuning. The various other modules described earlier, such as a delta-autocorrelation method and a metric based on the observing the error concealment processes in video decoding are part of future work. The process of fine tuning the AVQ metric with subjective scores measured over a broad range of video content and processing schemes is a continuing topic of interest.

CHAPTER XI

Conclusion

The importance of the measurement of video quality has been well recognized in the realm of digital communications. Our work involves focused subjective testing that relates to the mean time between visual artifacts as well as objective metrics that mimic judgments of human subjects. Degradation in video signals can be broadly classified into the reduction in information when they are compressed before the transmission process, and the errors caused while they are in transit over a network. Our objective metric is also capable of distinguishing between these two categories of visual distortion.

A new intuitive subjective testing scheme termed Mean Time Between Failures (*MTBF*) is used to designate subjective scores to an exhaustive test database. These subjective scores are then used as a basis for the comparison of existing objective metrics. A comparison study of *MTBF* scores with the current standard Mean Opinion Scores (*MOS*) has been performed to observe the statistical spread of the different kinds of subjective scores, and the correlations between the two. These results suggest that *MTBF* has a direct and predictable relationship with *MOS*, and that they have similar variations across different viewers, when computed over any clip

An important product of this project is the design and development of new objective metrics that do not require any reference to the original video signal. Our quality algorithm, termed the Automatic Video Quality (AVQ) metric has been successfully implemented in real-time. AVQ is shown to have a high level of correlation with actual subjective scores of *MTBF,* with the correlation approaching correlation values of metrics that use full or partial reference. When used to display video streams, the metric mimics human viewers and displays the quality in terms of the predicted *MTBF*, and also provides diagnostics about the likely source of the artifacts.

APPENDIX A([1])

AVQ in Real-Time: Implementation Issues

AVQ Architecture

Figure 52 shows the architectural components that make up the entire software stack. The primary goal towards building a modular architecture is to allow for flexibility to port the core engine to different execution environments such as in a network device (a router), a streaming media server farm, or a streaming media client.

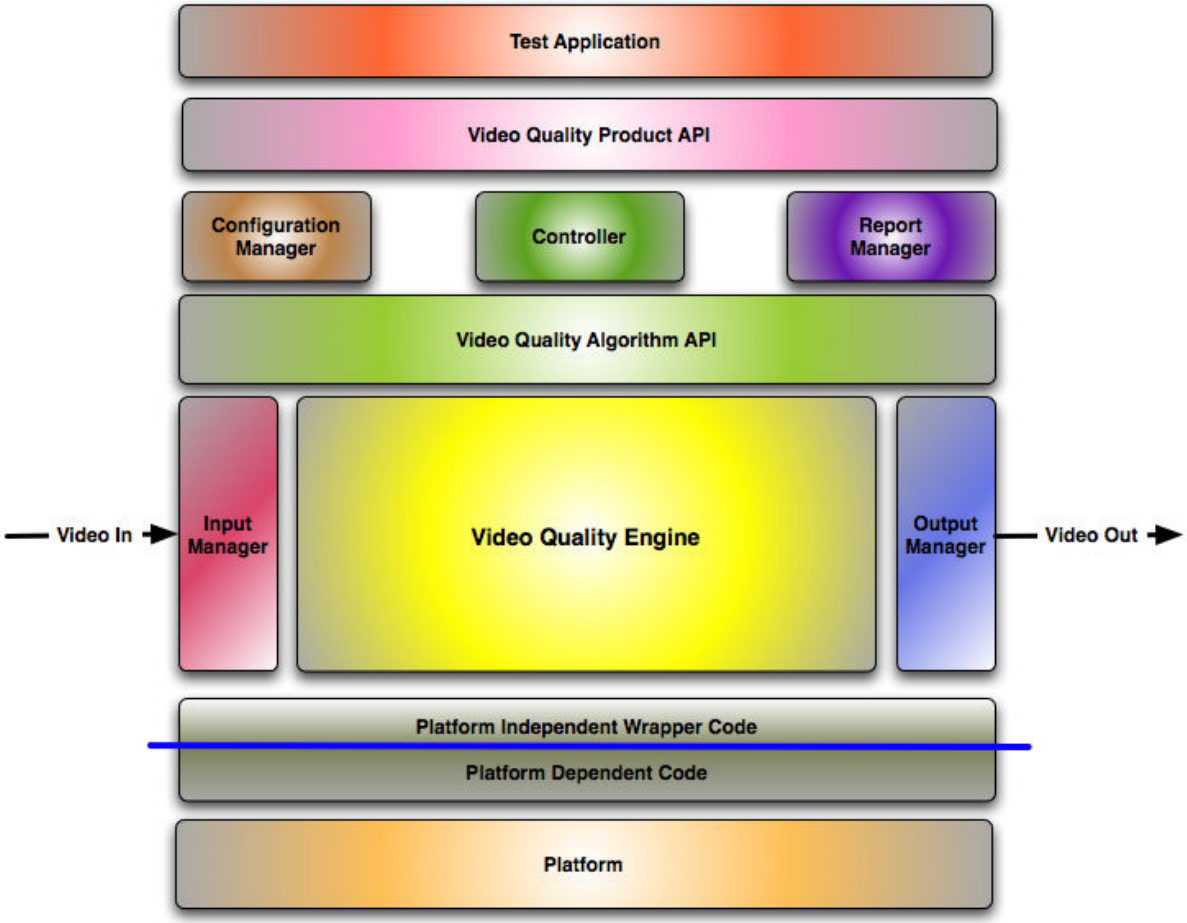


Figure 52: AVQ Architectural Component Overview

[1] documented by Pravin Mane <pmane4@ece.gatech.edu>
as part of the work on the real-time implementation of the AVQ metric

To achieve this goal of portability, AVQ has been architected in such as way that the core modules (Configuration manager, Controller, Report manager, Input Manager, Output manager and Video Quality Engine)

1. must be written in ANSI C.
2. must use Platform Independent Wrapper code layer to make any system calls so that the calls can be easily ported to different platforms and execution environments.

Following is a brief description of the components shown in Figure 52 above.

Controller

The controller provides following functionality through a set of APIs:

1. Acts as the main controller to control the entire run (or a sequence of runs) of the product.
2. Provides functionality to setup the execution environment for the video quality engine. The setup includes initialization and making appropriate input and output connections so that the video quality engine sits seamlessly in the video data flow within execution environments.
3. Provides control over input manager.
4. Provides control over output manager.
5. By using input manager and output manager, controls the flow of video information within the framework.
6. By using configuration manager, controller controls the changes to the video quality engine's configuration parameters.
7. By using report manager, sets up the reporting framework to report the results to external components registered with report manager. The reports include the results as well as any error conditions encountered.

Configuration Manager

The configuration manager provides a set of APIs and functionality to set and retrieve various parameters required for setting up or fine tuning the core engine.

Report Manager

In every major product there is always a user interface associated with the product. The UI to see the reports generated by the AVQ software stack can either be on same machine or on a different machine. The report manager provides a way to connect to such components that are interested in the results generated by video quality engine module. The report manager also provides a set of APIs for registering and communicating with external components. The external components can be interested in

1. Any notifications (error, progress etc.) from the framework or
2. The result reports (either at the end of a run or periodic reports) generated by the video quality engine component.

Input Manager

The input manager provides a set of APIs to connect the core engine module in between an existing path of video data. The input manager focuses on setting up and getting the video data from some external video source or a component in a video flow path. The input manager is expected to be a very thin layer of code.

Output Manager

The output manager provides a set of APIs to connect the core engine module in between an existing path of video data. The output manager focuses on setting up and passing the video data to some external video sink or another component in the flow of video data. The manager is expected to be a very thin layer of code.

Video Quality Engine

This is the core module of the software stack that does actual video quality measurement work. This module is expected to receive video related information from the video decoder and video output modules.

In the algorithm development phase, this is the most often changed module to cater to ongoing technological innovation, and hence is the most important part of the software stack and will be of utmost importance while licensing the technology to external customers.

Platform

The platform layer is a conceptual layer that takes into account all the hardware and Operating System on which the AVQ software is going to run.

Platform Wrapper

The platform wrapper layer provides a way to ease porting and execution of the software stack. This layer is designed in such a way that it provides a set of OS independent APIs to various software modules. By using the APIs exposed by this layer, the software stack can be ported to various platforms and execution environments easily. To port the software stack, the port should only have to implement Platform Dependent Code to cater to Platform Independent Wrapper code.

Test Application

The test application provides a UI for testing and showing demo of the software stack. It is expected that the test application will be thrown away when the technology is incorporated in an actual product. Currently, the Video Lan Client [81] is used as a test application/execution environment for the AVQ software.

Execution Environments

Following is a list of possible execution environments in which the AVQ technology described in this document is envision to be executed.

1. Microsoft DirectX based servers and clients.
2. Various streaming media servers such as VideoLan Server, Quicktime Server, GStreamer server etc.
3. Various streaming media clients such as VideoLan Client, Quicktime player, Digital Media adapters, Set top Boxes etc.
4. Various network devices such as switches, routers or caching servers in a network.

AVQ in Media Player

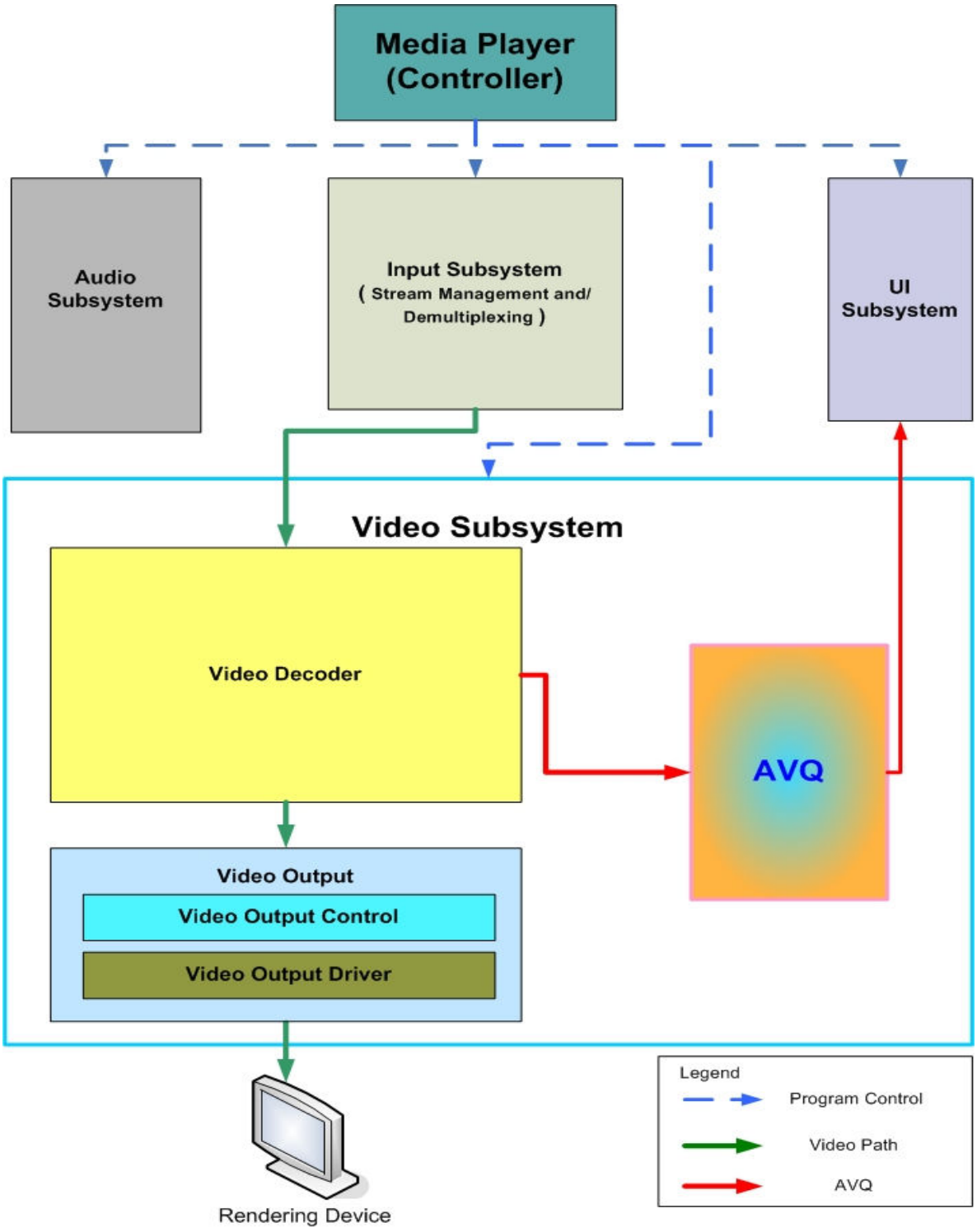


Figure 53: AVQ in a Media Player

Figure 53 shows a typical deployment of AVQ inside a media player capable of video decoding and rendering.

As shown in Figure 53, a typical video playback with AVQ technology will work as follows:

1. User selects a video file (also called as bit stream) to be played in the media player.
2. The media player determines the type of media file (MPEG2, MPEG4 etc.). Depending on the type of the media file, the media player will initialize appropriate video decoder and the AVQ component.
    a. Note that the decoder itself can initialize AVQ if the decoder has been equipped with AVQ, this is mostly an implementation issue.
3. The media player also initializes other components such as Input Subsystem, Audio subsystem and UI subsystem.
4. As video data is read from the file, video bit stream is passed by the Input Subsystem to the video decoding component (part of the Video Subsystem) which decodes the video. The video decoder pass some key data extracted from each video frame to the AVQ component.
5. Using the innovative patent pending technology, AVQ determines the quality of the video bit stream.
6. AVQ then sends periodic reports to the reporting framework/ UI subsystem to display the quality metric to the end user.

Figure 53 shows a typical deployment of AVQ inside a media player such as a Video LAN Client (VLC) player [81]. VLC is a media player with rich support for pluggable software based video decoders. In its current state, AVQ has already been incorporated in a VLC media player as a *libavq* library.

REFERENCES

[1] N. Suresh and N. Jayant, "Mean Time Between Failures: A Subjectively Meaningful Quality Metric", IEEE International Conference on Acoustics, Speech, and Signal Processing, Toulouse, France, May 14-19, 2006

[2] http://www.its.bldrdoc.gov/vqeg/ , Jan. 2005

[3] JNDmetrix algorithm http://www.sarnoff.com/ , Jan. 2005

[4] S. Wolf, and M. H. Pinson, "Spatial-temporal distortion metrics for in-service quality monitoring of any digital video system", *Proc. of SPIE Int. Symp. on Voice, Video, and Data Commun.,* Boston, MA, Sep. 1999

[5] H. R. Wu, M. Yuen: "A generalized block-edge impairment metric for video coding." *IEEE Signal Processing Letters 4(11):*317–320, 1997.

[6] N. Suresh, N. Jayant and O. Yang, "AVQ: A Zero-reference Metric for Automatic Measurement of the Quality of Visual Communications", Invited Talk, Third International Workshop on Video Processing and Quality Metrics for Consumer Electronics, Scottsdale, AZ, USA, Jan. 25-26, 2007

[7] VQM metric http://www.its.bldrdoc.gov/n3/video/vqmsoftware.htm , Jan. 2005

[8] M. Pinson and S. Wolf, "Comparing Subjective Video Quality Testing Methodologies", S*PIE Video Communications and Image Processing Conference,* Lugano, Switzerland, Jul. 8-11 2003

[9] O. Nemethova, M. Ries, A. Dantcheva, S. Fikar and M. Rupp, "Test Equipment of Time-Variant Subjective Perceptual Video Quality in Mobile Terminals", *International Conference on Human- Computer Interaction*, Phoenix, USA, November 14-16, 2005

[10] J. E. Caviedes, F. Oberti, "No-reference quality metric for degraded and enhanced video," *Visual Communications and Image Processing, Proc. SPIE Vol. 5150,* p. 621-632: 2003

[11] S. Yang, Y Hu, D. Tull and T. Nguyen, "Maximum Likelihood Parameter Estimation for Image Ringing Artifact Removal", *IEEE Trans. on CVST*, vol. 11, No. 8, pp. 963-73, Aug. 2001

[12] G. de Haan, *Video processing for multimedia systems*, University Press Facilities, Eindhoven, 2000

[13] A. Watson, G. Yang, J. Solomon, and J. Villasenor, .Visibility of Wavelet Quantization Noise., *IEEE Trans. Image Proc*., vol. 6, pp. 1164-75, Aug. 1997.

[14] M.D. Fairchild, .A Victory for Equivalent Background -- On Average., *Proc. Of IS&T/SID 7th Color Imaging Conference*, Scottsdale, pp. 87-92, 1999

[15] B. Spehar Spehar, J.S. DeBonet, and Q. Zaidi, .Brightness Induction from Uniform and Complex Surrounds: A General Model, Model., *Vision Res. 36*, pp. 1893-1906, 1996

[16] G.M. Johnson and M.D. Fairchild, .Sharpness Rules., *IS&T/SID 8th Color Imaging Conference*, Scottsdale, Arizona, pp. 24-30, November 2000

[17] Caviedes, J.; Gurbuz, S.; "No-reference sharpness metric based on local edge kurtosis," *International Conference on Image Processing, Vol. 3,* 53-56: 2002

[18] Stephen Wolf and Margaret H. Pinson, "Low bandwidth reduced reference video quality monitoring system", *First International Workshop on Video Processing and Quality Metrics for Consumer Electronics,* Scottsdale, AZ, USA, January 23-25, 2005

[19] N. Suresh and N. Jayant, "Mean Time Between Failures: A Functional Quality Metric for Consumer Video", *Invited Talk, First International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, Scottsdale, AZ, USA, January 23-25, 2005

[20] G.W. Cermak, "Subjective Quality of Video as a Function of Frequency of Artifacts," *International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, Scottsdale, AZ, USA, Jan. 25-26, 2007

[21] VQEG, "Final report from the Video Quality Experts Group on the validation of objective models of video quality assessment, phase II", http://www.vqeg.org/, August 25, 2003

[22] T. Vlachos, "Detection of blocking artifacts in compressed video," *IEE El. Let.*, vol. 36, no. 13, June 2000.

[23] Shizhong Liu Bovik, A.C., "Efficient DCT-domain blind measurement and reduction of blocking artifacts," *IEEE Transactions on Circuits and Systems for Video Technology 12(12):* 1139-1149, 2002

[24] S. Winkler et al., "Perceptual video quality and blockiness metrics for multimedia streaming applications," in *Proc. ISWPMC*, Sept. 2001.

[25] A.B. Watson, "Measurement of JND Scales for Digital Video Sequences," http://vision.arc.nasa.gov/jnd/JNDPreliminaryReport.pdf Jan, 2005

[26] N. Jayant, J. Johnston, R. Safranek, "Signal compression based on models of human perception", *Proceedings of the IEEE*, Volume 81, Issue 10, Oct. 1993 Page(s): 1385 – 1422

[27] Wolf et al., "Perception-based video quality measurement system," United States Patent 5446492

[28] C.C. Koh, S.K. Mitra, J.M. Foley, and I.E.J. Heynderickx, "Annoyance of individual artifacts in MPEG-2 compressed video and their relation to overall annoyance", *Proceedings of the SPIE, Vol. 5666, The International Society for Optical Engineering*, Santa Jose, pp. 595-606, Jan. 2005

[29] MSU Subjective Comparison of Modern Video Codecs http://www.compression.ru/video/codec_comparison/subjective_codecs_comparison_en.html , Jan. 2007

[30] M.C.Q. Farias, Mitra, S.K., "No-reference video quality metric based on artifact measurements," *IEEE International Conference on Image Processing, Vol. 3,* 141-144: 2005

[31] K. T. Tan and Mohammed Ghanbari, "Blockiness Detection for MPEG2-Coded Video," *IEEE Signal Processing Letters, 7(8)*, August 2000

[32] Tan, K.T., Ghanbari, M., "A combinational automated MPEG video quality assessment model," *International Conference on Image Processing and its Applications, Vol. 1,* 188-192: 1999

[33] Snell & Wilcox homepage: http://www.snellwilcox.com.

[34] Mike Knee "A Single-ended Picture Quality Measure for MPEG-2", http://www.broadcastpapers.com/whitepapers/SnellWilcoxQualityMeasure_101.pdf , Jan. 2006

[35] Knee, M.J. Diggins J. Improvements in data compression International Patent Application WO 00/22834. World Intellectual Property Bureau, 20 Apr. 2000

[36] G.A. Triantafyllidis, D. Tzovaras, M.G. Strintzis, "Blockiness detection in compressed data," *IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3*, 1533-1536, 2001

[37] Turaga et al., "Method and system for estimating no-reference objective quality of video data," United States Patent 7092448

[38] Chen et al., "System for extracting coding parameters from coded data," United States Patent 6101278

[39] J. E. Caviedes, "Method and apparatus for measuring the quality of video data," United States Patent 7038710, http://www.freepatentsonline.com/7038710.html Aug. 2006

[40] 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, Apr. 2004

[41] Ligang Lu Zhou Wang Bovik, A.C. Kouloheris, J. , "Full-reference video quality assessment considering structural distortion and no-reference quality evaluation of MPEG video," *International Conference on Multimedia and Expo, Vol. 1,* 61-64: 2002

[42] Z. Wang et al., "Blind measurement of blocking artifacts in images," in *Proc. IEEE ICIP*, 2000, (3), pp 981–984.

[43] Reibman, A.R. Vaishampayan, V., "Low complexity quality monitoring of MPEG-2 video in a network,"*, 2003. International Conference on Image Processing, Vol. 3:* 261-264, 2003

[44] Reibman, A.R.; Vaishampayan, V.A.; Sermadevi, Y.; "Quality monitoring of video over a packet network," *IEEE Transactions on Multimedia, 6(2),* 327-334, 2004

[45] Reibman, A.R.; Kanumuri, S.; Vaishampayan, V.; Cosman, P.C.; "Visibility of individual packet losses in MPEG-2 video, " *International Conference on Image Processing, Vol. 1,* 171-174, 2004

[46] Kanumuri, S.; Cosman, P.C.; Reibman, A.R.; Vaishampayan, V.A.; "Modeling packet-loss visibility in MPEG-2 video," *IEEE Transactions on Multimedia, 8(2),* 341-355, 2006

[47] Leontaris, A. Reibman, A.R., "Comparison of blocking and blurring metrics for video compression," *IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2*, 585-588, 2005

[48] www.arehna.di.uoa.gr/Eusipco2005/defevent/papers/cr1042.pdf , Feb. 2006

[49] F. Yang, S. Wan, Y. Chang, and H.R. Wu, "A Novel Objective No-Reference Metric for Digital Video Quality Assessment", *IEEE Signal processing letters*, Vol. 12, No. 10, Oct. 2005

[50] Pan, F. Lin, X. Rahardja, S. Ong, E.P. Lin, W.S., "Measuring blocking artifacts using edge direction information," *International Conference on Multimedia and Expo, Vol. 2,* 1491-1494: 2004

[51] Shengke Qiu; Huaxia Rui; Le Zhang; "No-reference Perceptual Quality Assessment for Streaming Video Based on Simple End-to-end Network Measures," *International Conference on Networking and Services,* July 2006

[52] Ee Ping Ong Weisi Lin Zhongkang Lu Susu Yao Mei Hwan Loke, "Perceptual Quality Metric for H.264 Low Bit Rate Videos," *International Conference on Multimedia and Expo,* 677-680: 2006

[53] A. Masry, S. Hemami, "CVQE: A Metric for Continuous Video Quality Evaluation at Low Bit Rates," http://foulard.ee.cornell.edu/publications/masry_hvei03.pdf , Jan. 2006

[54] A. B. Watson, J. Hu, and John F. McGowan III, "Digital video quality metric based on human vision," *Journal of Electronic Imaging, 10(1),* 20-29, 2001

[55] http://www.tek.com/site/ps/0,,25-11735-INTRO_EN,00.html, Jan. 2006

[56] Pixelmetrix Corporation homepage, http://www.pixelmetrix.com.

[57] "A Guide to Picture Quality Measurements for Modern Television Systems", Tektronix Inc, Beaverton OR, 1997 http://www.tektronix.com/Measurement/App_Notes/PicQuality/25W_11419_0.pdf, Jan. 2006

[58] http://www.tek.com/site/ps/0,,25-13522-INTRO_EN,00.html , Jan. 2006

[59] J. Lauterjung, "Picture Quality Measurement," Rohde & Schwarz GmbH & Co KG http://www.rohde-schwarz.com/www/downcent.nsf/file/PQM.pdf/$file/PQM.pdf , Jan. 2006

[60] Rhode & Schwarz homepage, http://www.rohde-schwarz.com , Jan. 2006

[61] J. Lauterjung, "First results of digital video quality measurements in DVB networks," Rohde & Schwarz www.rohde-schwarz.com/www/downcent.nsf/file/DVQ_results.pdf/$file/DVQ_results.pdf , Jan. 2006

[62] IneoQuest: "Media Delivery Index" http://ftp.ineoquest.com/pub/docs/Datasheets/Media_Delivery_Index.pdf, Jan. 2006

[63] Spirent Video Test System (VTS): IPTV Real-Time Video Quality Testing http://www.spirentcom.com/documents/3942.pdf , Jan. 2006

[64] Spirent Communications; white paper: "MPQM vs. Media Delivery Index: Toward a comparison framework for delivered video quality metrics," http://www.spirentcom.com/documents/4001.pdf , Jan. 2006

[65] Wolf, S., "Measuring the end-to-end performance of digital video systems," *IEEE Tran. on Broadcasting, 43(3)*, 320-328, 1997

[66] S. Tao, John Apostolopoulos, R. Guerin, "Real-Time Monitoring of Video Quality in IP Networks," *Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 2005)*, pages 129-134: 2005

www.hpl.hp.com/personal/John_Apostolopoulos/papers/nossdav05_final.pdf , Jan. 2006

[67] Z. Wang, S. Banerjee, S. Jamin, "Studying streaming video quality: from an application point of view," *Proceedings of the eleventh ACM international conference on Multimedia,* 327-330: 2003

[68] H. R. Sheikh, Z. Wang, L. Cormack and A. C. Bovik, "LIVE Image Quality Assessment Database", http://live.ece.utexas.edu/research/quality , Jan. 2006

[69] E.G. Technologies Inc. http://www.egtinc.com/ , May 2004

[70] MPEG-2 reference software: http://www.mpeg.org/MSSG/ , May 2004

[71] R Shu Tao, John Apostolopoulos, Roch Guerin, "Real-time monitoring of video quality in IP networks," in *Proc. of the international workshop on Network and operating systems support for digital audio and video*, 129–134, 2005

[72] C.J. van den Branden Lambrecht, "A working spatio-temporal model of the human visual system forimage restoration and quality assessment applications," *IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 4*, 2291-2294: 1996

[73] X. Yang, W. Lin, Z. Lu, E. Ong, S. Yao, "Motion-compensated residue preprocessing in video coding based on just-noticeable-distortion profile," *IEEE Tran. on Circuits and Systems for Video Technology, 15(6),* 742-752: 2005

[74] A. Leontaris, P.C. Cosman, A.R. Reibman, "Quality Evaluation of Motion-Compensated Edge Artifacts in Compressed Video," *IEEE Transactions on Image Processing, 16(4),* 943-956, 2007

[75] X. Marichal, W.-Y. Ma, and H. J. Zhang, "Blur determination in the compressed domain using DCT information," in *Proc. IEEE Int. Conf. Image Processing*, Oct. 1999, pp. 386–390

[76] H.264/ AVC reference software: http://iphome.hhi.de/suehring/tml/ , May 2005

[77] Windows Media Video 9 Series Codecs:
http://www.microsoft.com/windows/windowsmedia/9series/codecs/video.aspx , May 2005

[78] W. Gao; Mermer, C.; Y. Kim; "A de-blocking algorithm and a blockiness metric for highly compressed images", *Circuits and Systems for Video Technology, IEEE Transactions on, 12(12),* 1150 – 1159, 2002

[79] Nemethova, O. Ries, M. Siffel, E. Rupp, M., "Subjective evaluation of video quality for H.264 encoded sequences**," (***Mobile Future) The Symposium on Trends in Communications,* 191-194, Oct. 2004

[80] http://www.cost290.org/td2004/tds/td04005.pdf , Jan. 2007

[81] VLC media player: http://www.videolan.org/vlc/ , Aug. 2004

VITA

Nitin Suresh was born in the town of Tirupathi in the state of Andhra Pradesh, India. He grew up in the cities of New Delhi and Madras. He received the Bachelor of Technology degree in Electrical Engineering from the Indian Institute of Technology, Madras, India, in 2001 and the Masters degree in Electrical and Computer Engineering from the Georgia Institute of Technology, Atlanta, in May, 2003, where he is now working towards the Ph.D. degree. He has industrial experience in the quality, coding and architectural aspects of audio and video engineering. His current research interests are in Signal Processing with emphasis on Image and Video Processing algorithms and Quality Evaluation techniques.