RECOMMENDATION ITU-R BT.1683

Objective perceptual video quality measurement techniques for standard definition digital broadcast television in the presence of a full reference

(Question ITU-R 44/6)

(2004)

The ITU Radiocommunication Assembly,

considering

a) that the ability to measure automatically the quality of broadcast video has long been recognized as a valuable asset to the industry;

b) that conventional objective methods are no longer fully adequate for measuring the perceived video quality of digital video systems using compression;

c) that objective measurement of perceived video quality will complement conventional objective test methods;

d) that current formal subjective assessment methods are time-consuming and expensive and generally not suited for operational conditions;

e) that objective measurement of perceived video quality may usefully complement subjective assessment methods,

recommends

1 that the guidelines, scope and limitations given in Annex 1 be used in the application of the objective video quality models found in Annexes 2-5;

2 that the objective video quality models given in Annexes 2-5 be used for objective measurement of perceived video quality.

Annex 1

Summary

This Recommendation specifies methods for estimating the perceived video quality of a one-way video transmission system. This Recommendation applies to baseband signals. The estimation methods in this Recommendation are applicable to:

– codec evaluation, specification, and acceptance testing;

– potentially real-time, in-service quality monitoring at the source;

– remote destination quality monitoring when a copy of the source is available;

– quality measurement of a storage or transmission system that utilizes video compression and decompression techniques, either a single pass or a concatenation of such techniques.

Introduction

The ability to measure automatically the quality of broadcast video has long been recognized as a valuable asset to the industry. The broadcast industry requires such tools to replace or supplement costly and time-consuming subjective quality testing. Traditionally, objective quality measurement has been obtained by calculating peak signal-to-noise ratios (PSNRs). Although a useful indicator

of quality, PSNR has been shown to be a less than satisfactory representation of perceptual quality. To overcome the limitations associated with PSNR, research has been directed towards defining algorithms that can measure the perceptual quality of broadcast video. Such objective perceptual quality measurement tools may be applied to testing the performance of a broadcast network, as equipment procurement aids and in the development of new broadcast video coding techniques. In recent years, significant work has been dedicated to the development of reliable and accurate tools that can be used to objectively measure the perceptual quality of broadcast video. This Recommendation defines objective computational models that have been shown to be superior to PSNR as automatic measurement tools for assessing the quality of broadcast video. The models were tested on 525-line and 625-line material conforming to Recommendation ITU-R BT.601, which was characteristic of secondary distribution of digitally encoded television quality video.

The performance of the perceptual quality models was assessed through two parallel evaluations of the test video material[1]. In the first evaluation, a standard subjective method, the double stimulus continuous quality scale (DSCQS) method, was used to obtain subjective ratings of quality of video material by panels of human observers (Recommentdation ITU-R BT.500 – Methodology for the subjective assessment of the quality of television pictures). In the second evaluation, objective ratings were obtained by the objective computational models. For each model, several metrics were computed to measure the accuracy and consistency with which the objective ratings predicted the subjective ratings. Three independent laboratories conducted the subjective evaluation portion of the test. Two laboratories, Communications Research Center (CRC, Canada) and Verizon (United States of America), performed the test with 525/60 Hz sequences and a third lab, Fondazione Ugo Bordoni (FUB, Italy), performed the test with 625/50 Hz sequences. Several laboratories "proponents" produced objective computational models of the video quality of the same video sequences tested with human observers by CRC, Verizon and FUB. The results of the tests are given in Appendix 1.

This Recommendation includes the objective computational models shown in Table 1.

TABLE 1

Model number	Name	Video Quality Experts Group (VQEG) proponent	Country	Annex
1	British Telecom	D	United Kingdom	2
2	Yonsei University/Radio Research Laboratory/SK Telecom	E	Korea (Rep. of)	3
3	Center for Telecommunications Research and Development (CPqD)	F	Brazil	4
4	National Telecommunications and Information Administration/Institute for Telecommunication Science (NTIA/ITS)	H	United States of America	5

[1] ITU-R Doc. 6Q/14 [September, 2003] Final Report from the Video Quality Experts Group on the Validation of Objective Models of Video Quality Assessment, Phase II (FR-TV2).

A complete description of the above four objective computational models is provided in Annexes 2-5.

Existing video quality test equipment can be used until new test equipment implementing any of the four above models is readily available.

For any model to be considered for inclusion in the normative section of this Recommendation in the future, the model must be verified by an open independent body (such as VQEG) which will do the technical evaluation within the guidelines and performance criteria set out by Radiocommunication Study Group 6. The intention of Radiocommunication Study Group 6 is to eventually recommend only one normative full reference method.

1 Scope

This Recommendation specifies methods for estimating the perceived video quality of a one-way video system. This Recommendation applies to baseband signals. The objective video performance estimators are defined for the end-to-end quality between the two points. The estimation methods are based on processing 8-bit digital component video as defined by Recommendation ITU-R BT.601[2]. The encoder can utilize various compression methods (e.g. Moving Picture Experts Group (MPEG), ITU-T Recommendation H.263, etc.). The models proposed in this Recommendation may be used to evaluate a codec (encoder/decoder combination) or a concatenation of various compression methods and memory storage devices. While the derivation of the objective quality estimators described in this Recommendation might have considered error impairments (e.g. bit errors, dropped packets), independent testing results are not currently available to validate the use of the estimators for systems with error impairments. The validation test material did not contain channel errors.

1.1 Application

This Recommendation provides video quality estimations for television video classes (TV0-TV3), and multimedia video class (MM4) as defined in ITU-T Recommendation P.911, Annex B. The applications for the estimation models described in this Recommendation include but are not limited to:

– codec evaluation, specification, and acceptance testing, consistent with the limited accuracy as described below;

– potentially real-time, in-service quality monitoring at the source;

– remote destination quality monitoring when a copy of the source is available;

– quality measurement of a storage or transmission system that utilizes video compression and decompression techniques, either a single pass or a concatenation of such techniques.

1.2 Limitations

The estimation models described in this Recommendation cannot be used to replace subjective testing. Correlation values between two carefully designed and executed subjective tests (i.e. in two different laboratories) normally fall within the range 0.92 to 0.97. This Recommendation does not

---

[2] This does not preclude implementation of the measurement method for one-way video systems that utilize composite video input and outputs. Specification of the conversion between composite and component domains is not part of this Recommendation. For example, SMPTE 170M Standard specifies one method for performing this conversion for NTSC.

supply a means for quantifying potential estimation errors. Users of this Recommendation should review the comparison of available subjective and objective results to gain an understanding of the range of video quality rating estimation errors.

The predicted performance of the estimation models is not currently validated for video systems with transmission channel error impairments.

Annex 2

Model 1

CONTENTS

1 Introduction

The BT full-reference (BTFR) automatic video quality assessment tool produces predictions of video quality that are representative of human quality judgements. This objective measurement tool digitally simulates features of the human visual system (HVS) to give accurate predictions of video quality and offers a viable alternative to costly and time-consuming formal subjective assessments.

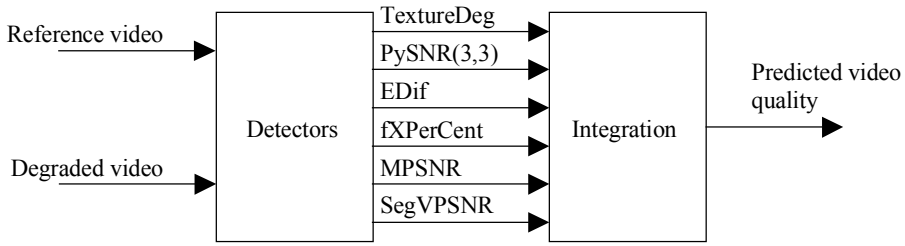
A software implementation of the model was entered in the VQEG2 tests and the resulting performance presented in a test report.

2 BTFR

The BTFR algorithm consists of detection followed by integration as shown in Fig. 1. Detection involves the calculation of a set of perceptually meaningful detector parameters from the undistorted (reference) and distorted (degraded) video sequences. These parameters are then input to the integrator, which produces an estimate of the perceived video quality by appropriate weighting. The choice of detectors and weighting factors are founded on knowledge of the spatial and temporal masking properties of the HVS and determined through calibration experiments.

FIGURE 1

Full-reference video quality assessment model



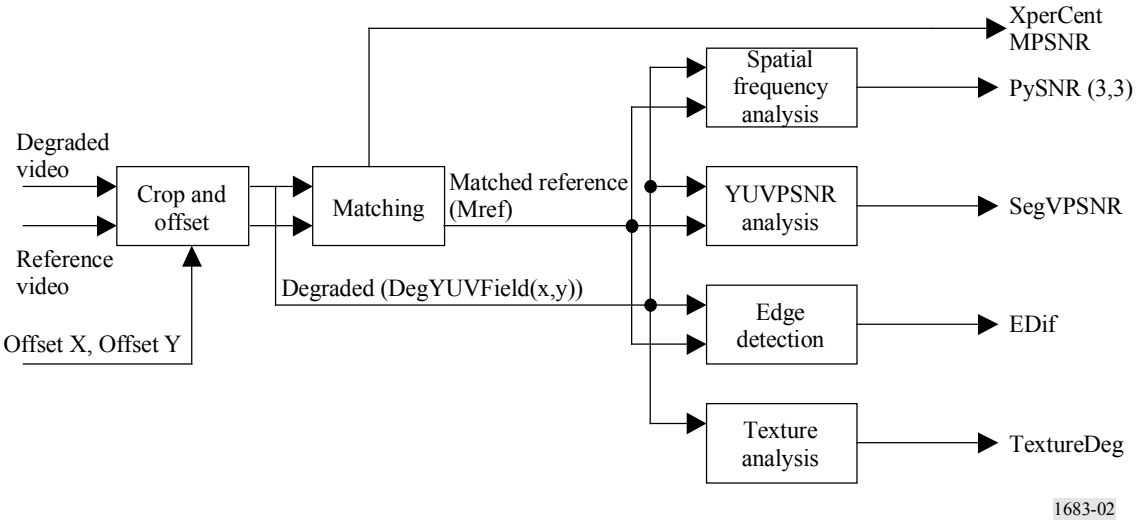
Input video of types 625 (720 × 576) interlaced at 50 fields/s and 525 (720 × 486) interlaced at 59.94 fields/s in *YUV*422 format are supported by the model.

3 Detectors

The detection module of the BTFR algorithm calculates a number spatial, temporal and frequency-based measures from the input *YUV* formatted sequences, as shown in Fig. 2.

FIGURE 2

Detection



3.1 Input conversion

First, the input sequences are converted from *YUV*422 interlaced format to a block *YUV*444 deinterlaced format so that each successive field is represented by arrays *RefY*, *RefU* and *RefV*:

$$RefY(x, y) \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{1}$$

$$RefU(x, y) \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{2}$$

$$RefV(x, y) \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{3}$$

where:

X: number of horizontal pixels within a field

Y: number of vertical pixels.

For a *YUV*422 input, each *U* and *V* value must be repeated to give the full resolution arrays (2) and (3).

3.2 Crop and offset

This routine crops with offset the degraded input sequence and crops without offset the reference input sequence. The offset parameters *XOffset* and *YOffset* are determined externally and define the number of pixels horizontal and vertical that the degraded sequence is offset from the reference. The picture origin is defined as being in the top left hand corner of the image, with a positive horizontal increment moving right and a positive vertical increment moving down the picture. A value of *XOffset* = 2 indicates that the degraded fields are offset to the right by 2 pixels and a value of *YOffset* = 2 indicates an offset down of 2 pixels. For an input field with *YUV* values stored in *YUV*444 format (see § 3.1) in arrays *InYField*, *InUField*, and *InVField* the cropped and offset output is calculated according to (4) to (20).

$$XStart = -XOffset \tag{4}$$

$$\text{if } (XStart < C_x) \qquad \text{then } XStart = C_x \tag{5}$$

$$XEnd = X - 1 - XOffset \tag{6}$$

$$\text{if } (XEnd > X - C_x - 1) \quad \text{then } XEnd = X - C_x - 1 \tag{7}$$

$$YStart = -YOffset \tag{8}$$

$$\text{if } (YStart < C_y) \quad \text{then } YStart = C_y \tag{9}$$

$$YEnd = Y - 1 - YOffset \tag{10}$$

$$\text{if } (YEnd > Y - C_y - 1) \quad \text{then } YEnd = Y - C_y - 1 \tag{11}$$

X and Y give the horizontal and vertical field dimensions respectively and C_x and C_y the number of pixels to be cropped from left and right and top and bottom.

For 625 sequences,

$$X = 720, \quad Y = 288, \quad C_x = 30, \quad C_y = 10 \tag{12}$$

For 525 sequences,

$$X = 720, \quad Y = 243, \quad C_x = 30, \quad C_y = 10 \tag{13}$$

Xstart, *Xend*, *Ystart* and *Yend* now define the region of each field that will be copied. Pixels outside this region are initialized according to equations (14) to (15), where *YField*, *UField* and *VField* are *XxY* output pixel arrays containing *Y*, *U* and *V* values respectively.

The vertical bars to the left and right of the field are initialized according to:

$$YField(x, y) = 0 \qquad x = 0...XStart - 1, XEnd + 1...X - 1 \qquad y = 0...Y - 1 \tag{14}$$

$$UField(x, y) = VField(x, y) = 128 \qquad x = 0...XStart - 1, XEnd + 1...X - 1 \qquad y = 0...Y - 1 \tag{15}$$

The horizontal bars at the top and bottom of the field are initialized according to:

$$YField\ (x, y) = 0 \qquad x = XStart\ ...XEnd, \qquad y = 0...YStart\ - 1, YEnd\ + 1...Y - 1 \tag{16}$$

$$UField(x, y) = VField(x, y) = 128 \qquad x = XStart...XEnd \qquad y = 0...YStart - 1, YEnd + 1...Y - 1 \tag{17}$$

Finally, the pixel values are copied according to:

$$YField(x, y) = InYField(x + XOffset, y + YOffset) \qquad x = XStart...XEnd \qquad y = YStart...YEnd \tag{18}$$

$$UField(x, y) = InUField(x + XOffset, y + YOffset) \qquad x = XStart...XEnd \qquad y = YStart...YEnd \tag{19}$$

$$VField(x, y) = InVField(x + XOffset, y + YOffset) \qquad x = XStart...XEnd \qquad y = YStart...YEnd \tag{20}$$

For the degraded input, cropping and shifting produces output field arrays *DegYField*, *DegUField* and *DegVField*, whilst cropping without shifting for the reference sequence produces *RefYField*, *RefUField* and *RefVfield*. These *XxY* two-dimensional arrays are used as inputs to detection routines described below.

3.3 Matching

The matching process produces signals for use within other detection procedures and also detection parameters for use in the integration procedure. The matching signals are generated from a process of finding the best match for small blocks within each degraded field from a buffer of neighbouring reference fields. This process yields a sequence, the matched reference, for use in place of the reference sequence in some of the detection modules.

The matching analysis is performed on 9 × 9 pixel blocks of the intensity arrays *RefYField* and *DegYField*. Adding a field number dimension to the intensity arrays, pixel (*Px*, *Py*) of the reference field *N* can be represented as:

$$Ref(N,Px,Py) = RefYField(Px,Py) \qquad \text{from field } N \tag{21}$$

A 9 × 9 pixel block with centre pixel (*Px*,*Py*) within the *N*-th field can be represented as:

$$BlockRef(N,Px,Py) = Ref(n,x,y) \qquad x = Px-4...Px+4, \qquad y = Py-4...Py+4 \tag{22}$$

Deg(*n, x, y*) and *BlockDeg*(*n, x, y*) can be similarly defined.

For *BlockDeg*(*N, Px, Py*), a minimum matching error, *E*(*N, Px, Py*), is calculated by searching neighbouring reference fields according to:

$$\begin{aligned} E(N,Px,Py) = \text{Min}((1/81)\sum_{j=-4}^{4}\sum_{k=-4}^{4}(&\text{Deg}(N,Px+j,Py+k) \\ &- Ref(n,x+j,y+k))^2) \end{aligned} \tag{23}$$

$$\begin{aligned} n &= N-4,\ldots,N+5 \\ x &= Px-4,...,Px,...,Px+4 \\ y &= Py-4,...,Py,...,Py+4 \end{aligned}$$

where *N* is the index of the degraded field containing the degraded block that is being matched.

If equation (23) determines that the best match to *BlockDeg*(*N, Px, Py*) is *BlockRef*(n_m, x_m, y_m) then a matched reference array *MRef* is updated according to:

$$MRef(N,Px+j,Py+k) = Ref(n_m,x_m+j,y_m+k) \qquad j=-4...4, k=-4...4 \tag{24}$$

The matching process of first searching for the best match for a degraded block followed by the copying of the resulting block into the matched reference array is repeated for the whole of the desired analysis region. This analysis region is defined by block centre points *Px*() and *Py*() according to:

$$Px(h) = 16+8\times h \qquad h = 0...Qx-1 \tag{25}$$

and

$$Py(v) = 16+8\times v \qquad v = 0...Qy-1 \tag{26}$$

where *Qx* and *Qy* define the number of horizontal and vertical analysis blocks.

The matching analysis of the *N*-th field therefore produces a matched reference sequence described by:

$$BlockMRef\ (N,Px(h),Py\ (v)) \qquad h = 0...Qx-1, \qquad v = 0...Qy-1 \tag{27}$$

and a set of best match error values:

$$E(N,Px(h),Py(v)) \qquad h = 0...Qx-1, \qquad v = 0...Qy-1 \tag{28}$$

A set of offset arrays *MatT*, *MatX* and *MatY* can be defined such that:

$$\begin{aligned} BlockMRef(N,Px(h),Py(v))) &= BlockRef(MatT(h,v),MatX(h,v),MatY(h,v)) \\ h &= 0...Qx-1, \qquad v = 0...Qy-1 \end{aligned} \tag{29}$$

The matching parameters for 625 and 525 broadcast sequences are given in Table 2.

TABLE 2

Search parameters for matching procedure

Parameter	625	525
Qx	87	87
Qy	33	28

The analysis region defined by equations (26) and (27) does not cover the complete field size. *MRef* must therefore be initialized according to equation (29) so that it may be used elsewhere unrestricted.

$$MRef(x,y) = 0 \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{30}$$

3.3.1 Matching statistics

Horizontal matching statistics from the matching process are calculated for use in the integration process. The best match for each analysis block, determined according to equation (23), is used in the construction of the histogram *histX* for each field according to:

$$\begin{gathered} histX(MatX(h,v) - Px(h) + 4 = histX(MatX(h,v) - Px(h) + 4) + 1 \\ h = 0...Qx-1, \qquad v = 0...Qx-1 \end{gathered} \tag{31}$$

where array *histX* is initialized to zero for each field. The histogram is then used to determine the measure *fXPerCent* according to:

$$fXPerCent = 100 \times \mathrm{Max}(histX(i)) / \sum_{j=0}^{8} histX(j) \qquad i = 0...8 \tag{32}$$

For each field, the *fXPerCent* measure gives the proportion (%) of matched blocks that contribute to the peak of the matching histogram.

3.3.2 MPSNR

The minimum error, *E*(), for each matched block is used to calculate a matched SNR according to:

$$\begin{gathered} \text{if} \left(\sum_{h=0}^{Qx-1} \sum_{v=0}^{Qy-1} E(N, Px(h), Py(v)) \right) > 0 \qquad \text{then} \\ MPSNR = 10 \log_{10} \left(Qx \times Qy \times 255^2 \Big/ \sum_{h=0}^{Qx-1} \sum_{v=0}^{Qy-1} E(N, Px(h), Py(v)) \right) \end{gathered} \tag{33}$$

$$\text{if} \left(\sum_{h=0}^{Qx-1} \sum_{v=0}^{Qy-1} E(N, Px(h), \ Py(v)) \right) = 0 \qquad \text{then} \quad MPSNR = 10 \log_{10}(255^2) \tag{34}$$

3.3.3 Matching vectors

Horizontal, vertical and delay vectors are stored for later use according to:

$$SyncT(h,v) = MatT(h,v) - N \qquad h = 0...Qx-1, \qquad v = 0...Qy-1 \tag{35}$$

$$SyncX(h,v) = MatX(h,v) - Px(h) \qquad h = 0...Qx-1, \qquad v = 0...Qy-1 \tag{36}$$

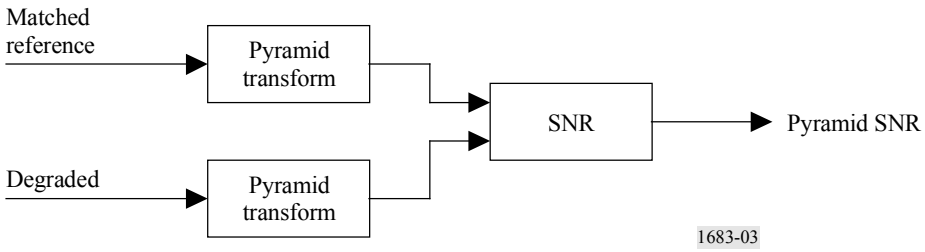
$$SyncY(h,v) = MatY(h,v) - Py(h) \qquad h = 0...Qx-1, \qquad v = 0...Qy-1 \tag{37}$$

3.4 Spatial frequency analysis

The spatial frequency detector is based on a "pyramid" transformation of the degraded and matched reference sequences. First each sequence is transformed to give reference and degraded pyramid arrays. Then, differences between the pyramid arrays are calculated using a mean squared error measure and the results output as a pyramid SNR.

FIGURE 3

Spatial frequency analysis



3.4.1 Pyramid transform

Firstly, the input field, *F*, is copied into a pyramid array, *P*, according to:

$$P(x,y) = F(x,y) \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{38}$$

This pyramid array is then updated by three stages (*stage* = 0..2) of horizontal and vertical analysis. The horizontal analysis *Hpy*(*stage*) is defined by equations (39) to (43).

First a temporary copy is made of the whole pyramid array:

$$PTemp(x,y) = P(x,y) \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{39}$$

Then *x* and *y* limits are calculated according to:

$$Tx = X/2^{(stage+1)} \tag{40}$$

$$Ty = Y/2^{stage} \tag{41}$$

Averages and differences of horizontal pairs of elements of the temporary array are then used to update the pyramid array according to:

$$P(x,y) = 0.5\ \ (PTemp(2x,y) + PTemp(2x+1,y)) \qquad x = 0...Tx-1, \qquad y = 0...Ty-1 \tag{42}$$

$$P(x+Tx,y) = PTemp(2x,y) - PTemp(2x+1,y) \qquad x = 0...Tx-1 \qquad y = 0...Ty-1 \tag{43}$$

The vertical analysis *Vpy*(*stage*) is defined by equations (44) to (48).

$$PTemp(x,y) = P(x,y) \qquad x = 0...X-1, \qquad y = 0...Y-1 \tag{44}$$

$$Tx = X/2^{stage} \tag{45}$$

$$Ty = Y/2^{(stage+1)} \tag{46}$$

Averages and differences of vertical pairs of elements of the temporary array are then used to update the pyramid array according to:

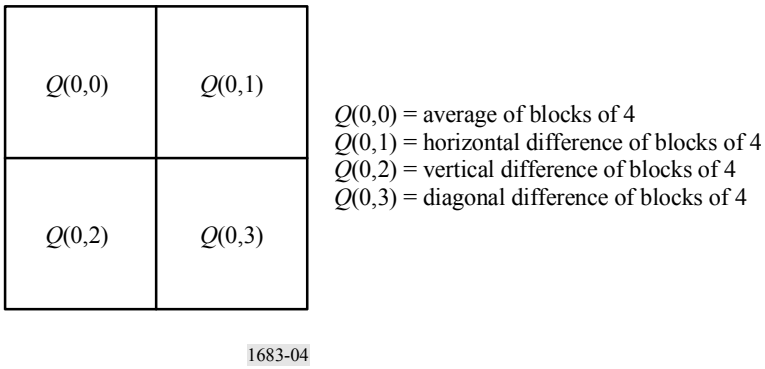
$$P(x,y) = 0.5\ \ (PTemp(x,2y) + PTemp(x,2y+1)) \qquad x = 0...Tx-1, \qquad y = 0...Ty-1 \qquad (47)$$

$$P(x,y+Ty) = PTemp(x,2y) - PTemp(x,2y+1) \qquad x = 0...Tx-1 \qquad y = 0...Ty-1 \qquad (48)$$

For stage 0, the horizontal analysis *Hpy*(0) followed by the vertical analysis *Vpy*(0) updates the whole of the pyramid array with the 4 quadrants *Q*(*stage*, 0...3) constructed according to:

FIGURE 4

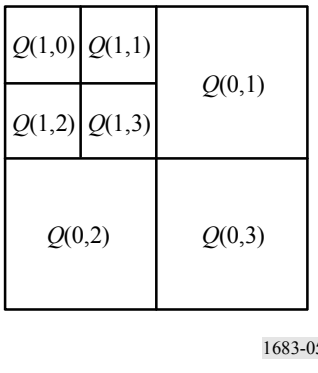
Quadrant output from stage 0 analysis



Stage 1 analysis is then performed on *Q*(0,0) to give results *Q*(1,0...3) that are stored in the pyramid according to:

FIGURE 5

Quadrant output from stage 1 analysis



Stage 2 analysis processes *Q*(1,0) and overwrites it with *Q*(2,0...3).

After the three stages of analysis, the resulting pyramid array has a total of 10 blocks of results. Three blocks *Q*(0,1...3) are from the stage $0,2 \times 2$ pixel analysis, three *Q*(1,1...3) from the stage 1, 4×4 analysis and 4 *Q*(2,0...3) from the stage 2, 8×8 analysis.

The three-stage analysis of the matched reference and degraded sequences produce the pyramid arrays *Pref* and *Pdeg*. Differences between these arrays are then measured in the pyramid SNR module.

3.4.2 Pyramid SNR

A squared error measure between the reference and degraded pyramid arrays is determined over quadrants 1 to 3 of stages 0 to 2 according to:

$$E(s,q) = (1/XY^2) \sum_{x=x1(s,q)}^{x2(s,q)-1} \sum_{y=y1(s,q)}^{y2(s,q)-1} (Pref(x,y) - Pdeg(x,y))^2 \quad s = 0...2 \quad q = 1...3 \tag{49}$$

where $x1$, $x2$, $y1$ and $y2$ define the horizontal and vertical limits of the quadrants within the pyramid arrays and are calculated according to:

$$x1(s,1) = X/2^{(s+1)} \qquad x2(s,1) = 2 \times x1(s,1) \qquad y1(s,1) = 0 \qquad y2(s,1) = Y/2^{(s+1)} \tag{50}$$

$$x1(s,2) = 0 \qquad x2(s,2) = X/2^{(s+1)} \qquad y1(s,2) = Y/2^{(s+1)} \qquad y2(s,2) = 2 \times y1(s,2) \tag{51}$$

$$x1(s,3) = X/2^{(s+1)} \qquad x2(s,3) = 2 \times x1(s,3) \qquad y1(s,3) = Y/2^{(s+1)} \qquad y2(s,3) = 2 \times y1(s,3) \tag{52}$$

The results from equation (49) are then used to determine a PSNR measure for each quadrant of each field according to:

$$\begin{aligned} \text{if} \quad (E > 0,0) \qquad & PySNR(s,q) = 10 \log_{10}(255^2/E(s,q)) \\ \text{then} \qquad & SNR = 10 \log_{10}(255^2 \times XY^2) \end{aligned} \tag{53}$$

where the number of stages $s = 0...2$ and the number of quadrants for each stage $q = 1...3$.

3.5 Texture analysis

The texture of the degraded sequence is measured by recording the number of turning-points in the intensity signal along horizontal picture lines. This may be calculated according to equations (54) to (59).

For each field, first a turning-point counter is initialized according to equation (54).

$$sum = 0 \tag{54}$$

Then, each line, $y = 0...Y - 1$, is processed for $x = 0...X - 2$ according:

$$last\_pos = 0, \qquad last\_neg = 0 \tag{55}$$

$$dif(x) = P(x,y) - P(x+1,y) \tag{56}$$

$$\text{if} \quad ((dif(x) < 0)\ AND\ (last\_neg < last\_pos))\ sum = sum + 1 \tag{57}$$

$$\text{if} \quad ((dif(x) > 0)\ AND\ (last\_neg > last\_pos))\ sum = sum + 1 \tag{58}$$

$$\text{if}\ (dif(x) > 0)\ last\_pos = x \tag{59}$$

$$\text{if}\ (dif(x) < 0)\ last\_neg = x \tag{60}$$

When all the lines for a field have been processed, the counter, *sum*, will contain the number of turning-points in the horizontal intensity signal. This is then used to calculate a texture parameter for each field according to:

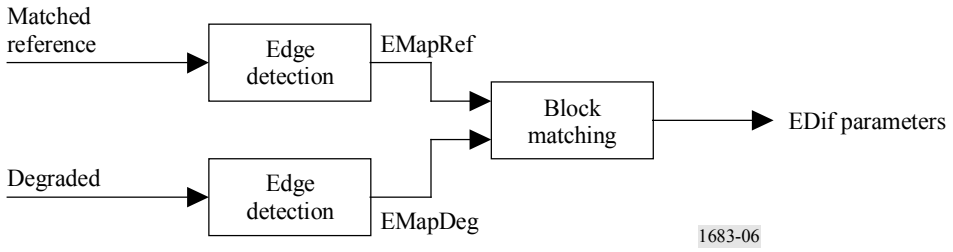
$$TextureDeg = sum \times 100 / XY \tag{61}$$

3.6 Edge analysis

Each field of the degraded and matched reference sequences is separately passed through an edge detection routine to produce corresponding edge field maps, which are then compared in a block matching procedure to produce the detection parameters.

FIGURE 6

Edge analysis



3.6.1 Edge detection

A Canny edge detector [Canny, 1986] was used to determine the edge maps, but other similar edge detection techniques may be used. The resulting edge maps, *EMapRef* and *EMapDeg*, are pixel maps with an edge indicated by a 1 and no edge by 0.

For an edge detected at pixel (*x, y*):

$$EMap(x,y)=1 \qquad x=0...X-1, \qquad y=0...Y-1 \tag{62}$$

For no edge detected at pixel (*x, y*):

$$EMap(x,y)=0 \qquad x=0...X-1, \qquad y=0...Y-1 \tag{63}$$

3.6.2 Edge differencing

The edge differencing procedure measures the differences between the edge maps for corresponding degraded and matched reference fields. The analysis is performed in *NxM* pixel non-overlapping blocks according to equations (64) to (68).

First, a measure of the number of edge-marked pixels in each analysis block is calculated, where *Bh* and *Bv* define the number of non-overlapping blocks to be analysed in the horizontal and vertical directions and *X*1 and *Y*1 define analysis offsets from the field edge.

$$Bref(x,y)=\sum_{i=i1}^{i2}\sum_{j=j1}^{j2} EMapRef(Nx+X1+i, My+Y1+j) \quad x=0...Bh-1, y=0...Bv-1 \tag{64}$$

$$BDeg(x,y)=\sum_{i=i1}^{i2}\sum_{j=j1}^{j2} EMapDeg(Nx+X1+i, My+Y1+j) \quad x=0...Bh-1, y=0...Bv-1 \tag{65}$$

The summation limits are determined according to:

$$i1=-(N \text{ div } 2) \qquad i2=(N-1) \text{ div } 2 \tag{66}$$

$$j1=-(M \text{ div } 2) \qquad j2=(M-1) \text{ div } 2 \tag{67}$$

where the “div” operator represents an integer division.

Then, a measure of the differences over the whole field is calculated according to:

$$EDif = (1/(N\ M\ Bh\ Bv)) \left(\sum_{x=0}^{Bh-1} \sum_{y=0}^{Bv-1} (BRef(x,y) - BDeg(x,y))^{Q} \right)^{1/Q} \tag{68}$$

For 720 × 288 pixel fields for 625 broadcast video:

$$N = 4,\ \ X1 = 6,\ \ Bh = 178 \qquad M = 4,\ \ Y1 = 10,\ \ Bv = 69,\ \ Q = 3 \tag{69}$$

For 720 × 243 pixel fields for 525 broadcast video:

$$N = 4,\ \ X1 = 6,\ \ Bh = 178 \qquad M = 4,\ \ Y1 = 10,\ \ Bv = 58,\ \ Q = 3 \tag{70}$$

3.7 MPSNR analysis

A matched signal to noise ratio is calculated for the pixel *V* values by use of the matching vectors defined in equations (35) to (37). For each set of matching vectors, an error measure, *VE*, is calculated according to:

$$\begin{aligned} VE(h,v) = (1/81) \sum_{i=-4}^{4} \sum_{j=-4}^{4} (DegV(N, Px(h)+i, Py(h)+j) - \\ RefVField(N + SyncT(h,v), Px(h) + SyncX(h,v) + i, Py(v) + SyncY(h,v) + j))^2 \end{aligned} \tag{71}$$

A segmental PSNR measure is then calculated for the field according to:

$$SegVPSNR = (1/Qx\ Qy) \sum_{h=0}^{Qx-1} \sum_{v=0}^{Qy-1} 10 \log_{10}(255^2 / (VE(h,v)+1)) \tag{72}$$

4 Integration

The integration procedure firstly requires the time averaging of the field-by-field detection parameters according to equation (73):

$$AvD(k) = (1/N) \sum_{n=0}^{N-1} D(k,n) \qquad k = 0...5 \tag{73}$$

where:

N: total number of fields in the tested sequences

$D(k, n)$: detection parameter k for field n.

The averaged detection parameters, $AvD(k)$, are then combined to give a predicted quality score, PDMOS, for the N field sequence according to equation (74):

$$PDMOS = Offset + \sum_{k=0}^{5} AvD(k) \times W(k) \tag{74}$$

Tables 3 and 4 show the integrator parameters for 625 and 525 sequences respectively.

TABLE 3

Integration parameters for 625 broadcast video

K	**Parameter name**	***W***
0	*TextureDeg*	–0.68
1	*PySNR*(3,3)	–0.57
2	*EDif*	+58 913.294
3	*fXPerCent*	–0.208
4	*MPSNR*	–0.928
5	*SegVPSNR*	–1.529
Offset	+176.486	
N	400	

TABLE 4

Integration parameters for 525 broadcast video

K	**Parameter name**	***W***
0	*TextureDeg*	+0.043
1	*PySNR*(3,3)	–2.118
2	*EDif*	+60 865.164
3	*fXPerCent*	–0.361
4	*MPSNR*	+1.104
5	*SegVPSNR*	–1.264
Offset	+260.773	
N	480	

5 Registration

The FR model requires both spatial and temporal alignment to operate effectively. The model incorporates inherent alignment and can accommodate spatial offsets between the reference and degraded sequences ±4 pixels and temporal offset of ±4 fields. Spatial and temporal offsets beyond these limits are not handled by the model and a separate registration module will be required to ensure the reference and degraded files are properly aligned.

6 References

CANNY. J. [1986] A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*. Vol. 8(**6**), p. 679-698.

Annex 2a

TABLE 5

525 subjective and objective data

Filename	Source sequence (SRC)	Hypothetical reference circuit (HRC)	Raw mean subjective rating	Model predicted rating based on raw data	Scaled mean subjective rating	Model predicted rating based on scaled data
V2src01_hrc01_525.yuv	1	1	–38.30757576	44.945049	0.5402368	0.69526
V2src01_hrc02_525.yuv	1	2	–39.56212121	38.646271	0.5483205	0.58989
V2src01_hrc03_525.yuv	1	3	–25.9469697	32.855755	0.4024097	0.50419
V2src01_hrc04_525.yuv	1	4	–17.24090909	21.062775	0.3063528	0.36089
V2src02_hrc01_525.yuv	2	1	–35.23636364	31.260744	0.5025558	0.48242
V2src02_hrc02_525.yuv	2	2	–18.01818182	18.732758	0.3113346	0.33715
V2src02_hrc03_525.yuv	2	3	–6.284848485	8.914509	0.1881739	0.25161
V2src02_hrc04_525.yuv	2	4	–6.983333333	4.16663	0.1907347	0.21776
V2src03_hrc01_525.yuv	3	1	–31.96515152	22.348713	0.4682724	0.37461
V2src03_hrc02_525.yuv	3	2	–17.47727273	10.44728	0.3088831	0.26352
V2src03_hrc03_525.yuv	3	3	–1.104545455	2.494911	0.1300389	0.20688
V2src03_hrc04_525.yuv	3	4	–1.171212121	0	0.1293293	0.19158
V2src04_hrc05_525.yuv	4	5	–50.64090909	40.82526	0.6742005	0.6249
V2src04_hrc06_525.yuv	4	6	–28.05454545	32.552322	0.4250873	0.49999
V2src04_hrc07_525.yuv	4	7	–23.87575758	25.286598	0.3762656	0.40764
V2src04_hrc08_525.yuv	4	8	–16.60757576	19.86405	0.2972294	0.3485
V2src05_hrc05_525.yuv	5	5	–31.86969697	30.812616	0.4682559	0.47645
V2src05_hrc06_525.yuv	5	6	–18.56515152	21.413895	0.3203024	0.3646
V2src05_hrc07_525.yuv	5	7	–8.154545455	15.446437	0.2071702	0.306
V2src05_hrc08_525.yuv	5	8	–4.006060606	10.836051	0.1652752	0.26662
V2src06_hrc05_525.yuv	6	5	–41.63181818	37.342789	0.5690291	0.56967
V2src06_hrc06_525.yuv	6	6	–29.48787879	26.660055	0.4370961	0.42391
V2src06_hrc07_525.yuv	6	7	–22.25909091	20.878248	0.3591788	0.35896
V2src06_hrc08_525.yuv	6	8	–12.03181818	16.896168	0.2482169	0.31941
V2src07_hrc05_525.yuv	7	5	–23.89545455	19.086998	0.3796362	0.34067
V2src07_hrc06_525.yuv	7	6	–10.15606061	10.69402	0.2276934	0.26548
V2src07_hrc07_525.yuv	7	7	–4.240909091	4.896546	0.1644409	0.22267
V2src07_hrc08_525.yuv	7	8	–5.98030303	1.555055	0.1819566	0.20099
V2src08_hrc09_525.yuv	8	9	–76.2	52.094177	0.9513387	0.83024
V2src08_hrc10_525.yuv	8	10	–61.34545455	47.395226	0.789748	0.7397
V2src08_hrc11_525.yuv	8	11	–66.02575758	52.457584	0.8405916	0.83753

TABLE 5

525 subjective and objective data

Filename	Source sequence (SRC)	Hypothetical reference circuit (HRC)	Raw mean subjective rating	Model predicted rating based on raw data	Scaled mean subjective rating	Model predicted rating based on scaled data
V2src08_hrc12_525.yuv	8	12	–37.20454545	37.931854	0.5221555	0.57874
V2src08_hrc13_525.yuv	8	13	–31.23030303	30.95985	0.4572049	0.4784
V2src08_hrc14_525.yuv	8	14	–31.26818182	33.293602	0.4614104	0.51031
V2src09_hrc09_525.yuv	9	9	–64.42878788	54.414772	0.8262912	0.87746
V2src09_hrc10_525.yuv	9	10	–49.92878788	36.080425	0.660339	0.55061
V2src09_hrc11_525.yuv	9	11	–53.73181818	46.338791	0.7100111	0.72031
V2src09_hrc12_525.yuv	9	12	–34.36969697	23.21393	0.4921708	0.38409
V2src09_hrc13_525.yuv	9	13	–22.85454545	16.955978	0.3656559	0.31998
V2src09_hrc14_525.yuv	9	14	–16.41666667	13.694396	0.2960957	0.29046
V2src10_hrc09_525.yuv	10	9	–72.11212121	48.179104	0.9084171	0.75433
V2src10_hrc10_525.yuv	10	10	–43.11666667	30.703861	0.5908784	0.475
V2src10_hrc11_525.yuv	10	11	–56.11969697	52.63887	0.7302376	0.84118
V2src10_hrc12_525.yuv	10	12	–19.55909091	21.95225	0.3345703	0.37033
V2src10_hrc13_525.yuv	10	13	–12.34393939	16.23988	0.2565459	0.31328
V2src10_hrc14_525.yuv	10	14	–16.05	23.201355	0.2953144	0.38395
V2src11_hrc09_525.yuv	11	9	–50.40454545	36.394535	0.6675853	0.55531
V2src11_hrc10_525.yuv	11	10	–54.26212121	37.812542	0.7054929	0.5769
V2src11_hrc11_525.yuv	11	11	–41.73636364	44.128036	0.5761193	0.68087
V2src11_hrc12_525.yuv	11	12	–19.03939394	14.619688	0.32761	0.29857
V2src11_hrc13_525.yuv	11	13	–17.72121212	14.12041	0.310495	0.29417
V2src11_hrc14_525.yuv	11	14	–19.4969697	14.927424	0.331051	0.30132
V2src12_hrc09_525.yuv	12	9	–61.35	40.051254	0.7883371	0.61229
V2src12_hrc10_525.yuv	12	10	–46.84545455	31.128973	0.6295301	0.48066
V2src12_hrc11_525.yuv	12	11	–51.80151515	41.77285	0.6809288	0.6406
V2src12_hrc12_525.yuv	12	12	–22.51969697	20.868282	0.3651402	0.35886
V2src12_hrc13_525.yuv	12	13	–14.17878788	15.040992	0.2714356	0.30234
V2src12_hrc14_525.yuv	12	14	–14.6030303	13.521517	0.2782449	0.28896
V2src13_hrc09_525.yuv	13	9	–55.25	38.691498	0.7211194	0.5906
V2src13_hrc10_525.yuv	13	10	–39.55	33.054504	0.5545722	0.50696
V2src13_hrc11_525.yuv	13	11	–40.03939394	45.9454	0.5525494	0.71318
V2src13_hrc12_525.yuv	13	12	–14	16.631002	0.2708744	0.31692
V2src13_hrc13_525.yuv	13	13	–14.33181818	15.113959	0.27549	0.30299
V2src13_hrc14_525.yuv	13	14	–14.31969697	16.611286	0.2733771	0.31674

TABLE 6

625 subjective and objective data

Filename	SRC	HRC	Raw mean subjective rating	Model predicted rating based on raw data	Scaled mean subjective rating	Model predicted rating based on scaled data
V2src1_hrc2_625.yuv	1	2	38.85185185	31.764214	0.59461	0.47326
V2src1_hrc3_625.yuv	1	3	42.07407407	21.868561	0.64436	0.36062
V2src1_hrc4_625.yuv	1	4	23.77777778	12.195552	0.40804	0.27239
V2src1_hrc6_625.yuv	1	6	18.14814815	9.169512	0.34109	0.24887
V2src1_hrc8_625.yuv	1	8	12.92592593	6.738072	0.2677	0.23128
V2src1_hrc10_625.yuv	1	10	11.88888889	2.553883	0.26878	0.20356
V2src2_hrc2_625.yuv	2	2	33.51851852	31.492788	0.54173	0.46985
V2src2_hrc3_625.yuv	2	3	46.48148148	31.1313	0.70995	0.46535
V2src2_hrc4_625.yuv	2	4	13.33333333	20.241726	0.27443	0.34432
V2src2_hrc6_625.yuv	2	6	8.814814815	17.39045	0.22715	0.31721
V2src2_hrc8_625.yuv	2	8	7.074074074	14.914576	0.21133	0.29513
V2src2_hrc10_625.yuv	2	10	3.407407407	7.352309	0.16647	0.23562
V2src3_hrc2_625.yuv	3	2	48.07407407	38.852715	0.73314	0.56845
V2src3_hrc3_625.yuv	3	3	50.66666667	38.244621	0.76167	0.55982
V2src3_hrc4_625.yuv	3	4	32.11111111	27.733229	0.49848	0.42454
V2src3_hrc6_625.yuv	3	6	22.33333333	24.80323	0.38613	0.39159
V2src3_hrc8_625.yuv	3	8	16.33333333	23.296747	0.34574	0.37544
V2src3_hrc10_625.yuv	3	10	11.96296296	16.33028	0.26701	0.30759
V2src4_hrc2_625.yuv	4	2	36.14814815	42.041592	0.58528	0.61514
V2src4_hrc3_625.yuv	4	3	55.03703704	49.283836	0.90446	0.72942
V2src4_hrc4_625.yuv	4	4	39.7037037	38.322186	0.62361	0.56091
V2src4_hrc6_625.yuv	4	6	38.03703704	36.863457	0.61143	0.54053
V2src4_hrc8_625.yuv	4	8	24.40740741	32.46579	0.43329	0.48214
V2src4_hrc10_625.yuv	4	10	12.88888889	25.918123	0.26548	0.40388
V2src5_hrc2_625.yuv	5	2	38.62962963	38.95779	0.61973	0.56995
V2src5_hrc3_625.yuv	5	3	44.18518519	40.076313	0.68987	0.58609
V2src5_hrc4_625.yuv	5	4	24.66666667	23.166002	0.41648	0.37406
V2src5_hrc6_625.yuv	5	6	23.62962963	20.592213	0.4218	0.34778
V2src5_hrc8_625.yuv	5	8	12.40740741	13.763152	0.27543	0.28531
V2src5_hrc10_625.yuv	5	10	7.37037037	8.418313	0.2022	0.24332
V2src6_hrc2_625.yuv	6	2	22.48148148	33.810165	0.38852	0.49949
V2src6_hrc3_625.yuv	6	3	27.07407407	25.004984	0.44457	0.39379
V2src6_hrc4_625.yuv	6	4	13.18518519	20.889347	0.27983	0.35074
V2src6_hrc6_625.yuv	6	6	14.44444444	17.418222	0.28106	0.31747

TABLE 6

625 subjective and objective data

Filename	SRC	HRC	Raw mean subjective rating	Model predicted rating based on raw data	Scaled mean subjective rating	Model predicted rating based on scaled data
V2src6_hrc8_625.yuv	6	8	8.740740741	15.486559	0.23726	0.30011
V2src6_hrc10_625.yuv	6	10	5.518518519	11.509192	0.17793	0.2669
V2src7_hrc4_625.yuv	7	4	39.25925926	45.231079	0.59953	0.66412
V2src7_hrc6_625.yuv	7	6	33.85185185	43.131519	0.55093	0.63163
V2src7_hrc9_625.yuv	7	9	27.07407407	39.506535	0.45163	0.57784
V2src7_hrc10_625.yuv	7	10	19.25925926	34.418381	0.35617	0.50749
V2src8_hrc4_625.yuv	8	4	15.85185185	40.408993	0.32528	0.59095
V2src8_hrc6_625.yuv	8	6	17.03703704	38.552574	0.32727	0.56418
V2src8_hrc9_625.yuv	8	9	14.85185185	35.577034	0.30303	0.52297
V2src8_hrc10_625.yuv	8	10	11.48148148	30.278536	0.26366	0.45484
V2src9_hrc4_625.yuv	9	4	28.96296296	30.515778	0.47656	0.45775
V2src9_hrc6_625.yuv	9	6	30.51851852	26.971027	0.49924	0.41577
V2src9_hrc9_625.yuv	9	9	19.66666667	23.351355	0.39101	0.37601
V2src9_hrc10_625.yuv	9	10	20.92592593	17.856861	0.37122	0.32152
V2src10_hrc4_625.yuv	10	4	40.33333333	43.640377	0.70492	0.63942
V2src10_hrc6_625.yuv	10	6	37.33333333	40.552502	0.58218	0.59305
V2src10_hrc9_625.yuv	10	9	30.92592593	36.747391	0.49711	0.53893
V2src10_hrc10_625.yuv	10	10	21.2962963	30.161013	0.37854	0.45341
V2src11_hrc1_625.yuv	11	1	50.25925926	55.909908	0.79919	0.84263
V2src11_hrc5_625.yuv	11	5	35.51851852	44.049999	0.59256	0.64572
V2src11_hrc7_625.yuv	11	7	18.7037037	26.877754	0.34337	0.4147
V2src11_hrc10_625.yuv	11	10	15.07407407	23.420477	0.30567	0.37674
V2src12_hrc1_625.yuv	12	1	36.33333333	43.837097	0.61418	0.64244
V2src12_hrc5_625.yuv	12	5	38.44444444	40.349903	0.6661	0.59008
V2src12_hrc7_625.yuv	12	7	31.11111111	37.254383	0.53242	0.54594
V2src12_hrc10_625.yuv	12	10	26.14814815	28.953564	0.44737	0.43887
V2src13_hrc1_625.yuv	13	1	43.7037037	38.333649	0.74225	0.56108
V2src13_hrc5_625.yuv	13	5	43.2962963	34.290554	0.66799	0.5058
V2src13_hrc7_625.yuv	13	7	25.2962963	26.990025	0.42065	0.41598
V2src13_hrc10_625.yuv	13	10	15.88888889	20.181463	0.33381	0.34373

Annex 3

Model 2

CONTENTS

1 Introduction

Traditionally, the evaluation of video quality is performed by a number of evaluators who subjectively evaluate the video quality. The evaluation can be done with or without reference videos. In referenced evaluation, evaluators are shown two videos: the reference (source) video and the processed video that is to be compared with the source video. By comparing the two videos, the evaluators give subjective scores to the videos. Therefore, it is often called a subjective test of video quality. Although the subjective test is considered to be the most accurate method since it reflects human perception, it has several limitations. First of all, it requires a number of evaluators. Thus, it is time-consuming and expensive. Furthermore, it cannot be done in real time. As a result, there has been a great interest in developing objective methods for video quality measurement. An important requirement for an objective method for video quality measurement is that it should provide consistent performance results over a wide range of video sequences that are not used in the design stage. Toward this goal, a model was developed, which is easy to implement, fast enough for real-time implementations and robust over a wide range of video impairments. The model is a product of collaborative works from Yonsei University, SK Telecom, and Radio Research Laboratory, Republic of Korea.

2 Objective measurement of video quality based on edge degradation

2.1 Edge PSNR (EPSNR)

The model for objective video quality measurement is a full reference method. In other words, it is assumed that a reference video is provided. By analysing how humans perceive video quality, it is observed that the human visual system is sensitive to degradation around the edges. In other words, when the edge areas of a video are blurred, evaluators tend to give low scores to the video even though the overall mean squared error is small. It is further observed that video compression algorithms tend to produce more artefacts around edge areas. Based on this observation, the model provides an objective video quality measurement method that measures degradation around the edges. In the model, an edge detection algorithm is first applied to the source video sequence to locate the edge areas. Then, the degradation of those edge areas is measured by computing the mean squared error. From this mean squared error, the EPSNR is computed and used as a video quality metric after post-processing.

In the model, an edge detection algorithm needs to be first applied to locate edge areas. One can use any edge detection algorithm, though there may be minor differences in the results. For example, one can use any gradient operator to locate edge areas. A number of gradient operators have been proposed. In many edge detection algorithms, the horizontal gradient image $g_{horizontal}(m,n)$ and the vertical gradient image $g_{vertical}(m,n)$ are first computed using gradient operators. Then, the magnitude gradient image $g(m,n)$ may be computed as follows:

$$g(m,n)=\left|g_{horizontal}(m,n)\right|+\left|g_{vertical}(m,n)\right|$$

Finally, a thresholding operation is applied to the magnitude gradient image $g(m,n)$ to find edge areas. In other words, pixels whose magnitude gradients exceed a threshold value are considered as edge areas.

Figures 7-11 illustrate the above procedure. Figure 7 shows a source image. Figure 8 shows a horizontal gradient image $g_{horizontal}(m,n)$, which is obtained by applying a horizontal gradient operator to the source image of Fig. 7. Figure 9 shows a vertical gradient image $g_{vertical}(m,n)$, which is obtained by applying a vertical gradient operator to the source image of Fig. 7. Figure 10 shows the magnitude gradient image (edge image) and Fig. 11 shows the binary edge image (mask image) obtained by applying thresholding to the magnitude gradient image of Fig. 10.

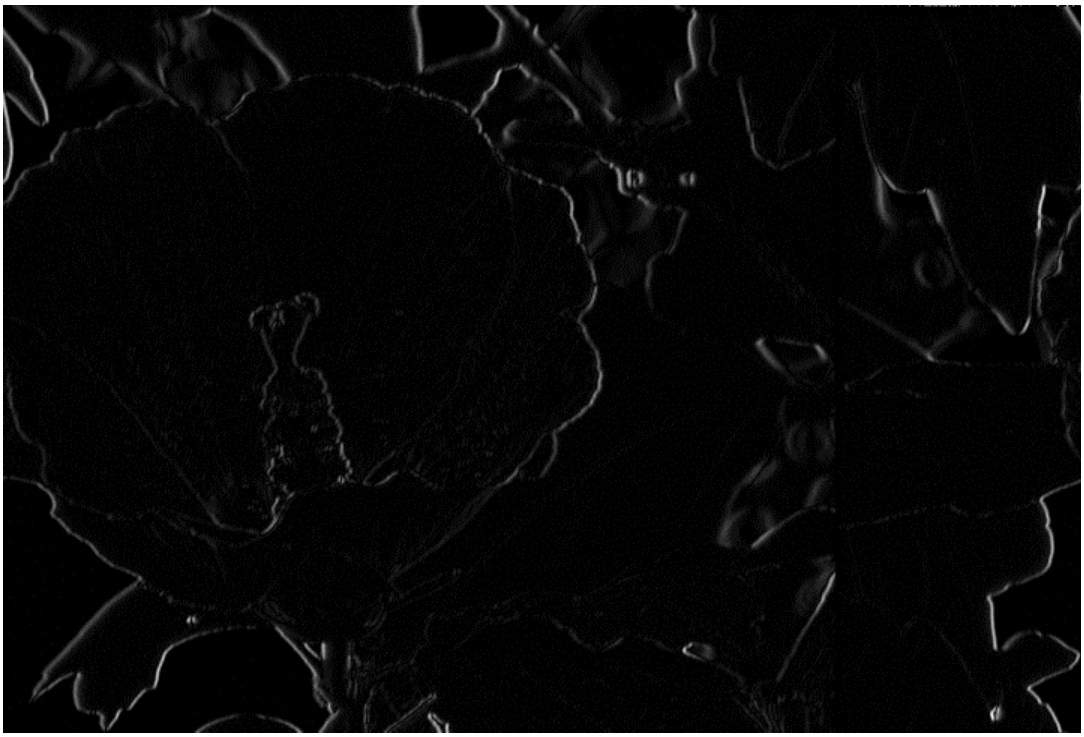
FIGURE 7

A source image (original image)

1683-07

FIGURE 8

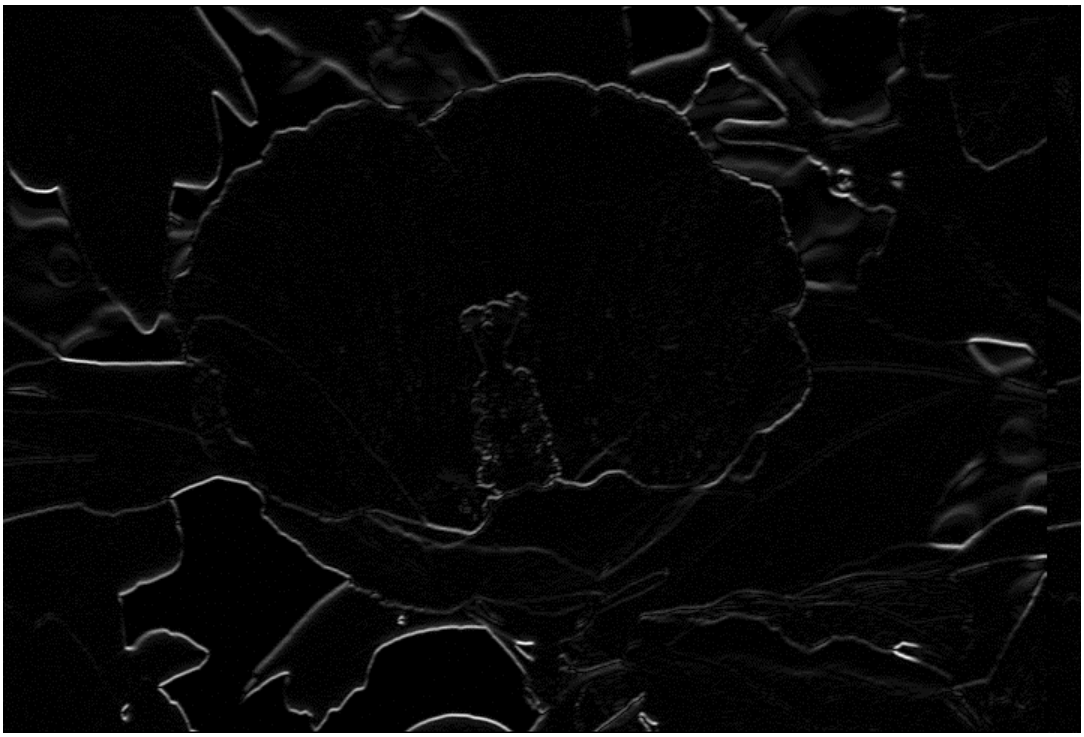
A horizontal gradient image, which is obtained by applying a horizontal gradient operator to the source image of Fig. 7



1683-08

FIGURE 9

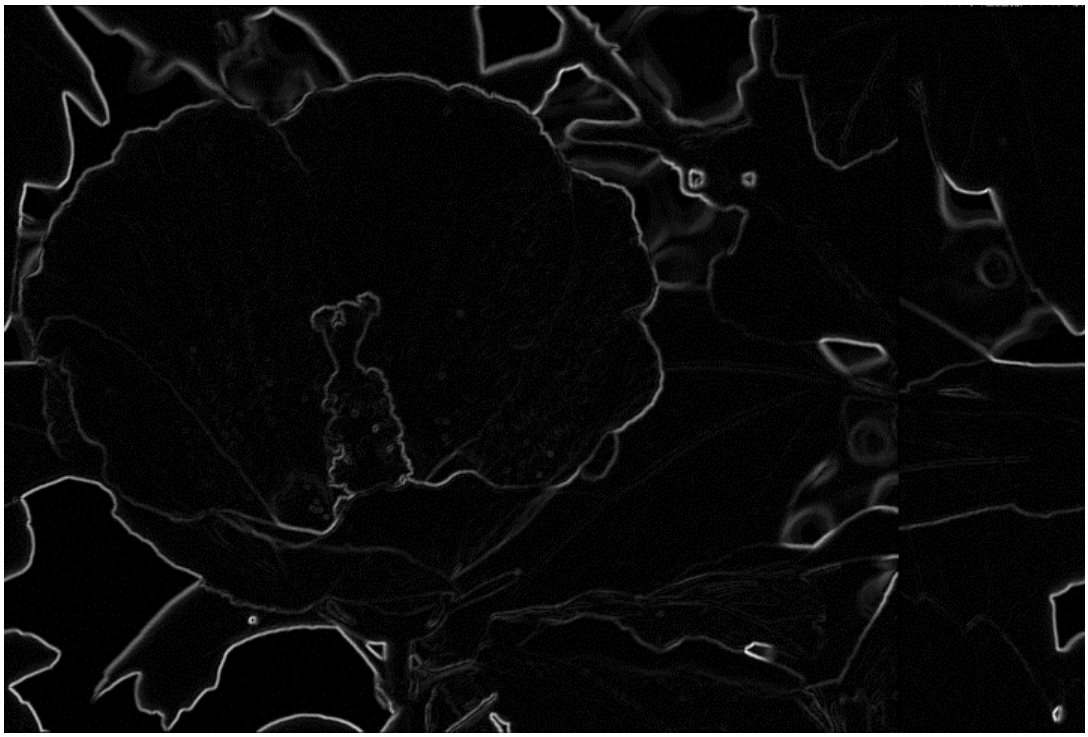
A vertical gradient image, which is obtained by applying a vertical gradient operator to the source image of Fig. 7



1683-09

FIGURE 10

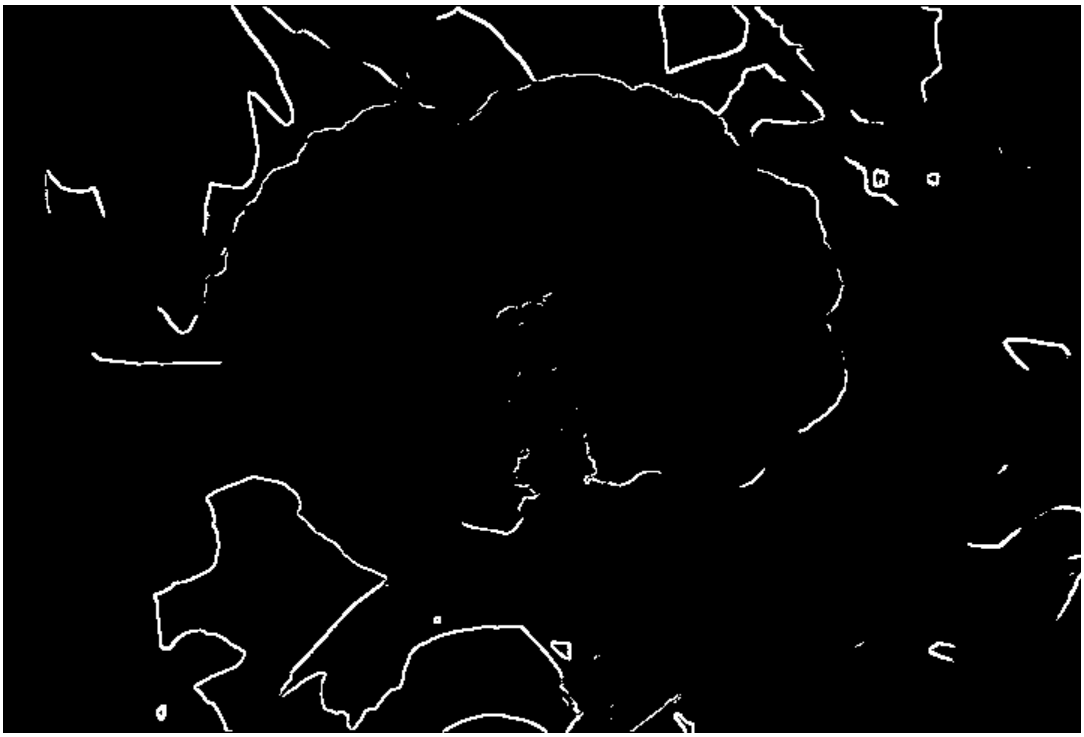
A magnitude gradient image



1683-10

FIGURE 11

A binary edge image (mask image) obtained by applying thresholding to the magnitude gradient image of Fig. 10

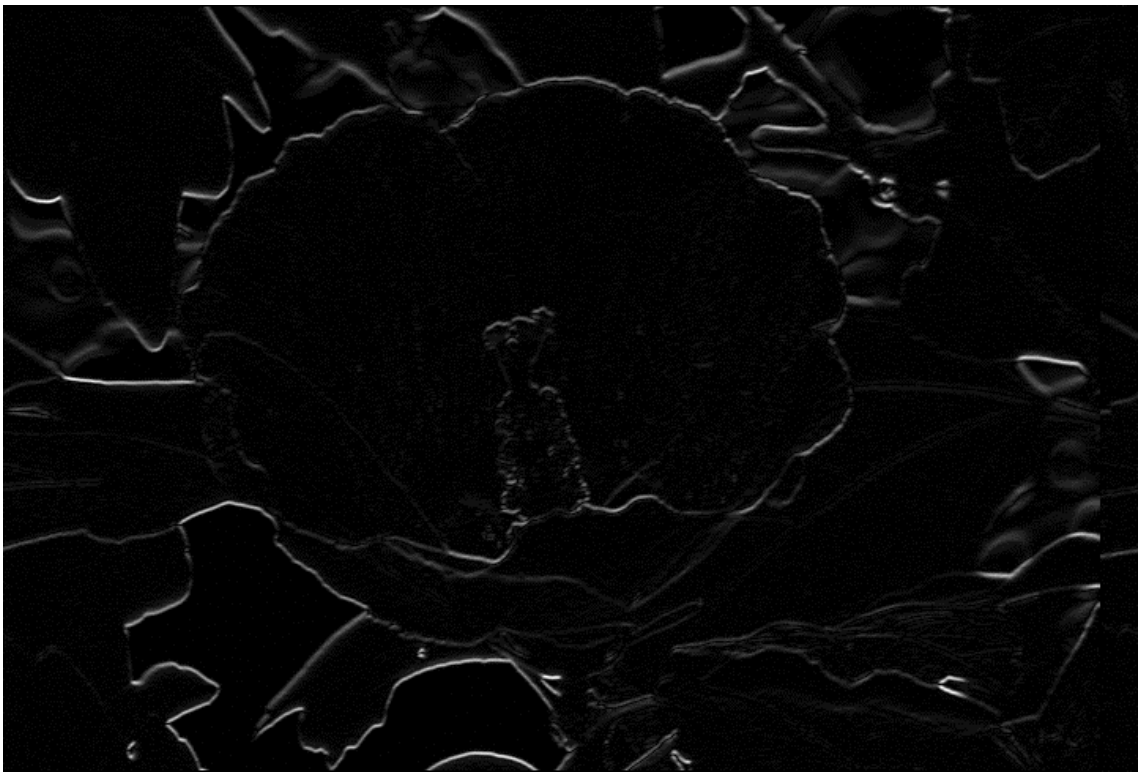


1683-11

Alternatively, one may use a modified procedure to find edge areas. For instance, one may first apply a vertical gradient operator to the source image, producing a vertical gradient image. Then, a horizontal gradient operator is applied to the vertical gradient image, producing a modified successive gradient image (horizontal and vertical gradient image). Finally, a thresholding operation may be applied to the modified successive gradient image to find edge areas. In other words, pixels of the modified successive gradient image, which exceed a threshold value, are considered as edge areas. Figures 12-15 illustrate the modified procedure. Figure 12 shows a vertical gradient image $g_{vertical}(m,n)$, which is obtained by applying a vertical gradient operator to the source image of Fig. 7. Figure 13 shows a modified successive gradient image (horizontal and vertical gradient image), which is obtained by applying a horizontal gradient operator to the vertical gradient image of Fig. 12. Figure 14 shows the binary edge image (mask image) obtained by applying thresholding to the modified successive gradient image of Fig. 13.

FIGURE 12

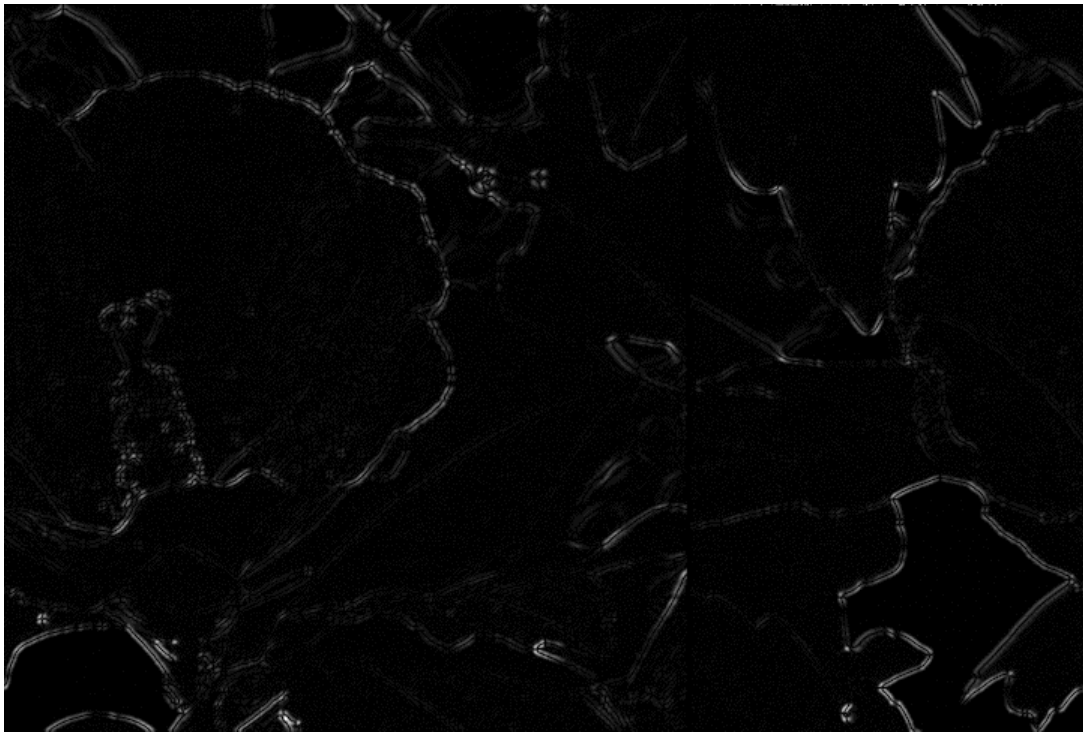
A vertical gradient image, which is obtained by applying a vertical gradient operator to the source image of Fig. 7



1683-12

FIGURE 13

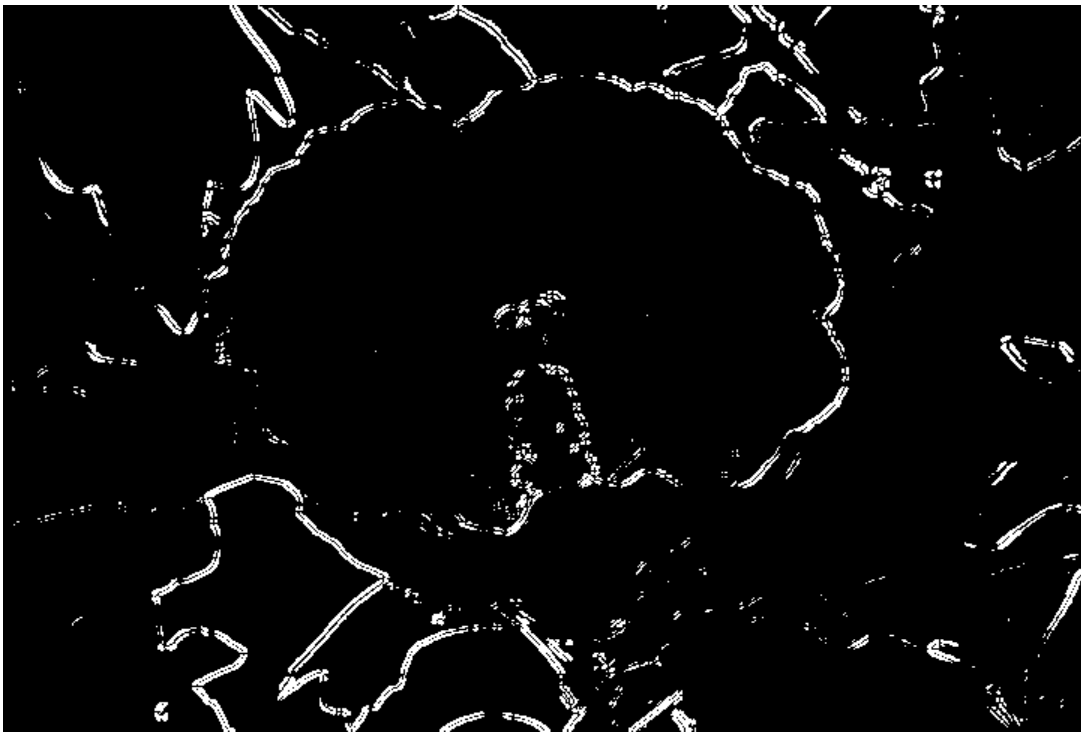
A modified successive gradient image (horizontal and vertical gradient image), which is obtained by applying a horizontal gradient operator to the vertical gradient image of Fig. 12



1683-13

FIGURE 14

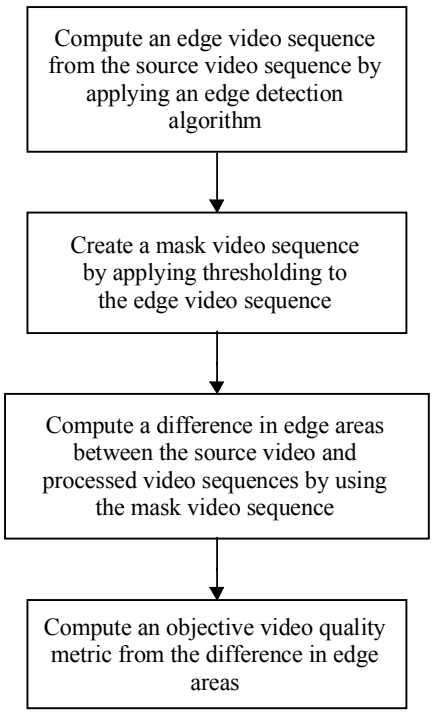
A binary edge image (mask image) obtained by applying thresholding to the modified successive gradient image of Fig. 13



1683-14

FIGURE 15

A block diagram of EPSNR



1683-15

It is noted that both methods can be understood as an edge detection algorithm. One may choose any edge detection algorithm depending on the nature of videos and compression algorithms. However, some methods may outperform other methods.

Thus, in the model, an edge detection operator is first applied, producing edge images (Figs. 10 and 13). Then, a mask image (binary edge image) is produced by applying thresholding to the edge image (Figs. 11 and 14). In other words, pixels of the edge image whose value is smaller than threshold, t_e, are set to zero and pixels whose value is equal to or larger than the threshold are set to a non-zero value. Figures 11 and 14 show examples of mask images. It is noted that this edge detection algorithm is applied to the source image. Although one may apply the edge detection algorithm to processed images, it is more accurate to apply it to the source images. Since a video can be viewed as a sequence of frames or fields, the above-stated procedure can be applied to each frame or field of videos. Since the model can be used for field-based videos or frame-based videos, the terminology "image" will be used to indicate a field or frame.

Next, differences between the source video sequence and processed video sequence, corresponding to non-zero pixels of the mask image are computed. In other words, the squared error of edge areas of the l-th frame is computed as follows:

$$se_e^l = \sum_{i=1}^{M} \sum_{j=1}^{N} \{S^l(i,j) - P^l(i,j)\}^2 \qquad \text{if } \left| R^l(i,j) \right| \neq 0 \tag{75}$$

where:

$S^l(i,j)$: l-th image of the source video sequence

$P^l(i,j)$: l-th image of the processed video sequence

$R^l(i,j)$: l-th image of the mask video sequence

M: number of rows

N: number of columns.

When the model is implemented, one may skip the generation of the mask video sequence. In fact, without creating the mask video sequence, the squared error of edge areas of the l-th frame is computed as follows:

$$se_e^l = \sum_{i=1}^{M} \sum_{j=1}^{N} \{S^l(i,j) - P^l(i,j)\}^2 \qquad \text{if } \left| Q^l(i,j) \right| \geq t_e \tag{76}$$

where:

$Q^l(i,j)$: l-th image of the edge video sequence

t_e: threshold.

Although the mean squared error is used in equation (75) to compute the difference between the source video sequence and the processed video sequence, any other type of difference may be used. For instance, the absolute difference may be also used. In the model submitted to the VQEG Phase II test, t_e, was set to 260 and the modified edge detection algorithm was used with the Sobel operator.

This procedure is repeated for the entire video sequences and the edge mean squared error is computed as follows:

$$mse_e = \frac{1}{K} \sum_{l=1}^{L} se_e^l \tag{77}$$

where:

L: number of images (frames or fields)

K: total number of pixels of the edge areas.

Finally, the PSNR of the edge areas (EPNSR) is computed as follows:

$$EPSNR = 10 \log_{10}\left(\frac{P^2}{mse_e}\right) \tag{78}$$

where:

P: peak pixel value.

In the model, this EPSNR is used as a basic objective video quality score. Figure 15 shows a block diagram of computing the EPSNR.

2.2 Post adjustments

2.2.1 De-emphasis of high EPSNR

When the value of EPSNR is over 35, it appears that the EPSNR overestimates perceptual quality. Thus, the following piecewise linear scaling is used:

$$EPSNR = \begin{cases} EPSNR & \text{if } 0 \le EPSNR \le 35 \\ EPSNR \times 0.9 & \text{if } 35 < EPSNR \le 40 \\ EPSNR \times 0.8 & \text{if } EPSNR > 40 \end{cases} \tag{79}$$

2.2.2 Considering blurred edges

It is observed that when edges are severely blurred in low quality videos, evaluators tend to give lower subjective scores. In other words, if the edge areas of the processed video sequence are substantially smaller than those of the source video sequence, the evaluators give lower scores. Furthermore, it is observed that some video sequences have a very small number of pixels which have high frequency components. In other words, the number of pixels of edge areas is very small. In order to take into account these problems, the edge areas of the source and processed video sequences are computed and the EPSNR is modified as follows:

$$MEPSNR = \begin{cases} MEPSNR - 60 \times \left(0.1.225 - \left(\frac{EP_{common}}{EP_{src}}\right)^2\right) \\ MEPSNR \end{cases}$$

$$\text{if} \quad EPNSR < 25 \quad \text{and} \quad \left(\frac{EP_{common}}{EP_{scr}}\right)^2 < 0.35 \tag{80}$$

$$\text{and} \quad \frac{EP_{hrc}}{EP_{src}} < 0.13 \text{ elsewhere}$$

where:

$MEPSNR$: modified EPSNR

EP_{common}: total number of common edge pixels in the SRC and HRC video sequences (i.e. edge pixels occurring at the same location)

EP_{src}: total number of edge pixels in the SRC (source) video sequence.

For some video sequences, EP_{src} can be very small. If EP_{src} is smaller than 10 000 pixels (about 10 000/240 = 41.7 pixels per frame for the 8 s 525 videos and about 10 000/200 = 50 pixels per frame for the 8 s 625 videos), the user may reduce threshold t_e in equation (76) by 20 until EP_{src} is larger than or equal to 10 000 pixels. If EP_{src} is smaller than 10 000 pixels even when t_e is reduced to 80, the post adjustment using equation (80) is not used. In this case, the *EPSNR* is computed using $t_e = 60$. If this option is taken, the user may delete the condition of $EP_{hrc}/EP_{src} < 0.13$ in equation (80).

2.2.3 Scaling

Next, scale objective scores are rescaled so that they will be between 0 (not distinguishable from the original video) and 1.

$$VQM = 1 - MEPSNR \times 0.02 \tag{81}$$

This VQM is used as the objective score of the model.

2.3 Registration accuracy

The recommended registration accuracy for the model is a half-pixel accuracy in the interlaced videos, which is equivalent to a quarter-pixel accuracy in the progressive video format. The cubic spline interpolation [Lee *et al.*, 1998] or better is strongly recommended to calculate sub-pixel values.

2.4 The block diagram of the model

Figure 16 shows the complete block diagram of the model.

3 Objective data

The model was applied to the VQEG Phase II video data[1]. However, after the model was submitted, registration and operator errors were found. The objective data presented in this Annex is the same data as in the VQEG Phase II Final Report. Consequently, when the method described in this Annex is properly implemented, the user may obtain different objective data from those of this Annex. Tables 7 to 8 show the objective data of the 525 and 625 video data sets.

4 Conclusion

A new model for objective measurement of video quality is proposed based on edge degradation. The model is extremely fast. Once the bit-map is generated, the model is several times faster than the conventional PSNR, providing a significant improvement. Therefore, the model is well suited to applications which require real-time video quality evaluation.

5 Reference

LEE C., EDEN M. and UNSER M., [1998] High quality image resizing using oblique projection operators. *IEEE Trans. Image Processing*, Vol. 5, **5**, p. 679-692.

FIGURE 16

The complete block-diagram of the model

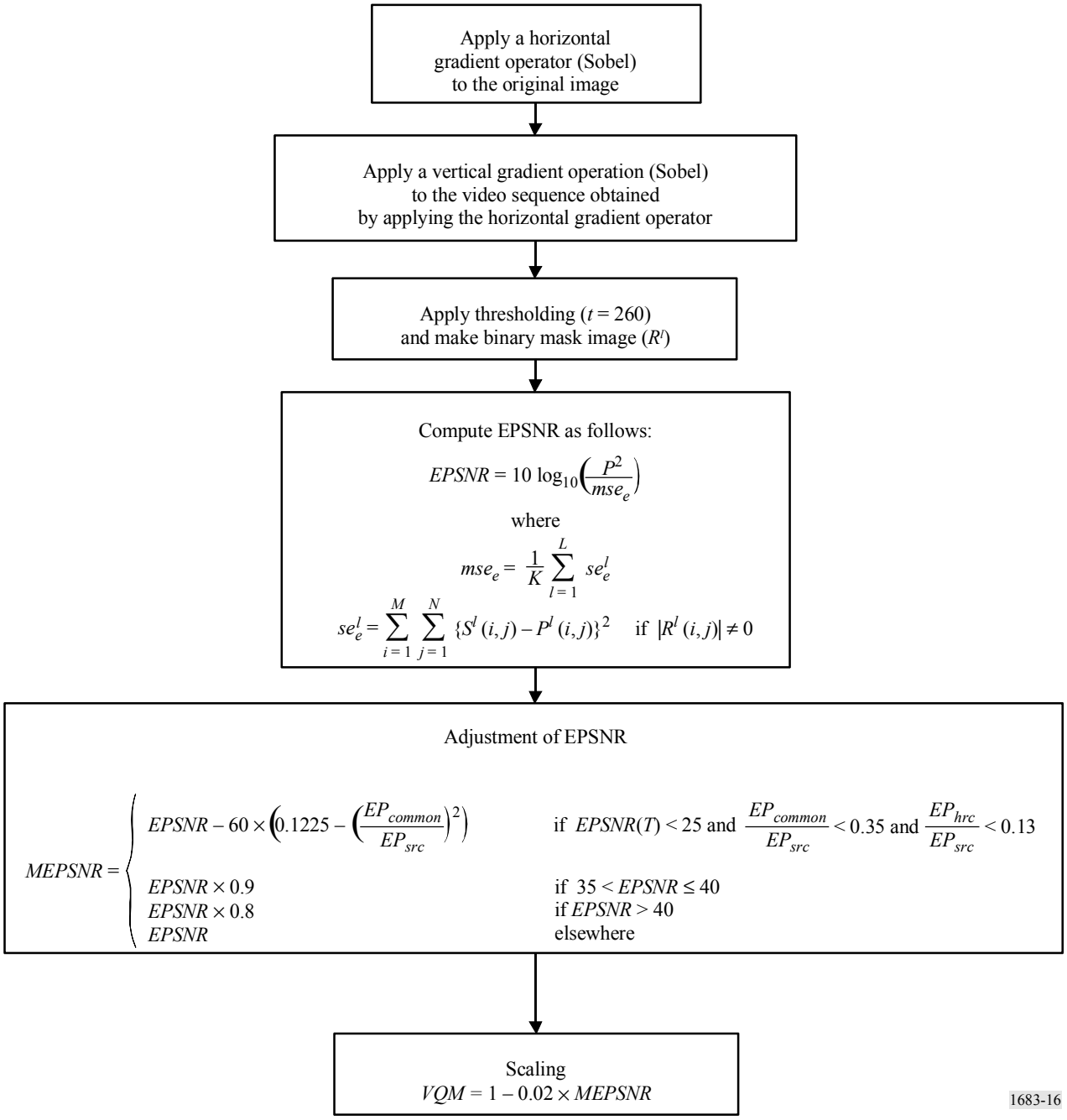


TABLE 7

The 525 VQM matrix (yonsei_1128c.exe)[1]

SRC (image)	HRC																													
	1		2		3		4		5		6		7		8		9		10		11		12		13		14	
1	1	0.679	4	0.525	7	0.512	10	0.419																				
2	2	0.431	5	0.365	8	0.313	11	0.342																				
3	3	0.558	6	0.452	9	0.340	12	0.305																				
4									13	0.668	17	0.581	21	0.556	25	0.535												
5									14	0.543	18	0.485	22	0.443	26	0.410												
6									15	0.631	19	0.477	23	0.441	27	0.411												
7									16	0.467	20	0.415	24	0.376	28	0.346												
8																	29	0.787	35	0.734	41	0.740	47	0.551	53	0.520	59	0.537
9																	30	0.848	36	0.559	42	0.723	48	0.495	54	0.462	60	0.465
10																	31	0.552	37	0.449	43	0.542	49	0.352	55	0.308	61	0.377
11																	32	0.610	38	0.628	44	0.633	50	0.475	56	0.471	62	0.498
12																	33	0.576	39	0.539	45	0.577	51	0.470	57	0.436	63	0.448
13																	34	0.554	40	0.569	46	0.517	52	0.399	58	0.382	64	0.412

[1] After the model was submitted, registration and operator errors were found. The objective data presented in this Annex is the same data as in the VQEG Phase II Final Report. Consequently, when the method described in this Annex is properly implemented, the user may obtain different objective data from those of Table 7.

TABLE 8

The 625 VQM matrix (yonsei_1128c.exe)[1]

SRC (image)	HRC																			
	1		2		3		4		5		6		7		8		9		10	
1			4	0.612	10	0.531	16	0.452			29	0.434			42	0.436			52	0.382
2			5	0.544	11	0.540	17	0.451			30	0.437			43	0.440			53	0.363
3			6	0.572	12	0.571	18	0.497			31	0.479			44	0.478			54	0.418
4			7	0.601	13	0.656	19	0.557			32	0.547			45	0.526			55	0.472
5			8	0.603	14	0.621	20	0.500			33	0.492			46	0.444			56	0.390
6			9	0.591	15	0.520	21	0.483			34	0.469			47	0.461			57	0.423
7							22	0.576			35	0.555					48	0.531	58	0.501
8							23	0.512			36	0.500					49	0.482	59	0.457
9							24	0.507			37	0.487					50	0.468	60	0.436
10							25	0.610			38	0.594					51	0.575	61	0.540
11	1	0.753							26	0.594			39	0.508					62	0.485
12	2	0.643							27	0.556			40	0.550					63	0.496
13	3	0.669							28	0.524			41	0.481					64	0.441

[1] After the model was submitted, registration and operator errors were found. The objective data presented in this Annex is the same data as in the VQEG Phase II Final Report. Consequently, when the method described in this Annex is properly implemented, the user may obtain different objective data from those of Table 8.

Annex 4

Model 3

CONTENTS

1 Introduction

This Annex presents a methodology for video quality assessment using objective parameters based on image segmentation. Natural scenes are segmented into plane, edge and texture regions, and a set of objective parameters are assigned to each of these contexts. A perceptual-based model that predicts subjective (Recommendation ITU-R BT.500 and Recommendation ITU-R BT.802 – Test pictures and sequences for subjective assessments of digital codecs conveying signals produced according to Recommendation ITU-R BT.601) ratings is defined by computing the relationship between objective measures and results of subjective assessment tests, applied to a set of natural

scenes processed by MPEG-2 video codecs. In this model, the relationship between each objective parameter processed by several compression systems (like MPEG-2 and MPEG-1 codecs) and the subjective impairment level is approximated by a logistic curve, resulting in an estimated impairment level for each parameter. The final result is achieved through a linear combination of estimated impairment levels, where the weight of each impairment level is proportional to its statistical reliability.

In § 2, a general description of CPqD-IES (image evaluation based on segmentation) system is presented. In § 3, the steps to register spatial and temporal misalignments, as well as the correction of gain are described. In § 4, the algorithm to segment images into plane, edge and texture regions is explained. In § 5, the objective measurement carried out over each region and each image component is described. Section 6 describes the way that the database of impairment models was constructed. The calculations of the parameters also are described in this Section. Section 7 describes the estimation of video quality rating from the parameters in database of impairment models. Annex 4a presents results of video quality rating (VQR) estimated during VQEG Phase II tests[1].

2 General description of the IES system

Figure 17 presents an overview of the CPqD-IES algorithm for natural scenes. Each natural scene is represented by one original (reference) scene ***O*** and one impaired scene ***I***, which results from a codec operation applied to ***O***. Offset and gain corrections are applied to ***I*** in order to create a corrected impaired scene ***I'***, such that each frame *f* of ***I'*** corresponds to the reference frame *f* of ***O*** for *f* = 1, 2, ..., *n* (§ 3.2).

Input scenes ***I*** and ***O*** to the CPqD-IES algorithm are in YCbCr4:2:2 format according to Recommendation ITU-R BT.601 – Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios.

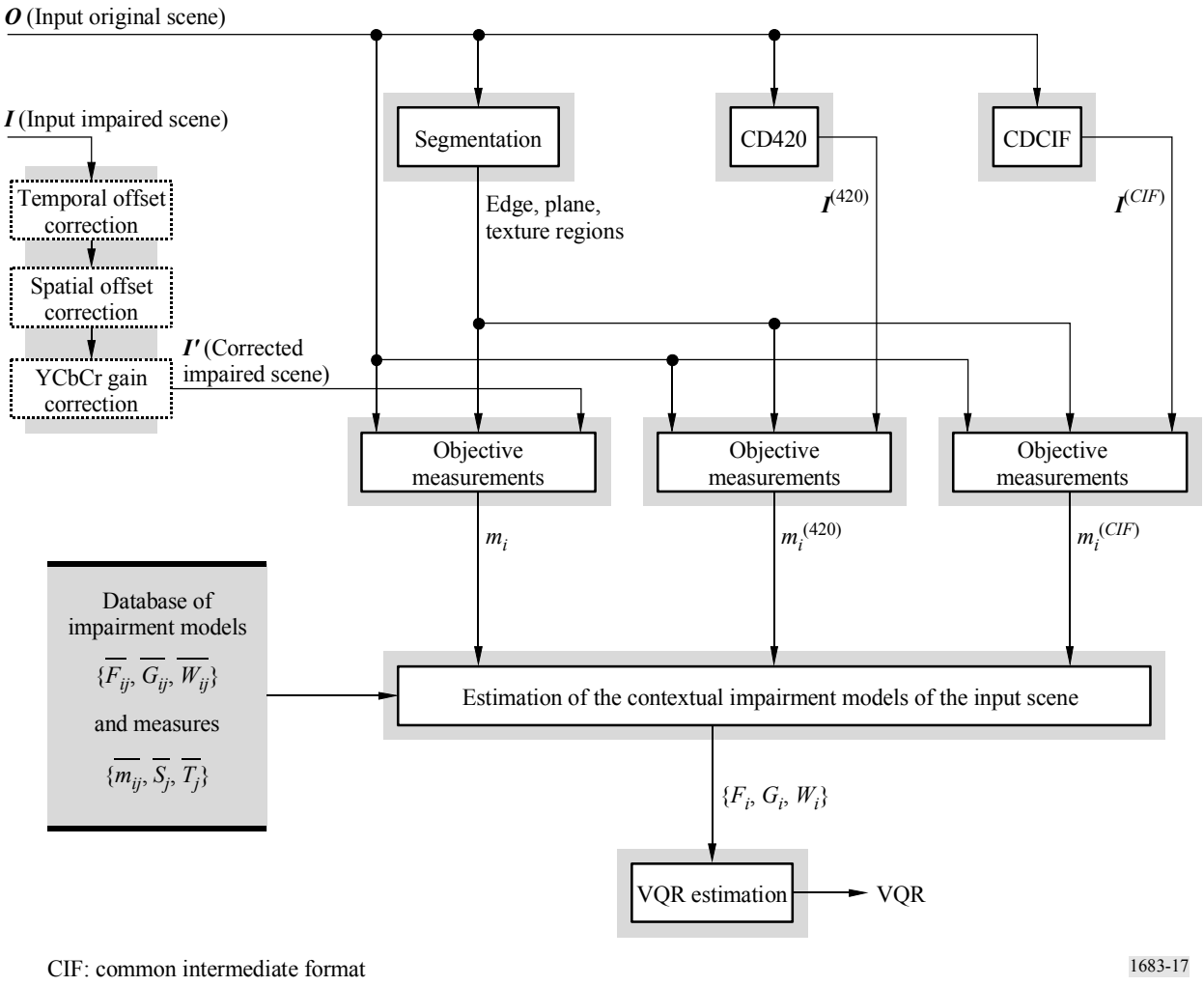
The Y component of each image frame *f* of ***O*** is segmented into three categories: texture, edge, and plane regions (§ 4). One objective measure is computed based on the difference between the corresponding frames of ***O*** and ***I'***, for each of these contexts and for each image component YCbCr, forming a set of 9 objective measures {*m*1, *m*2, ..., *m*9} for each image frame *f* (§ 5). Each objective measure m_i, $i = 1, 2, ..., 9$, produces a contextual impairment level L_i based on its impairment estimation model, which is given by:

$$L_i = 100 / \left[1 + \left(\frac{F_i}{m_i} \right)^{G_i} \right] \tag{82}$$

where F_i and G_i are two parameters computed (§ 7) based on a database of impairment models (Section 6), spatial *S* and temporal *F* attributes (§ 5), and on the objective measures $m_i^{(420)}$ and $m_i^{(CIF)}$ for frame *f*, resulting from the codec operations CD420 and CDCIF applied to ***O*** (§ 7). The two reference impairment codecs, CD420 (coder/decoder MPEG-2 4:2:0) and CDCIF (coder/decoder MPEG-1 CIF), are totally based on the routines extracted directly from MPEG2 (ITU-T Recommendation H.262 – Information technology – Generic coding of moving pictures and associated audio information: Video.) and MPEG1 [ISO/IEC, 1992], available at http://www.mpeg.org/MPEG/MSSG. In the current implementation of the CPqD-IES algorithm, these routines operate in intra mode using a fix quantization step of 16. It is important to note that CD420 and CDCIF do not introduce offset and gain differences with respect to ***O***.

FIGURE 17

General overview of the CPqD-IES algorithm



The video quality rate VQR_f of frame f is obtained by linear combination of the contextual impairment levels L_i, i = 1, 2, ..., 9, as follows:

$$VQR_f = \sum_{i=1}^{9} W_i \, L_i \tag{83}$$

where W_i is the weight of the impairment level L_i for this particular natural scene, which is computed as described in § 7.

Now, the sequence of values VQR_1, VQR_2, ... VQR_n is transformed by a median filter of size 3 into another sequence VQR'_1, VQR'_2, ... VQR'_n, by excluding the median value computation within the 1-neighbourhood of VQR_1 and VQR_n. During the median filtering, the algorithm avoids repetition of two consecutive median values. That is, if the median value VQR'_{f-1} computed within the

1-neighbourhood of VQR_f is equal to the median value VQR'_{f-2}, computed within the 1-neighbourhood of VQR_{f-1}, then the algorithm chooses VQR'_{f-1} as the minimum value computed within the 1-neighbourhood of VQR_f. This algorithm can be described as follows.

1) For each f from 2 to $n - 1$, do

2) Compute *med*, the median value among VQR_{f-1}, VQR_f, VQR_{f+1}

3) If $med = VQR'_{f-2}$ then

4) Compute VQR'_{f-1} as the minimum value among VQR_{f-1}, VQR_f, VQR_{f+1}

5) Else

6) $VQR'_{f-1} \leftarrow med$.

The final VQR is then the average of the VQR_f' values.

$$VQR = \frac{1}{n-2} \cdot \sum_{f=1}^{n-2} VQR'_f \tag{84}$$

Equations (82), (83) and the above algorithm describe the process to estimate the VQR from the contextual impairment models $\{F_i, G_i, W_i\}$ and the objective measures m_i, i = 1, 2, ..., 9. The next sections complete the description of the method by presenting the details inside the remaining blocks of Fig. 17.

3 Correction of offset and gain

3.1 Temporal offset

The temporal offset, *dt*, is an integer ranging from –2 to 2. Input scenes with temporal offsets out of this range are not considered. Let $\boldsymbol{I}_{dt}$ be the impaired scene $\boldsymbol{I}$ with a displacement of f frames. A dissimilarity coefficient between $\boldsymbol{O}$ and each displaced scene $\boldsymbol{I}_{dt}$ is calculated. The displacement with lowest dissimilarity coefficient is used as temporal offset, and the output $\boldsymbol{I}_{dt}$ is then $\boldsymbol{I}$ displaced by this offset for the next computation. The dissimilarity coefficient between $\boldsymbol{O}$ and $\boldsymbol{I}_{dt}$ is obtained as described below, where n is the number of frames in the temporal intersection between them:

1) $\xi_T \leftarrow 0$

2) For each f from 1 to n, do

3) Compute S_b

4) Compute S'_b

5) Compute D_b

6) Compute μ, the mean value of the pixels in Db

7) $\xi_T \leftarrow \xi_T + (\mu/n)$

8) Return ξ_T (dissimilarity coefficient between $\boldsymbol{O}$ and $\boldsymbol{I}_{dt}$).

Where:

S_b: magnitude of the Sobel's gradient of the component Y of the f-th frame of $\boldsymbol{O}$

S'_b: magnitude of the Sobel's gradient of the component Y of the f-th frame of $\boldsymbol{I}_{dt}$

D_b: pixel-wise absolute difference between S_b and S'_b.

3.2 Spatial offset

The spatial offset (d_x, d_y) is one of the following integer horizontal and vertical displacements d_x = –6, –5, ... 6 and d_y = –6, –5, ..., 6. Consider $\boldsymbol{I}_{dx,dy}$ the impaired scene $\boldsymbol{I}_{dt}$ with all frames displaced by (d_x, d_y) pixels. A dissimilarity coefficient between $\boldsymbol{O}$ and $\boldsymbol{I}_{dx,dy}$ is calculated. The spatial displacement with lowest dissimilarity is used as spatial offset, and the output $\boldsymbol{I}_{dx,dy}$ is then $\boldsymbol{I}_{dt}$ displaced by this offset for the next computation.

The dissimilarity between $\boldsymbol{O}$ and $\boldsymbol{I}_{dx,dy}$ is described below:

1) $\xi_S \leftarrow 0$; $c \leftarrow 0$
2) For each f from 1 to n, do
3) For x from $x0$ to ($x0 + w/4$) do
4) For y from $y0$ to ($y0 + h/4$) do
5) $\xi_S \leftarrow \xi_S + |Y(4x,4y) - Y'(4x + dx, 4y + dy)| +$
$+ |Cb(4x,4y) - Cb'(4x + dx, 4y + dy)| +$
$+ |Cr(4x,4y) - Cr'(4x + dx, 4y + dy)|$
6) $c \leftarrow c + 3$
7) $\xi_S \leftarrow \xi_S / c$
8) Return ξ_S (dissimilarity coefficient between $\boldsymbol{O}$ and $\boldsymbol{I}_{dx,dy}$).

Where:

$w \times h$: size of the intersection area between $\boldsymbol{O}$ and $\boldsymbol{I}_{dx,dy}$

$Y(x, y)$, $Cb(x, y)$, $Cr(x, y)$: values in the image components of a frame f of $\boldsymbol{O}$ for a pixel (x, y)

$Y'(x + dx, y + dy)$

$Cb'(x + dx, y + dy)$

$Cr'(x + dx, y + dy)$: values in the image components of a frame f of $\boldsymbol{I}_{dx,dy}$ for a pixel ($x + dx$, $y + dy$).

3.3 Gain

The amplitude gain between $\boldsymbol{O}$ and $\boldsymbol{I}_{dx,dy}$ is calculated for each image component Y, C_B and C_R, separately. The algorithm computes the average of the gains over all n frames and corrects each image component accordingly. The output $\boldsymbol{I'}$ is the impaired scene used for all subsequent computations. The amplitude gain between an image component C of the frame f in $\boldsymbol{I}_{dx,dy}$ with respect to the same component C of the frame f in $\boldsymbol{O}$ is obtained by blurring both images C' and C, using a Gaussian filter [Gonzalez and Woods, 1992] with kernel:

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

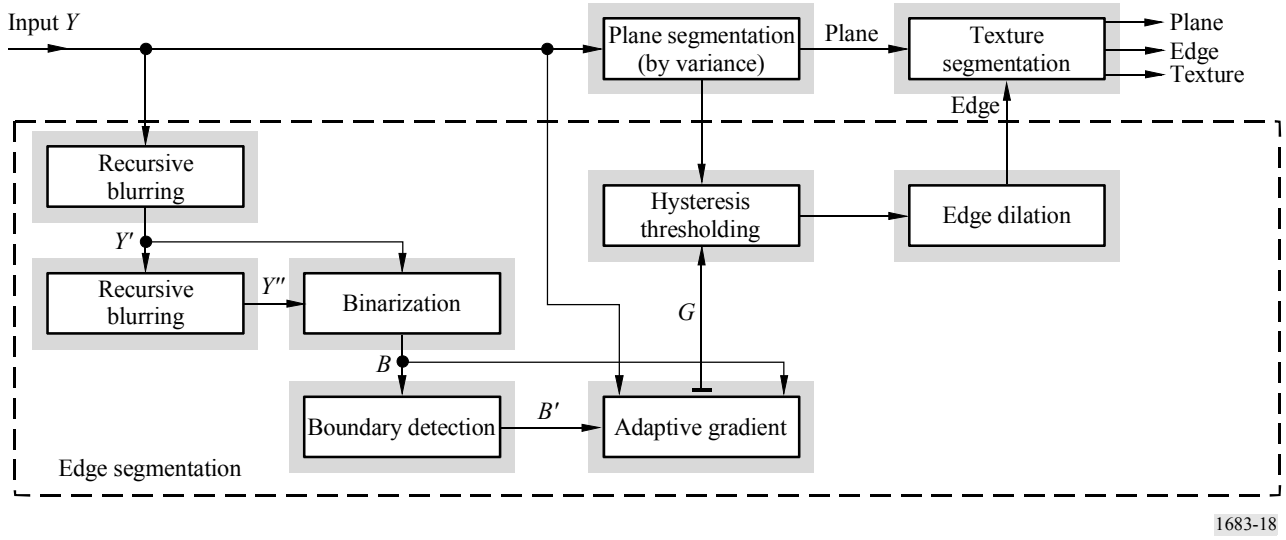
and computing the ratio between the sum of their pixel values in the blurred images. Only 1 out of each 16 pixels is considered (by sweeping the blurred component images with horizontal and vertical increments of 4 pixels, as in the ξ_s calculation algorithm presented in § 3.2).

4 Image segmentation

Initially, the segmentation algorithm classifies each pixel in the component Y of a given frame f of the original scene $\boldsymbol{O}$ into plane and non-plane regions. The algorithm also applies to Y an edge detector and the edge region is defined by edges which fall within the boundary of the plane region. The texture region is composed of the remaining pixels of the image Y (see Fig. 18).

FIGURE 18

Block diagram of the segmentation process



The segmentation is computed over each frame of the Y component from the input original scene $\boldsymbol{O}$. For the C_B and C_R components, the regions are brought out by the position of the pixels in Y component, after up sampling in C_B and C_R.

4.1 Plane regions

The brightness variance of each pixel in Y is computed within the 5×5 neighbourhood of pixels around it. The image variance is thresholded such that pixels with variance value below 25^2 are classified as belonging to the plane region. This process creates small pixel components misclassified within the texture region. A 3×3-median filter is applied to remove these small components. Finally, the binary image of the plane regions is submitted to a morphological dilation using a circular structuring element with diameter of 11 pixels [Gonzalez and Woods, 1992].

4.2 Edge regions

A recursive filtering is applied to Y, creating a first blurred image Y', and then it is applied to Y' in order to create a second blurred image Y''. Each recursive filtering consists of four rasters in the input image. This algorithm is described below for the component image Y of the one frame of the input scene $\boldsymbol{O}$.

1) For y varying from 0 to $(h - 1)$ do

2) For x varying from 0 to $(w - 2)$ do

3) $Y(x + 1, y) \leftarrow Y(x, y) + 0.7\,[Y(x + 1, y) - Y(x, y)]$;

4) For y varying from 0 to $(h - 1)$ do

5) For x varying from $(w - 1)$ to 1 do

6) $Y(x - 1, y) \leftarrow Y(x, y) + 0.7\,[Y(x - 1, y) - Y(x, y)]$;

7) For x varying from 0 to $(w - 1)$ do

8) For y varying from 0 to $(h - 2)$ do

9) $Y(x, y + 1) \leftarrow Y(x, y) + 0.7\,[Y(x, y + 1) - Y(x, y)]$;

10) For x varying from 0 to $(w - 1)$ do

11) For y varying from $(h - 1)$ to 1 do

12) $Y(x, y + 1) \leftarrow Y(x, y) + 0.7\,[Y(-1) - Y(x, y)]$;

13) Save image Y in image Y'.

Where:

$Y(x, y)$: brightness of the pixel (x, y)

h: number of lines in Y

w: number of columns in Y.

The second application of the above algorithm will create Y''. A binary image B is created from Y' and Y'':

$$B(x,y) = \begin{cases} 1 & \text{if } Y'(x,y) \geq Y''(x,y) \\ 0 & \text{otherwise} \end{cases} \tag{85}$$

After that, the algorithm identifies the boundary pixels of the regions in B with pixel-value 1 by creating a second binary image B':

$$B'(x,y) = \begin{cases} 1 & \text{if } B(x,y) = 1 \text{ and } B(x',y') = 0 \text{ for any pixel } (x',y') \in N_8(x,y) \\ 0 & \text{otherwise} \end{cases} \tag{86}$$

where $N_8(x, y)$ is the set of pixels (x', y') within the 3 × 3 neighbourhood of (x, y) (i.e. its 8 neighbours).

An adaptive gradient filter is applied to Y restricted to the pixels where $B'(x, y) = 1$:

$$G(x,y) = \begin{cases} |\mu_1 - \mu_0| & \text{if } B'(x,y) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{87}$$

where:

μ_1: mean value of $Y(x', y')$, for all $(x', y') \in N_8(x, y)$ such that $B(x', y') = 1$

μ_0: mean value of $Y(x', y')$, for all $(x', y') \in N_8(x, y)$ such that $B(x', y') = 0$.

Note that, the algorithm uses B instead of B' to compute the mean values μ_1 and μ_0.

A hysteresis thresholding [Trucco and Verri, 1998] is applied to G restricted to pixels which have been classified in § 4.1 as belonging to the plane region. The lower threshold is 30 and the upper threshold is 40. The algorithm first identifies the pixels in G, such that $G(x, y) > 40$, and then it applies a region growing algorithm along the lines of G by using these pixels as seeds and by restricting the growth to pixels in the same line whose $G(x, y) > 30$. All 4-connected pixel components with less than 6 pixels are eliminated from this result. The final binary image is dilated by a circular structuring element with diameter of 5 pixels ignoring the restriction to the plane region. The pixels with value 1 in this dilation are classified as belonging to the edge region.

4.3 Texture regions

The texture region consists of the pixels in Y which were neither classified as belonging to the edge region nor to the plane region in the above sections.

5 Objective measurement

Consider S_b, the image of magnitude of the Sobel's gradient computed for a given component (Y, C_B or C_R) of a given frame f of the original scene $\boldsymbol{O}$, and S'_b, the image of magnitude of the Sobel's gradient for the same component of frame f of the impaired scene $\boldsymbol{I'}$. The image D_b of the pixel-wise absolute difference between S_b and S'_b is computed and the region $\Re$ of pixels in D_b that belong to a given context (plane, edge or texture) is considered. The absolute Sobel's difference (ASD) for this image component and context is defined as the average of the pixel values in D_b restricted to $\Re$.

This procedure produces a set of nine objective measures $\{m_1, m_2, ..., m_9\}$ for each image frame f, $f = 1, 2, ..., n$, considering all three contexts and three image components.

The same process is applied to create objective measures $\{m_1^{(420)}, m_2^{(420)}, ..., m_9^{(420)}\}$ and $\{m_1^{(CIF)}, m_2^{(CIF)}, ..., m_9^{(CIF)}\}$, for the frame f with respect to the MPEG-2 4:2:0 and MPEG-1 CIF CODEC operations over $\boldsymbol{O}$ (see Fig. 17). These measures are used as references together with spatial S and temporal T attributes in order to determine the contextual impairment model for $\boldsymbol{I'}$ (§ 7). The temporal attribute T is the mean value of the pixel-wise absolute difference between the segmentations of the frames f and $f - 1$, normalized within [0,1]. The spatial attribute S is defined as the ratio $m_7^{(CIF)}/m_7^{(420)}$, normalized in [0,1], where $m_7^{(CIF)}$ and $m_7^{(420)}$ are the corresponding ASDs for the texture region in the component Y of the frame f.

6 Database of impairment models

The IES system uses a database of impairment models for scenes different from the reference scene $\boldsymbol{O}$ in order to estimate the video quality rate of $\boldsymbol{I'}$. This database consists of information about twelve 60 Hz scenes representing different degrees of motion (dynamic and static scenes), nature (real and synthetic scenes), and context (amount of texture, plane, and edge pixels). This database was created as follows.

The mean values of the objective measures $\left\{\overline{m}_{1,j}, \overline{m}_{2,j}, ..., \overline{m}_{9,j}\right\}$, $\left\{\overline{m}_{1,j}^{(420)}, \overline{m}_{2,j}^{(420)}, ..., \overline{m}_{9,j}^{(420)}\right\}$, and $\left\{\overline{m}_{1,j}^{(CIF)}, \overline{m}_{2,j}^{(CIF)}, ..., \overline{m}_{9,j}^{(CIF)}\right\}$ were computed over the frames of each scene j, $j = 1, 2, ..., 12$. The values of S_j and T_j were calculated as the average of the spatial and temporal attributes, computed as described in § 5, over frames of each scene j. All impaired scenes of the database were also submitted to subjective evaluation, obtaining a subjective impairment level SL_j, normalized between 0% and 100% for each scene j.

According to equation (82), each objective measure $m_{i,j}$, $i = 1, 2, ..., 9$ and $j = 1, 2, ..., 12$, is related to a contextual impairment level $L_{i,j}$. The values of $F_{i,j}$ and $G_{i,j}$ in equation (82) were found for each scene j by minimizing the expectation of the mean square error $E[(\overline{SL}_j - \overline{L}_{i,j})^2]$. Moreover, the values of $W_{i,j}$ in equation (83) were computed in order to minimize the expectation of the mean square error

$$E\left[\left(\overline{SL}_j - \sum_{i=1}^{9} \overline{W}_{i,j}\ \overline{L}_{i,j}\right)^2\right] \tag{88}$$

Finally, the database of impairment models consists of nine sets $\{\overline{F}_{i,j}, \overline{G}_{i,j}, \overline{W}_{i,j}, \overline{S}_j, \overline{T}_j\}$, $i = 1, 2, ..., 9$, of parameters for each scene j, $j = 1, 2, ..., 12$. Table 9 contains the values of $\overline{S}_j$ and $\overline{T}_j$ to calculate the attributes $\{\overline{F}_{i,j}, \overline{G}_{i,j}, \overline{W}_{i,j}\}$.

TABLE 9

Temporal *T* and spatial *S* attributes

Scene *j*	*T* (temporal)	S_Y (spatial *Y*)	S_{Cb} (spatial C_B)	S_{Cr} (spatial C_R)
1	27.01	36.79	25.20	38.01
2	25.33	26.08	5.93	67.99
3	45.54	60.97	10.28	28.75
4	36.40	30.47	6.46	63.07
5	32.02	72.50	11.72	15.78
6	12.63	84.22	2.85	12.94
7	28.38	61.53	11.08	27.39
8	10.19	46.08	5.45	48.47
9	0.01	5.89	5.07	89.03
10	7.26	4.75	2.00	93.25
11	7.60	69.16	9.41	21.43
12	14.27	69.61	3.89	26.50

7 Estimation of impairment models

The contextual impairment models for a given frame f of $\boldsymbol{I'}$ consist of the parameters $\{F_i, G_i, W_i\}$ of equations (82) and (83), $i = 1, 2, ..., 9$. This section describes how to compute these parameters using the $I^{(420)}$ and $I^{(CIF)}$ impaired scenes as references.

7.1 Computation of W_i

The contextual local distances $D_{i,j}$ between a frame f of the impaired scenes, $I^{(420)}$ and $I^{(CIF)}$, and each scene j of the database are defined as:

$$\begin{aligned}
\overline{L}_{i,j}^{(420)} &= 100 / \left[1 + \left(\overline{F}_{i,j} / \overline{m}_i^{(420)} \right)^{\overline{G}_{i,j}} \right] \\
\overline{L}_{i,j}^{(CIF)} &= 100 / \left[1 + \left(\overline{F}_{i,j} / \overline{m}_i^{(CIF)} \right)^{\overline{G}_{i,j}} \right] \\
L_{i,j}^{(420)} &= 100 / \left[1 + \left(\overline{F}_{i,j} / m_i^{(420)} \right)^{\overline{G}_{i,j}} \right] \\
L_{i,j}^{(CIF)} &= 100 / \left[1 + \left(\overline{F}_{i,j} / m_i^{(CIF)} \right)^{\overline{G}_{i,j}} \right]
\end{aligned} \tag{89}$$

$L_{i,j}^{(420)}$ and are estimated impairment levels of the input scene ***O***, calculated with parameters $\overline{F}_{i,j}$ and $\overline{G}_{i,j}$, in the context *i*, of the scenes *j* from the database.

$$D_{i,j} = \frac{1}{2} \cdot \left(\left| L_{i,j}^{(420)} - \overline{L}_{i,j}^{(420)} \right| + \left| L_{i,j}^{(CIF)} - \overline{L}_{i,j}^{(420)} \right| \right) \tag{90}$$

The algorithm finds the set Ω of the six closest scenes of the database based on the $D_{i,j}$ distance and defines $W_{i,j}$ as:

$$a_k = \begin{cases} 1 & \text{if (scene } k) \in \Omega \\ 0 & \text{otherwise} \end{cases} \tag{91}$$

$$W_{i,j} = \frac{a_j \; D_{i,j}^{-1}}{\sum_{k=1}^{12} a_k \; D_{i,k}^{-1}} \tag{92}$$

Consider now that $i = \{1, 2, ..., 9\} \equiv \{$(*plane*, *Y*), (*plane*, C_B), (*plane*, C_R), (*edge*, *Y*), (*edge*, C_B), (*edge*, C_R), (*texture*, *Y*), (*texture*, C_B), (*texture*, C_R)$\}$, where (*plane*, *C*), (*edge*, *C*) and (*texture*, *C*) represent the plane, edge, and texture regions of the image component *C*, $C = Y, C_B, C_R$.

Let u = *texture*, *edge*, *plane* and $v = Y, C_B, C_R$, the values W_i, $i = 1, 2, ..., 9$, are computed as:

$$E_i = \sum_{j=1}^{12} D_{i,j} \; W_{i,j}$$

$$\kappa_{u,v} = \begin{cases} 1 & \text{if } v = Y_i \\ \frac{1}{2} & \text{otherwise} \end{cases}$$

$$\tau = \sum_u \left[\frac{1}{E_{u,Y}} + \frac{1}{2} \left(\frac{1}{E_{u,C_B}} + \frac{1}{E_{u,C_R}} \right) \right]$$

$$W_i = \frac{\kappa_i}{\tau} \cdot \frac{1}{E_i} \tag{93}$$

7.2 Computation of F_i and G_i

The contextual impairment levels $L_i^{(420)}$ and $L_i^{(CIF)}$ of frame *f* for CD420 and CDSIF are computed as:

$$L_i^{(420)} = \frac{1}{\gamma} \cdot \sum_{j=1}^{12} W_{i,j} \; L_{i,j}^{(420)} \tag{94}$$

$$L_i^{(CIF)} = \frac{1}{\gamma} \cdot \sum_{j=1}^{12} W_{i,j} \; L_{i,j}^{(CIF)} \tag{95}$$

where γ is a factor restricted into [1/2, 2], which is computed based on the vector distances, D_j, between the spatial and temporal attributes (see § 5), (S_j, T_j) and $(\overline{S}_j, \overline{T}_j)$, of the input scene and each database scene, respectively.

$$D_j = \left(S - \overline{S}_j\right)^2 + (T - \overline{T}_j)^2 \tag{96}$$

$$\begin{aligned} w_j &= \frac{D_j^{-1}}{\sum_{k=1}^{12} D_k^{-1}} \\ a &= \sum_{j=1}^{12} w_j \left[\frac{\overline{S}_j \cdot \overline{T}_j}{2} + (1-\overline{T}_j^2) \cdot \left(1 - \frac{\overline{S}_j^2}{2} \right) \right] \\ b &= \frac{ST}{2} + (1-T^2) \cdot \left(1 - \frac{S^2}{2} \right) \\ \gamma &= 1 + a - b \end{aligned} \tag{97}$$

The parameters F_i and G_i are finally obtained by solving the equation system below:

$$L_i^{(420)} = 100 / \left[1 + \left(\frac{F_i}{m_i^{(420)}} \right)^{G_i} \right] \tag{98}$$

$$L_i^{(CIF)} = 100 / \left[1 + \left(\frac{F_i}{m_i^{(CIF)}} \right)^{G_i} \right] \tag{99}$$

8 References

GONZALEZ, R. C. and WOODS, R. E. [1992] *Digital Image Processing*. Addison-Wesley.

ISO/IEC[1992] Standard ISO/IEC 11172 – Information technology – Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s.

TRUCCO, E. and VERRI, A. [1998] *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall.

Annex 4a

Objective results in VQEG-Phase II tests

TABLE 10

625/60 raw objective data matrix

SRC	HRC									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1		0.6343	0.5083	0.287		0.2461		0.1951		0.1548
2		0.5483	0.5966	0.3649		0.3185		0.2668		0.1597
3		0.5998	0.6299	0.4551		0.3927		0.3428		0.2553
4		0.6055	0.8159	0.5684		0.5397		0.4158		0.309
5		0.6483	0.7268	0.4358		0.418		0.2874		0.1898
6		0.6146	0.4908	0.3671		0.3139		0.2562		0.2107
7				0.5865		0.5536			0.4841	0.3917
8				0.5023		0.457			0.3949	0.3158
9				0.4563		0.3927			0.3399	0.2667
10				0.7036		0.6511			0.6025	0.5083
11	0.8124				0.6374		0.3205			0.3221
12	0.7015				0.547		0.4997			0.3922
13	0.709	0.5098					0.4199			0.3298

TABLE 11

525/60 raw objective data matrix

SRC	HRC													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	0.5472	0.3698	0.3429	0.1918										
2	0.5075	0.226	0.1028	0.0789										
3	0.3549	0.127	0.058	0.0339										
4					0.6062	0.419	0.36	0.3108						
5					0.4444	0.2957	0.2152	0.1635						
6					0.6098[(1)]	0.3462	0.2546	0.1967						
7					0.2404	0.135	0.0864	0.0609						
8									0.8666	0.7554	0.6944	0.7048	0.6685	0.494
9									0.8896	0.7134	0.6204	0.6504	0.6246	0.2326
10									0.8776	0.6419	0.4788	0.6392	0.6237	0.1571
11									0.8623	0.7207	0.5719	0.5619	0.5796	0.3012
12									0.8262	0.6193	0.5139	0.5391	0.4946	0.1992
13									0.8223	0.5609	0.3454	0.437	0.4246	0.215

(1) The SRC = 6, HRC = 5 value was taken out of the analysis because it exceeded the temporal registration requirements of the VQEG test plan.

Annex 5

Model 4

This Annex provides a full functional description of the NTIA VQM and its associated calibration techniques.

The calibration algorithms described in this Annex are sufficient to ensure proper operation of the NTIA video quality estimator. In general, these algorithms have a spatial registration accuracy of plus or minus 1/2 pixel and a temporal registration accuracy of plus or minus one interlaced field.

CONTENTS

1 Introduction

This Annex provides a complete technical description of the NTIA General Model and its associated calibration techniques (e.g. estimation and correction of spatial registration, temporal registration, and gain/offset errors). The General Model is proponent H in the VQEG Phase II Full Reference Television tests. The General Model was designed to be a general purpose VQM for video systems that span a very wide range of quality and bit rates. Extensive subjective and objective tests were conducted to verify the performance of the General Model before it was submitted to the VQEG Phase II test. While the VQEG Phase II tests only evaluated the performance of the General Model on MPEG-2 and ITU-T Recommendation H.263 video systems, the General Model should work well for many other types of coding and transmission systems.

The calibration algorithms described in this Annex are sufficient to ensure proper operation of the video quality estimator. In general, these algorithms have a spatial registration accuracy of plus or minus 1/2 pixel and a temporal registration accuracy of plus or minus one interlaced field.

The General Model and its associated automatic calibration techniques have been completely implemented in user friendly software. This software is available to all interested parties via a no-cost evaluation license agreement (see www.its.bldrdoc.gov/n3/video/vqmsoftware.htm for more information).

2 Normative reference

Recommendation ITU-R BT.601 – Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios.

3 Definitions

4:2:2 – A Y, C_b, C_r image sampling format where chrominance planes (C_b and C_r) are sampled horizontally at half the luminance (Y) plane's sampling rate. See Recommendation ITU-R BT.601 (see Section 2).

Absolute temporal information (ATI): A feature derived from the absolute value of temporal information images that are computed as the difference between successive frames in a video clip. ATI quantifies the amount of motion in a video scene. See § 7.5 for the precise mathematical definition.

Big *YUV*: The binary file format used for storing clips that have been sampled according to Recommendation ITU-R BT.601. In the Big *YUV* format, all the video frames for a scene are stored in one large binary file, where each individual frame conforms to Recommendation ITU-R BT.601 sampling. The Y represents the luminance channel information, the U represents the blue colour difference channel (i.e. C_B in Recommendation ITU-R BT.601), and the V represents the red colour difference channel (i.e. C_R in Recommendation ITU-R BT.601). The pixel ordering in the binary file is the same as that specified in SMPTE 125M [SMPTE, 1995a]. The full specification of the Big *YUV* file format is given in Section 5 and software routines for reading and displaying Big *YUV* files are given in [Pinson and Wolf, 2002].

Clip: Digital representation of a scene that is stored on computer media.

Clip VQM: The VQM of a single clip of processed video.

Chrominance (C, C_B, C_R): The portion of the video signal that predominantly carries the colour information (C), perhaps separated further into a blue colour difference signal (C_B) and a red colour difference signal (C_R).

Codec: Abbreviation for a coder/decoder or compressor/decompressor.

Common intermediate format (CIF): A video sampling structure used for video teleconferencing where the luminance channel is sampled at 352 pixels by 288 lines (see ITU-T Recommendation H.261 – Video codec for audiovisual services at $p \times 64$ kbit/s.).

Feature: A quantity of information associated with, or extracted from, a spatial-temporal sub-region of a video stream (either an original video stream or a processed video stream).

Field: One half of a frame, containing all of the odd or even lines.

Frame: One complete television picture.

Frames per second (FPS): The number of original frames per second transmitted by the video system under test. For instance, an NTSC video system transmits approximately 30 fps.

Gain: A multiplicative scaling factor applied by the hypothetical reference circuit (HRC) to all pixels of an individual image plane (e.g. luminance, chrominance). Gain of the luminance signal is commonly known as contrast.

General Model: The video quality model, or VQM, that is the subject of this Annex 5. The General Model was submitted to the phase II tests performed by the Video Quality Experts Group (VQEG). The VQEG Phase II final report describes the performance of the General Model (see proponent H[1]).

H.261: Abbreviation for ITU-T Recommendation H.261.

Hypothetical reference circuit (HRC): A video system under test such as a codec or digital video transmission system.

Input video: Video before being processed or distorted by an HRC (see Fig. 19). Input video may also be referred to as original video.

Institute for Radio Engineers (IRE) unit: A unit of voltage commonly used for measuring video signals. One IRE is equivalent to 1/140 of a volt.

International Telecommunication Union (ITU): An international organization within the United Nations System where governments and the private sector coordinate global telecommunications networks and services. The ITU includes the Radiocommunication Sector (ITU-R) and the Telecommunication Standardization Sector (ITU-T).

Luminance (*Y*): The portion of the video signal that predominantly carries the luminance information (i.e. the black and white part of the picture).

Mean opinion score (MOS): The average subjective quality judgment assigned by a panel of viewers to a processed video clip.

Moving Picture Experts Group (MPEG): A working group of ISO/IEC in charge of the development of standards for coded representation of digital audio and video (e.g. MPEG-1, MPEG-2, MPEG-4).

National Television Systems Committee (NTSC): The 525-line analogue colour video composite system [SMTPE, 1999].

Offset or level offset: An additive factor applied by the HRC to all pixels of an individual image plane (e.g. luminance, chrominance). Offset of the luminance signal is commonly known as brightness.

Original region of interest (OROI): A region of interest (ROI) extracted from the original video, specified in rectangle coordinates.

Original video: Video before being processed or distorted by an HRC (see Fig. 19). Original video may also be referred to as input video since this is the video input to the digital video transmission system.

Original valid region (OVR): The valid region of an original video clip, specified in rectangle coordinates.

Output video: Video that has been processed or distorted by an HRC (see Fig. 19). Output video may also be referred to as processed video.

Over-scan: The portion of the video that is not normally visible on a standard television monitor.

Phase-alternate line (PAL): The 625-line analogue colour video composite system.

Parameter: A measure of video distortion that is the result of comparing two parallel streams of features, one stream from the original video and the corresponding stream from the processed video.

Processed region of interest (PROI): A region of interest (ROI) extracted from the processed video and corrected for spatial shifts of the HRC, specified in rectangle coordinates.

Processed video: Video that has been processed or distorted by an HRC (see Fig. 19). Processed video may also be referred to as output video since this is the video output from the digital video transmission system.

Processed valid region (PVR): The valid region of a processed video clip from an HRC, specified in rectangle coordinates. The PVR is always referenced to the original video so it is necessary to correct for any spatial shifts of the video by the HRC before computing PVR. Thus, PVR is always contained within the OVR. The region between the PVR and the OVR is that portion of the video that was blanked or corrupted by the HRC.

Production aperture: The image lattice that represents the maximum possible image extent in a given standard. The production aperture represents the desirable extent for image acquisition, generation, and processing, prior to blanking. For Recommendation ITU-R BT.601 sampled video, the production aperture is 720 pixels × 486 lines for 525-line systems and 720 pixels × 576 lines for 625-line systems [SMPTE, 1995b].

Quarter common intermediate format (QCIF): A video sampling structure used for video teleconferencing where the luminance channel is sampled at 176 pixels by 144 lines (see ITU-T Recommendation H.261).

Recommendation ITU-R BT.601: A common 8-bit video sampling standard (see § 2) that samples the luminance (Y) channel at 13.5 MHz, and the blue and red colour difference channels (C_B and C_R) at 6.75 MHz. See § 5 for more information.

Rectangle coordinates: A rectangular shaped image sub-region that is completely contained within the production aperture that is specified by four coordinates (top, left, bottom, right). Numbering starts from zero so that the (top, left) corner of the sampled image is (0,0). See § 5.3.

Reduced-reference: A video quality measurement methodology that utilizes low bandwidth features extracted from the original or processed video streams, as opposed to using full-reference video that requires complete knowledge of the original and processed video streams (ITU-T Recommendation J.143 – User requirements for objective perceptual video quality measurements in digital cable television). Reduced-reference methodologies have advantages for end-to-end in-service quality monitoring since the reduced-reference information is easily transmitted over ubiquitous telecommunication networks.

Reframing: The process of reordering two consecutively sampled interlaced fields of processed video into a frame of video. Reframing is necessary when HRCs do not preserve standard interlace field types (e.g. an NTSC field type one is output as an NTSC field type two and vice versa). See § 6.1.2.

Region of interest (ROI): An image lattice (specified in rectangle coordinates) that is used to denote a particular sub-region of a field or frame of video. Also see SROI.

Scene: A sequence of video frames.

Spatial information (SI): A feature based on statistics that are extracted from the spatial gradients (i.e. edges) of an image or video scene. ITU-T Recommendation P.910 – Subjective video quality assessment methods for multimedia applications, provides a definition of SI based on statistics extracted from 3 × 3 Sobel-filtered images [Jain, 1989] while § 7.2 of this Annex provides a definition of SI based on statistics extracted from much larger 13 × 13 edge-filtered images (see Fig. 29).

Spatial region of interest (SROI): The specific image lattice (specified in rectangle coordinates) that is used to calculate the VQM of a video clip. The SROI is a rectangular subset that lies completely inside the processed valid region. For Recommendation ITU-R BT.601 sampled video, the recommended SROI is 672 pixels × 448 lines for 525-line systems and 672 pixels × 544 lines for 625-line systems, centred within the production aperture. This recommended SROI corresponds to approximately the portion of the video picture that is visible on a monitor, excluding the over-scan area. Also see ROI.

Spatial registration: The process that is used to estimate and correct for spatial shifts of the processed video sequence with respect to the original video sequence.

Spatial-temporal (S-T) sub-region: A block of image pixels in an original or processed video stream that includes a vertical extent (number of rows), a horizontal extent (number of columns), and a time extent (number of frames). See Fig. 27.

Society of Motion Picture and Television Engineers (SMPTE): An industry-leading society for the motion picture and television industries devoted to advancing theory and application in motion imaging, including film, television, video, computer imaging, and telecommunications. The industry relies on SMPTE to generate standards, engineering guidelines, and recommended practices to be followed by respective field professionals.

Temporal information (TI): A feature based on statistics that are extracted from the temporal gradients (i.e. motion) of a video scene. (See ITU-T Recommendation P.910) and § 7.5 of this Annex all provide definitions of TI based on statistics extracted from simple frame differences.

Temporal region of interest (TROI): The specific time segment, sequence, or subset of frames that is used to calculate a clip's VQM. The TROI is a contiguous segment of frames that lies completely inside the temporal valid region. The maximum possible TROI is the fully registered time segment and contains all temporally registered frames within the TVR. If reframing is required, the processed clip is always reframed, not the original clip.

Temporal registration: The process that is used to estimate and correct for the temporal shift (i.e. video delay) of the processed video sequence with respect to the original video sequence (see § 6.4.1).

Temporal valid region (TVR): The maximum time segment, sequence, or subset of video frames that may be used for calibration and VQM calculation. Frames outside of this time segment will always be considered invalid.

Uncertainty (*U*): The estimated error (plus or minus) in the temporal registration after allowance is made for the best guess of the HRC video delay. See § 6.4.

Valid region (VR): The rectangular portion of an image lattice (specified in rectangle coordinates) that is not blanked or corrupted due to processing. The valid region is a subset of the production aperture of the video standard and includes only those image pixels that contain picture information that has not been blanked or corrupted. See original valid region and processed valid region.

Video Quality Experts Group (VQEG): A group of international video quality experts that conduct validation tests for objective video performance metrics. Results from VQEG are forwarded to the ITU and may be used as the basis for international video quality measurement recommendations.

Video quality metric, model, or measurement (VQM): An overall measure of video impairment (see Clip VQM, General Model). VQM is reported as a single number and has a nominal output range from zero to one, where zero is no perceived impairment and one is maximum perceived impairment.

4 Overview of the VQM computation

This Annex provides a complete description of the General Model and its associated calibration algorithms. These automated objective measurement algorithms provide close approximations to the overall quality impressions, or mean opinion scores, of digital video impairments that have been graded by panels of viewers (see Recommendation ITU-R BT.500). Figure 19 gives an overview diagram of the processes required to compute the General VQM. These processes include sampling of the original and processed video streams (§ 5), calibration of the original and processed video streams (§ 6), extraction of perception-based features (§ 7), computation of video quality parameters (§ 8), and calculation of the General Model (§ 9). The General Model tracks the perceptual changes in quality due to distortions in any component of the digital video transmission system (e.g. encoder, digital channel, decoder).

The method of measurement documented herein utilizes high bandwidth reduced-reference parameters (see ITU-T Recommendation J.143). These reduced reference parameters utilize features extracted from spatial-temporal (S-T) regions of the video sequence (see § 7.1.1). Hence, the method of measurement presented here may also be used to perform in-service video quality monitoring in situations were an ancillary data channel is available to transmit the extracted features between the source and destination ends of an HRC as shown in Fig. 19.

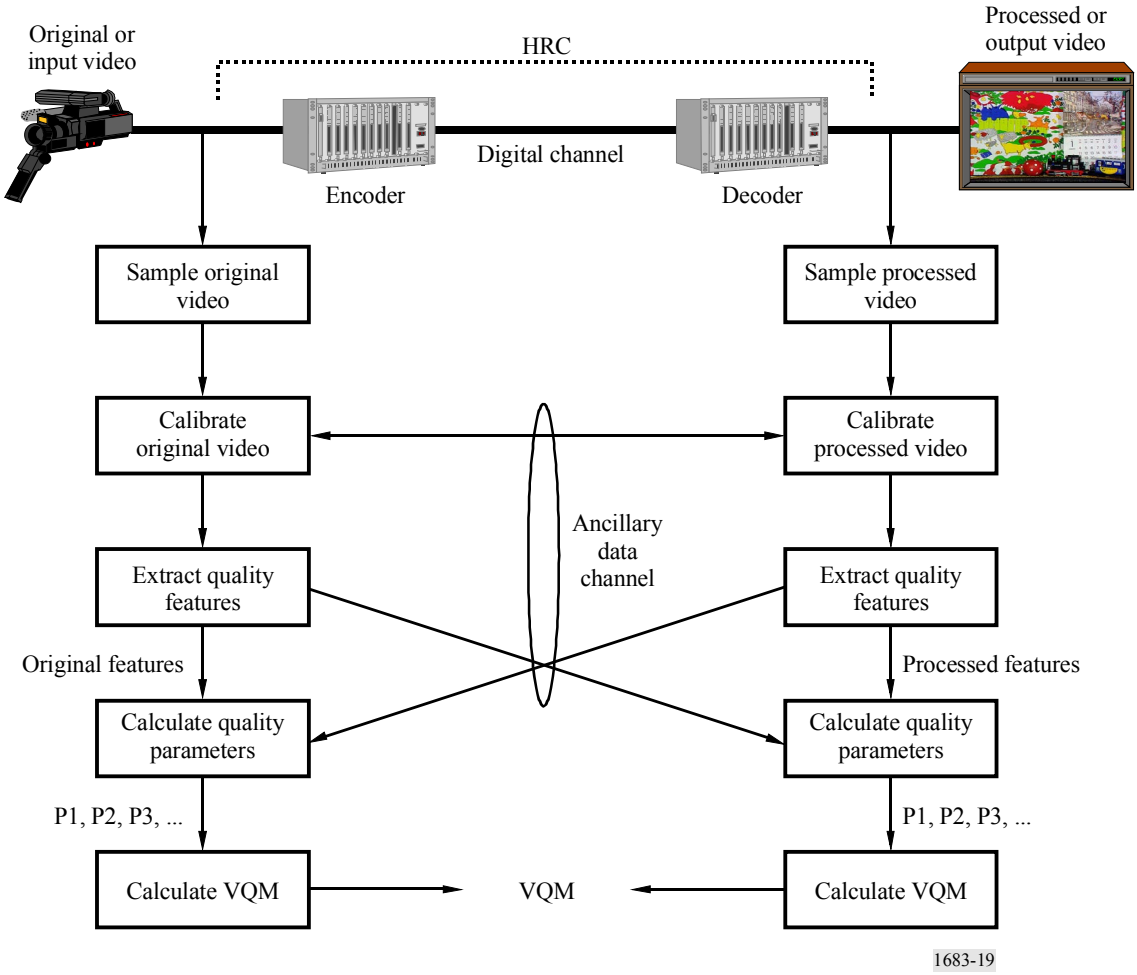
5 Sampling

The computer-based algorithms in this Annex assume that the original and processed video streams are available as digital representations stored on computer media (referred to as a clip in this Annex). If the video is analogue format, one of the most widely used digital sampling standards is Recommendation ITU-R BT.601 (§ 2). Composite video such as NTSC and PAL must first be converted into component video that contains the following three signals: luminance (Y), blue colour difference, C_B, and red colour difference, C_R. Recommendation ITU-R BT.601 sampling is also commonly known as 4:2:2 sampling since the Y channel is sampled at full rate while the C_B and C_R channels are sampled at half rate. Recommendation ITU-R BT.601 specifies a 13.5 MHz sample rate that produces 720 Y samples per video line. Since there are 486 lines that contain picture information in the 525-line NTSC standard, the complete Recommendation ITU-R BT.601 sampled Y video frame will be 720 pixels by 486 lines. Likewise, when 625-line PAL video is sampled according to Recommendation ITU-R BT.601, the Y video frame will contain 720 pixels by 576 lines. If 8 bits are used to uniformly sample the Y signal, Recommendation ITU-R BT.601 specifies that reference black (i.e. 7.5 IRE units) be sampled as a “16” and reference white (i.e. 100 IRE units) be sampled as a “235.” Thus, a working margin is available for video signals

that exceed the reference black and white levels before they are clipped by the analogue to digital converter. The chrominance channels (C_B and C_R) are each sampled at 6.75 MHz such that the first pair of chrominance samples (C_B, C_R) is associated with the first Y luminance sample, the second pair of chrominance samples is associated with the third luminance sample, and so forth. Since the chrominance channels are bipolar, zero signal is sampled as a "128".

FIGURE 19

Steps required to compute VQM

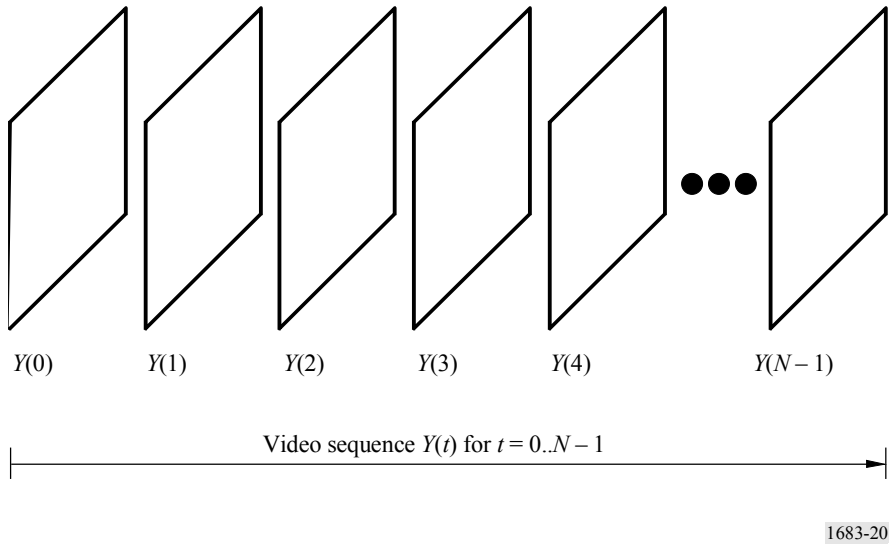


5.1 Temporal indexing of original and processed video files

A luminance video frame that results from Recommendation ITU-R BT.601 sampling will be denoted as $Y(t)$. The variable t is being used here as an index for addressing the sampled frames within the original and processed Big *YUV* files; it does not denote actual time. If the Big *YUV* file contains N frames, as shown in Fig. 20, $t = 0$ denotes the first frame that was sampled and $t = (N - 1)$ denotes the last frame that was sampled.

FIGURE 20

Temporal indexing of frames in big *YUV* files



All the algorithms are written and described from the viewpoint of operation on sampled file pairs: one original video sequence and an associated processed video sequence. To avoid confusion, both files are assumed to be the same length. Furthermore, an initial assumption will be made that the first frame of the original file aligns temporally to the first frame of the processed file, within plus or minus some temporal uncertainty.

For real-time, in-service implementations, this balanced uncertainty presumption can be replaced with a one-sided uncertainty. Causality constrains the range of temporal uncertainty. For example, a processed frame occurring at time $t = n$ must come from original frames occurring at or before time $t = n$.

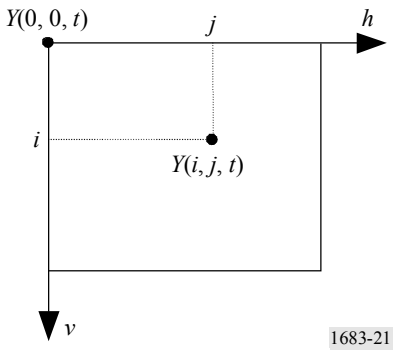
The above assumption regarding original and processed video files (i.e. that the first frames align) is equivalent to selecting the best guess for the temporal delay of the HRC shown in Fig. 19. Therefore, the uncertainty that remains in the video delay estimate will be denoted as plus or minus U.

5.2 Spatial indexing of original and processed video frames

The coordinate system used for the sampled luminance frames is shown in Fig. 21. The horizontal and vertical coordinates of the upper left corner of the luminance frames are defined to be ($v = 0$, $h = 0$), where the horizontal axis, h, coordinate values increase to the right and the vertical axis, v coordinate values increase down. Horizontal axis coordinates range from 0 to one less than the number of pixels in a line. Vertical axis coordinates range from 0 to one less than the number of lines in the image, which will be specified in frame lines for progressive systems and either field lines or frame lines for interlace systems. The amplitude of a sampled pixel in $Y(t)$ at row i (i.e. $v = i$), column j (i.e. $h = j$), and time t is denoted as $Y(i, j, t)$.

FIGURE 21

Coordinate system used for sampled luminance *Y* frames



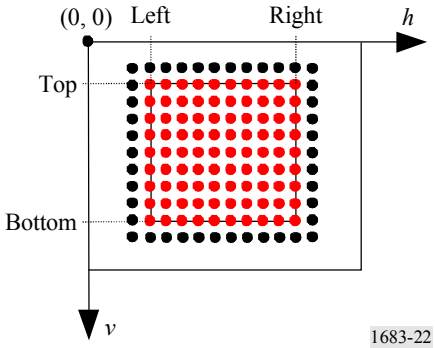
A clip of video sampled according to Recommendation ITU-R BT.601 is stored in "Big *YUV*" file format, where the *Y* denotes the Recommendation ITU-R BT.601 luminance information, the *U* denotes the blue colour-difference information (i.e. C_B in Recommendation ITU-R BT.601), and the *V* denotes the red colour-difference information (i.e. C_R in Recommendation ITU-R BT.601). In the Big *YUV* file format, all the frames are stored sequentially in one large continuous binary file. The image pixels are stored sequentially by video scan line as bytes in the following order: C_{B0}, Y_0, C_{R0}, Y_1, C_{B2}, Y_2, C_{R2}, Y_3, etc., where the numerical subscript denotes the pixel number (pixel replication or interpolation must be used to find the C_B and C_R chrominance samples for Y_1, Y_3, …). This byte ordering is equivalent to that specified in SMPTE 125M [SMPTE, 1995a].

5.3 Specifying rectangular sub-regions

Rectangular subregions of a sampled image are used to control the computation of VQM. For instance, VQM may be computed over the valid region of the sampled image or over a user-specified spatial region of interest that is smaller than the valid region. Specification of rectangular sub-regions will use rectangle coordinates defined by the four quantities top, left, bottom, and right. Figure 22 illustrates the specification of a rectangular subregion for a single frame of sampled video. The red image pixels are included in the subregion but the black image pixels are excluded. In the calculation of VQM, an image is often divided into a large number of smaller subregions that abut. The rectangular sub-region definition used in Fig. 22 defines the grid used to display these abutted sub-regions and the math used to extract features from each abutted subregion.

FIGURE 22

Rectangle coordinates for specifying image subregions



5.4 Considerations for video sequences longer than 10 s

The video quality measurements in this Annex were based upon subjective test results that utilized 8 to 10 s video clips. When working with longer video sequences, the sequence should be divided into shorter video segments, where each segment is assumed to have its own calibration and quality attributes. Dividing the video stream into overlapping segments and processing each segment independently is one method for emulating continuous quality assessments for long video sequences using the VQM techniques presented herein.

6 Calibration

Four steps are required to properly calibrate the sampled video in preparation for feature extraction. These steps are:

Step 1: spatial registration estimation and correction,

Step 2: valid region estimation to limit the extraction of features to those pixels that contain picture information,

Step 3: gain and level offset estimation and correction (commonly known as contrast and brightness), and

Step 4: temporal registration estimation and correction.

Step 2 must be performed on both the original and processed video streams. Steps 1, 3, and 4 must be performed on the processed video stream. Normally, the spatial registration, gain, and level offset are constant for a given video system and hence these quantities only need to be calculated once. However, it is common for the valid region and temporal registration to change depending upon scene content. For instance, full screen and letterbox scenes will have different valid regions, and videoconferencing systems often have variable video delays that depend upon scene content (e.g. talking head versus sports action). In addition to the calibration techniques presented here, the reader may also want to examine (see ITU-T Recommendation P.931 – Multimedia communications delay, synchronization and frame rate measurement). for alternate spatial and temporal registration methods.

Calibrating prior to feature extraction means that VQM will not be sensitive to horizontal and vertical shifts of the image, temporal shifts of the video stream that result from non-zero video delays, and changes in image contrast and brightness that fall within the dynamic range of the video sampling unit. While these calibration quantities can have a significant impact on the overall perceived quality (e.g. low contrast images from a video system with a gain of 0.3), the philosophy taken here is to report calibration information separately from VQM. Spatial shifts, valid regions, gains, and offsets can normally be adjusted using good engineering practices, while temporal delays provide important quality information when evaluating two-way or interactive video systems.

All of the video quality features and parameters (§ 7 and 8) assume that only one video delay will be removed to temporally register the processed video sequence (i.e. constant video delay). Some video systems or HRCs delay individual processed frames by different amounts (i.e. variable video delay). For the purposes of this Annex, all video systems are treated as having a constant video delay. Variations from this delay are considered degradations that are measured by the features and parameters. This approach appears to yield higher correlations to subjective score than video quality measurements based on processed video sequences where variable video delay has been removed. When working with long video sequences (see § 5.4), the sequence should be divided into shorter video segments, where each segment has its own constant video delay. This allows for some delay variation as a function of time. A more continuous estimation of delay variations may be obtained by dividing the sequence into overlapping time segments.

If the HRC being tested also spatially scales the picture or changes its size (e.g. zoom), then an additional step to estimate and remove this spatial scaling would have to be included in the calibration process. Spatial scaling is beyond the scope of this Annex.

6.1 Spatial registration

6.1.1 Overview

The spatial registration process determines the horizontal and vertical spatial shift of the processed video relative to the original video. A positive horizontal shift is associated with a processed image that has been moved to the right by that number of pixels. A positive vertical shift is associated with a processed image that has been moved down that number of lines. Thus, spatial registration of interlace video results in three numbers: the horizontal shift in pixels, the vertical field one shift in field lines, and the vertical field two shift in field lines. Spatial registration of progressive video results in two numbers: the horizontal shift and the vertical shift in frame lines. The accuracy of the spatial registration algorithm is to the nearest pixel for horizontal shifts and to the nearest line for vertical shifts. After the spatial registration has been calculated, the spatial shift is removed from the processed video stream (e.g. a processed image that was shifted down is shifted back up). For interlace video, this may include reframing of the processed video stream as implied by comparison of the vertical field one and two shifts.

When operating on interlace video, all operations will consider video from each field separately; when operating on progressive video, all operations will consider the entire video frame simultaneously. For simplicity, the spatial registration algorithm will first be entirely described for interlace video, this being the more complicated case. The modifications needed to operate on progressive video are identified in § 6.1.6.

Spatial registration must be determined before PVR, gain and level offset, and temporal registration. Specifically, each of those quantities must be computed by comparing original and processed video content that has been spatially registered. If the processed video stream were spatially shifted with respect to the original video stream and this spatial shift were not corrected, then these estimates would be corrupted because they would be based on dissimilar video content. Unfortunately, spatial registration cannot be correctly determined unless the PVR, gain and level offset, and temporal registration are also known. The interdependence of these quantities produces a "chicken or egg" measurement problem. Calculation of the spatial registration for one processed field requires that one know the PVR, gain and level offset, and the closest matching original field. However, one cannot determine these quantities until the spatial shift is found. A full exhaustive search over all variables would require a tremendous number of computations if there were wide uncertainties in the above quantities.

The solution presented here performs an iterative search to find the closest matching original field for each processed field. This search includes iteratively updating estimates for PVR, gain and level offset, and temporal registration. For some processed fields, however, the spatial registration algorithm could fail. Usually, when the spatial registration is incorrectly estimated for a processed field, the ambiguity is due to characteristics of the scene. Consider, for example, a digitally created interlace scene containing a pan to the left. Because the pan was computer generated, this scene could have a horizontal pan of exactly one pixel every field. From the spatial registration search algorithm’s point of view, it would be impossible to differentiate between the correct spatial registration computed using the matching original field, and a two pixel horizontal shift computed using the field that occurs two fields prior to the matching original field. For another example, consider an image consisting entirely of digitally perfect black and white vertical lines. Because the image contains no horizontal lines, the vertical shift is entirely ambiguous. Because the pattern of vertical lines repeats, the horizontal shift is ambiguous, two or more horizontal shifts being equally likely.

Therefore, the iterative search algorithm should be applied to a sequence of processed fields. The individual estimates of spatial shifts from multiple processed fields can then be used to produce a more robust estimate. Spatial shift estimates from multiple sequences or scenes may be further combined to produce an even more robust estimate for the HRC being tested; assuming that the spatial shift is constant for all scenes passing through the HRC.

6.1.2 Interlace issues

Vertical spatial registration of interlaced video is a greater challenge than progressive video, since the spatial registration process must differentiate between field one and field two. There are three vertical shift conditions that must be differentiated to obtain the correct vertical shift registration for interlaced systems: vertical field one equals vertical field two, vertical field one is one less than vertical field two, and everything else.

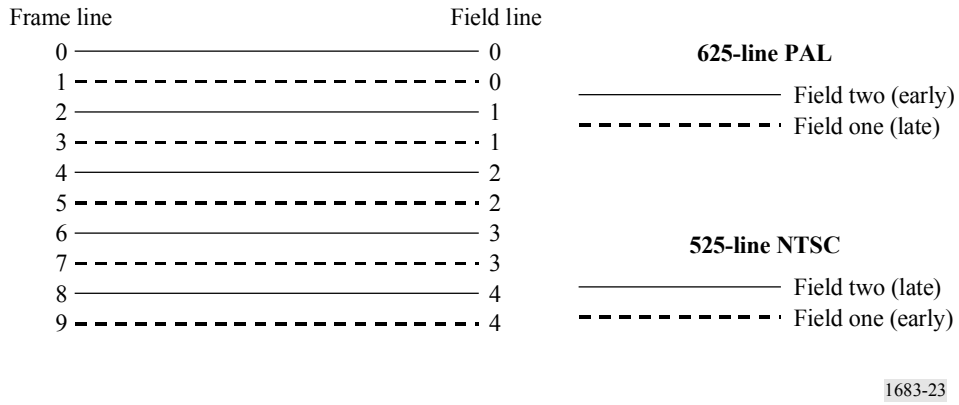
Some HRCs shift field one and field two identically, yielding a vertical field one shift that is equal to the vertical field two shift. For HRCs that do not repeat fields or frames (i.e. HRCs that transmit the full frame rate of the video standard), this condition means that what was a field one in the original video stream is also a field one in the processed video stream, and what was a field two in the original is also a field two in the processed.

Other HRCs reframe the video, shifting the sampled frame by an odd number of frame lines. What used to be field one of the original becomes field two of the processed, and what used to be field two of the original becomes the next frame's field one. Visually, the displayed video appears correct since the human cannot perceive a one-line frame shift of the video.

As shown in Fig. 23, field one starts with frame line one, and contains all odd-numbered frame lines. Field two starts with frame line zero (topmost frame line), and contains all even-numbered frame lines. For NTSC, field one occurs earlier in time and field two occurs later in time. For PAL, field two occurs earlier in time and field one occurs later in time.

FIGURE 23

Diagram depicting interlaced fields and frame/field line numbering scheme



Reframing occurs when either the earlier field moves into the later field and the later field moves into the earlier field of the next frame (one-field delay), or when the later field moves into the earlier field and the earlier field of the next frame moves into the later field of the current frame (one-field advance). For example, when NTSC original field two is moved into the next NTSC frame's field one, the top line of the field moves from original field-two frame line 0 to processed field-one frame line 1. In field line numbering, the top line stays in field line 0, so processed field one has a zero vertical shift (since vertical shifts are measured for each field using field lines).

When original NTSC field one is moved to that frame's field two, the top line of the field moves from original field one, frame line 1 to processed field two, frame line 2. In field line numbering, the top line moves from field line 0 to field line 1, so processed field two has a one field line vertical shift. The general rule for both NTSC and PAL is that when the field-two vertical shift (in field lines) is one greater than the field-one vertical shift (in field lines), reframing has occurred.

If the field-two vertical shift is not equal to or one more than the field-one vertical shift, the HRC has corrupted the proper spatial sampling of the two interlaced fields of video and the resulting video will appear to "bob" up and down. Such an impairment is both obvious and annoying to the viewer, and hence seldom occurs in practice since the HRC designer discovers and corrects the error. Therefore, most of the time, spatial registration simplifies into two common patterns. In systems that do not reframe, field-one vertical shift equals field-two vertical shift; and in systems that reframe, field-one vertical shift plus one equals field-two vertical shift.

Additionally, notice that spatial registration includes some temporal registration information, specifically whether the video has been reframed or not. The temporal registration process may or may not be able to detect reframing, but even if it can, reframing is inherent to the spatial registration process. Therefore, spatial registration must be able to determine whether the processed field being examined best aligns with an original field one or field two. The spatial registration for each field can only be correctly computed when the processed field is compared to the original field that created it. Aside from the reframing issue, use of the wrong original field (field one versus field two) can produce spatial registration inaccuracies due to the inherent differences in the spatial content of the two interlaced fields.

6.1.3 Required inputs to the spatial registration algorithm

This section gives a list of the input variables that are required by the spatial registration algorithm. These inputs specify items such as the range of spatial shifts and temporal fields over which to search. If these ranges are overly generous, the speed of convergence of the iterative search algorithm used to find the spatial shift may be slow and the probability of false spatial registration for scenes with repetitive content is increased (e.g. someone waving their hand). Conversely, if these ranges are too restrictive, the search algorithm will encounter, and slowly extend, the search range boundaries with successive iterations. While this built-in search intelligence is useful if the user mis-guesses the search uncertainties by a small amount, the undesirable side effect is to dramatically increase run time when the user mis-guesses by a large amount. Alternatively, the search algorithm may fail to find the correct spatial shift in this case.

6.1.3.1 Expected range of spatial shifts

The expected range of spatial shifts for 525-line and 625-line video sampled according to Recommendation ITU-R BT.601 lies between ±20 pixels horizontally and ±12 field lines vertically. This range of expected shifts has been determined empirically by processing video data from hundreds of HRCs. The expected range of spatial shifts for video sampled according to other formats smaller than Recommendation ITU-R BT.601 (e.g. CIF), is presumed to be half of that observed for 525-line and 625-line systems. This search algorithm should operate correctly, albeit a bit slower, when the processed field has spatial shifts that lie outside of the expected range of spatial shifts. This is because the search algorithm will expand the search beyond the expected range of spatial shifts when warranted. Excursions exceeding 50% of the expected range, however, may report a failure to find the correct spatial registration.

6.1.3.2 Temporal uncertainty

The user must also specify the temporal registration uncertainty, i.e. the range of original fields to examine for each processed field. This temporal uncertainty is expressed as a number of frames before and after the default temporal registration. If the original and processed video sequences are

stored as files, then a reasonable default temporal registration is to assume that the first frames in each file align. The temporal uncertainty that is specified should be large enough to include the actual temporal registration. An uncertainty of plus or minus one second (30 frames for 525-line NTSC video; 25 frames for 625-line PAL video) should be sufficient for most video systems. HRCs with long video delays may require a larger temporal uncertainty. The search algorithm may examine temporal registrations outside of the specified uncertainty range when warranted (e.g. when the farthest original field is chosen as the best temporal registration).

6.1.3.3 PVR guess

The PVR guess specifies the portion of the processed image that has not been blanked or corrupted due to processing, presuming no spatial shift has occurred (since the spatial shift has not yet been measured). Although the PVR guess could be determined empirically, a user-specified PVR guess that excludes the over-scan is a good choice. In most cases this will eliminate invalid video from being used in the spatial registration algorithm. For 525-line/NTSC video sampled according to Recommendation ITU-R BT.601, the over-scan covers approximately 18 frame lines at the top and bottom of the frame, and 22 pixels at the left and right sides of the frame. For 625-line/PAL video sampled according to Recommendation ITU-R BT.601, the over-scan covers approximately 14 frame lines at the top and bottom of the frame, and 22 pixels at the left and right sides of the frame. When using other image sizes (e.g. CIF), a reasonable default PVR for these image sizes should be selected.

6.1.4 Sub-algorithms used by the spatial registration algorithm

The spatial registration algorithm makes use of a number of sub-algorithms, including estimation of gain and level offset, and the formula used to determine the closest matching original field for a given processed field. These sub-algorithms have been designed to be computationally efficient, since they must be performed many times by the iterative search algorithm.

6.1.4.1 ROI used by all calculations

All field comparisons made by the algorithm will be between spatially shifted versions of a ROI extracted from the processed video (to compensate for the spatial shifts introduced by the HRC) and the corresponding ROI extracted from the original video. The spatially shifted ROI from the processed video will be denoted as PROI (i.e. processed ROI) and the corresponding ROI from the original video will be denoted as OROI (original ROI). The rectangle coordinates that specify OROI are fixed throughout the algorithm and are chosen to give the largest possible OROI that meets both of the following requirements:

– The OROI must correspond to a PROI that lies within the PVR for all possible spatial shifts that will be examined.

– The OROI is centred within the original image.

6.1.4.2 Gain and level offset

The following algorithm is used to estimate the gain of the processed video. The processed field being examined is shift-corrected using the current estimate for spatial shift. After this shift-correction, a PROI is selected that corresponds to the fixed OROI determined in § 6.1.4.1. Next, the standard deviation of the luminance (Y) pixels from this PROI and the standard deviation of the luminance pixels (Y) from the OROI are calculated. Gain is then estimated as the standard deviation of PROI pixels divided by the standard deviation of OROI pixels.

The reliability of this gain estimate improves as the algorithm iterates toward the correct spatial and temporal shift. A gain of 1.0 (i.e. no gain correction) may be used during the first several iteration cycles. The above gain calculation is sensitive to impairments in the processed video such as

blurring. However, for the purposes of spatial registration, this gain estimate is appropriate because it makes the processed video look as much like the original video as possible. To remove gain from the processed field, each luminance pixel in the processed field is divided by the gain.

There is no need to determine or correct for level offset, since the spatial registration algorithm's search criteria are unaffected by level offsets (see § 6.1.4.3).

6.1.4.3 Formulae used to compare PROI with OROI

After correcting the PROI for gain[3] (§ 6.1.4.2), the standard deviation of the (OROI-PROI) difference image is used to choose between two or more spatial shifts or temporal shifts. The gain estimate from the previous best match is used to correct the PROI gain. To search among several spatial shifts (with temporal shift held constant), compute the standard deviation of the (OROI-PROI) difference image for several PROI generated using different spatial shifts. For a given processed field, the combination of spatial and temporal shifts that produce the smallest standard deviation (i.e. most cancellation with the original) is chosen as the best match.

6.1.5 Spatial registration using arbitrary scenes

Spatial registration of a processed field from a scene must examine a plurality of original fields and spatial shifts since both the temporal shift (i.e. video delay) and the spatial shift are unknown. As a result, the search algorithm is complex and computationally intense. Furthermore, the scene content is arbitrary, and so the algorithm may find an incorrect spatial registration (see § 6.1.1). Therefore, the prudent course is to compute the spatial registration of several processed fields from several different scenes that have all been passed through the same HRC, and combine the results into one robust estimate of spatial shift. A single HRC should have one constant spatial registration. If not, these time varying spatial shifts would be perceived as an impairment (e.g. the video would bounce up and down or from side to side). This section describes the spatial registration algorithm from the bottom up, in that the core components of the algorithm are described first, and then their application for spatial registering scenes and HRCs is described.

6.1.5.1 Best original field match in time

When spatially registering, using scene content, the algorithm must find the original field that most closely matches the current processed field. Unfortunately, that original field may not actually exist. For example, a processed field may contain part of two different original fields since it may have been interpolated from other processed fields. The current estimate of the best original field match (i.e. that original field that most closely matches the current processed field) is kept at all stages of the search algorithm.

An initial assumption is made that the first field of the processed Big *YUV* file aligns with the first field of the original Big *YUV* file, within plus or minus some temporal uncertainty in frames (denoted here as *U*). For each processed field that is examined by the algorithm, there must be a buffer of *U* original frames before and after this field. Thus, the algorithm starts examining processed fields that are *U* frames into the file, and examines every frequencyth frame thereafter (denoted here as *F*), stopping *U* frames before the end of the file.

The final search results from the previous processed field (gain, vertical and horizontal shift, temporal shift) are used to initialize the search for the current processed field. The best original field

---

[3] Gain compensation can sometimes be omitted to decrease the computational complexity. However, omission of gain correction is only recommended during early stages of the iterative search algorithm, where the goal is to find the approximate spatial registration (e.g. see § 6.1.5.2 and 6.1.5.3).

match to the current processed field is computed assuming a constant video delay. For example, if processed field *N* was found to best align with original field *M* in the Big *YUV* files, then processed field *N* + *F* would be assumed to be best aligned to original field *M* + *F* at the start of the search.

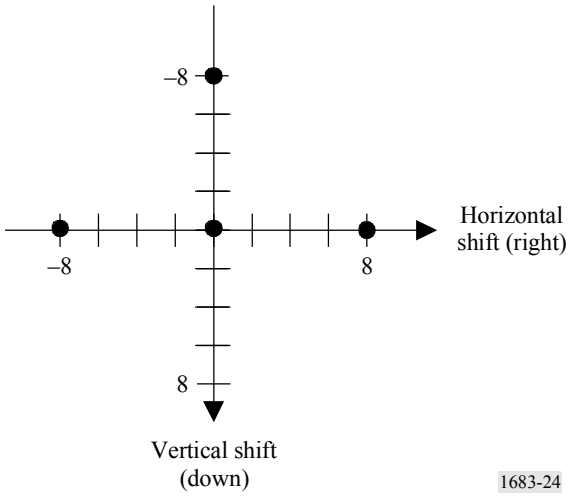
6.1.5.2 Broad search for the temporal shift

A full search of all possible spatial shifts across the entire temporal uncertainty for each processed field would require a large number of computations. Instead, a multi-step search is used, where the first step is a broad search, over a very limited set of spatial shifts, whose purpose is to get close to the correct matching original field.

For the selected processed frame, this broad search examines field one of this frame (see Fig. 23) and considers only those original fields of field type one that are spaced two frames apart (i.e. four fields apart) across the entire range of plus and minus the temporal registration uncertainty. The broad search considers the following four spatial shifts of the processed video: no shift, eight pixels to the left, eight pixels to the right, and eight field lines up (see Fig. 24). In Fig. 24, positive shifts mean the processed video is shifted down and to the right with respect to the original video. The "eight field lines down" shift is not considered because empirical observations have revealed that very few video systems move the video picture down. The previous best estimate for spatial shift (i.e. from a previously processed field) is also included as a fifth possible shift when it is available. The closest matching original field to the selected processed field is found using the comparison technique described in § 6.1.4.3. The temporal shift implied by the closest matching original field becomes the starting point for the next step of the algorithm, a broad search for the spatial shift (§ 6.1.5.3). According to the coordinate system in Fig. 21, a positive temporal shift means that the processed video has been shifted in the positive time direction (i.e. the processed video is delayed with respect to the original video). With respect to the original and processed Big *YUV* files, a positive field shift thus means that fields must be discarded from the beginning of the processed Big *YUV* file while a negative field shift means that fields must be discarded from the beginning of the original Big *YUV* file.

FIGURE 24

Spatial shifts considered by the broad search for the temporal shift



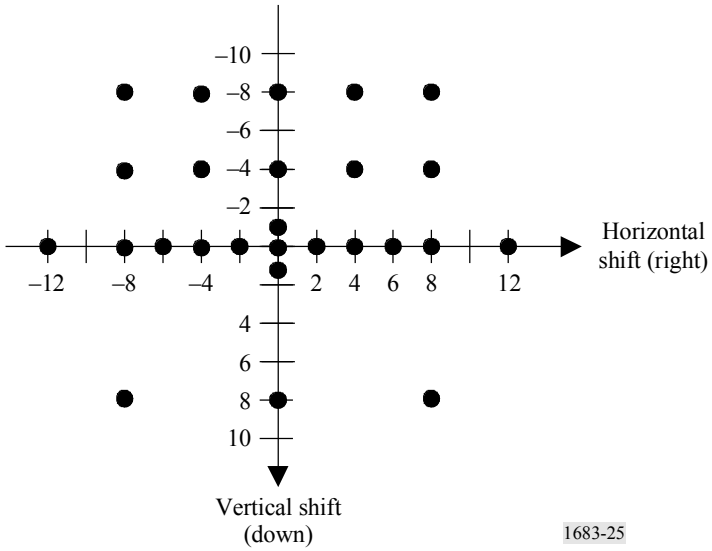
6.1.5.3 Broad search for the spatial shift

Using the temporal registration found by the broad search for temporal shift (see § 6.1.5.2), a broad search for the correct spatial shift is now performed using a more limited range of original fields. The range of original fields that are considered for this search include the best matching original

field of field type one (from § 6.1.5.2) and the four next closest original fields that are also of field type one (field type ones from the 2 frames before and after the best matching original field). The broad search for spatial shift covers the range of spatial shifts given in Fig. 25. Notice that fewer downward shifts are considered (as in § 6.1.5.2), since these are less likely to be encountered in practice. The set of spatial shifts and original fields is searched using the comparison technique described in § 6.1.4.3. The resulting best temporal and spatial shifts now become the improved estimates for the next step of the algorithm given in § 6.1.5.4.

FIGURE 25

Spatial shifts considered by the broad search for the spatial shift

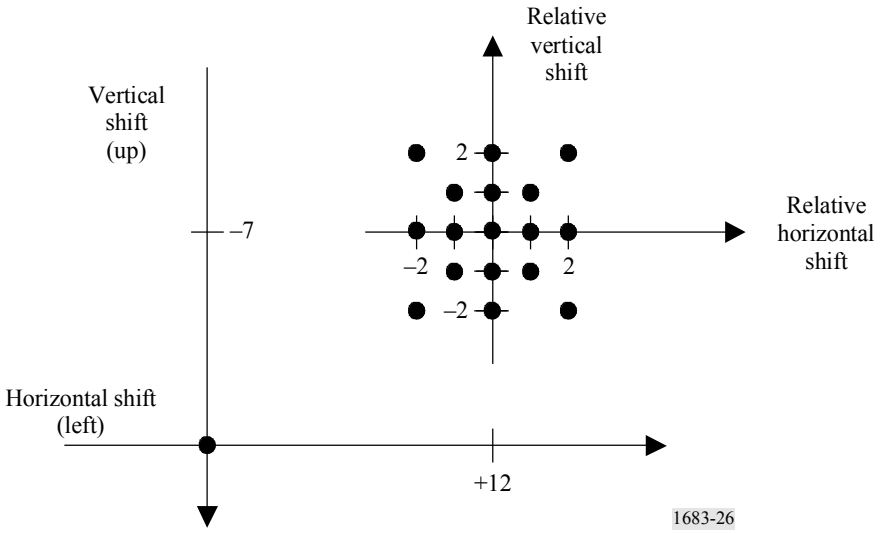


6.1.5.4 Fine search for the spatial-temporal shift

The fine search includes a much smaller set of shifts centred around the current spatial registration estimate and just five fields centred around the current best matching original field. Thus, if the best matching original field were a field type one, the search would include three field type ones and the two field type twos. The spatial shifts that are considered include the current shift estimate, all eight shifts that are within one pixel or one line of the current estimate, eight shifts that are two pixels or two lines from the current shift estimate, and the zero shift condition (see Fig. 26). In the example shown in Fig. 26, the current spatial shift estimate for the processed video is a shift of 7 field lines up and 12 pixels to the right of the original video. The set of spatial shifts shown in Fig. 26 form a near-complete local search of the spatial registrations near the current spatial registration estimate. The zero shift condition is included as a safety check that helps prevent the algorithm from wandering and converging to a local minimum. The set of spatial shifts and original fields is thoroughly searched using the comparison technique described in § 6.1.4.3. The resulting best temporal and spatial shifts now become the improved estimates for the next step of the algorithm given in § 6.1.5.5.

FIGURE 26

Spatial shifts considered by the fine search for the spatial shift



6.1.5.5 Repeated fine searches

Iteration through the fine search of § 6.1.5.4 will move the current estimate for spatial shift a little closer to either the actual spatial shift or (more rarely) a false minimum. Likewise, one iteration through the fine search will move the current estimate for the best-aligned original field either a little closer to the actual best-aligned original field or (more rarely) a little closer to a false minimum. Thus, each fine search will move these estimates closer to a stable value. Because fine searches examine a very limited area spatially and temporally, they must be performed repetitively to assure that convergence has been reached. When gain compensation is being used, the processed field's gain is estimated anew between each fine search (see § 6.1.4.2).

Repeated fine searches are performed on the processed field (see § 6.1.5.4) until the best spatial shift and the original field associated with that spatial shift remain unchanged from one search to the next. Repeated fine searches are stopped if the algorithm is alternating between two spatial shifts (e.g. a horizontal shift 3 and then a horizontal shift 4, with everything else remaining the same). This alternation is indicated when the current best estimate for spatial shift and the original field associated with that spatial shift, are identical to those found two iterations ago.

Sometimes the repeated search algorithm fails to converge. If the algorithm fails to converge within some requested maximum number of iterations, the iterative search algorithm is terminated and a "failure to find shift" condition is reported for that processed field. This special case does not normally pose a problem because multiple processed fields are examined for each scene (§ 6.1.5.6) and multiple scenes are examined for each HRC (§ 6.1.5.7).

6.1.5.6 Algorithm for one scene

An initial baseline (i.e. starting) estimate for vertical shift, horizontal shift, and temporal registration is computed without any gain compensation as follows. The first temporal uncertainty, U, processed frames in the Big *YUV* file are skipped. A broad search for the temporal shift is performed on the next processed field of field type one (see § 6.1.5.2). Notice that this broad search will search the first $U \cdot 2 + 1$ frames of the original video sequence for a field type one that best aligns. Then, a broad search for the spatial shift is performed centred on this best-aligned original field (see § 6.1.5.3). Next perform up to five fine spatial-temporal searches to fine-tune the spatial and temporal estimates (see § 6.1.5.4 and 6.1.5.5). If these repeated fine searches fail to find a stable

result, discard this processed field from consideration. Repeat the above procedure every frequencyth, *F*, frame until an original field of field type one is found that produces stable results. The baseline estimate will be updated periodically, as described below.

The spatial shift estimates are calculated for both field types of a frame in the processed Big *YUV* file as follows. Using the baseline estimate as a starting point, perform up to three repeated fine searches on the first processed field of field type one. If the baseline estimate is correct or very nearly correct, the repeated fine searches will yield a stable result. If so, the spatial shift and temporal delay for that processed field are stored in an array that is dedicated to storing the field one results. If a stable result is not found, most likely the spatial shift is correct but the temporal shift estimate is off (i.e. the current estimate of temporal shift is more than two frames away from the true temporal shift). So a broad search for the temporal shift is conducted that includes the current best estimate of spatial shift. This broad search will normally correct the temporal delay estimate. When the broad search for the temporal shift completes, its output is used as the starting point, and up to five repeated fine searches are performed. If this second repeated fine search fails to find a stable result, then report a failed spatial registration for this frame (i.e. both field type one and field type two). If a stable result is found from this second search, then the spatial shift and temporal delay for that field are stored in the field one array. Also, the spatial shift and temporal delay used as the starting point for the next processed field of field type one are updated (i.e. for the first processed field, the baseline results are used and after that, the last stable result is used). After the spatial shift has been estimated for the first processed field of field type one, the spatial shift for the first processed field of field type two is estimated. Using the field one spatial results as the starting point, the same steps are used to find the field-two spatial shift (i.e. the three fine searches, and if needed a broad search for the temporal shift followed by five repeated fine searches). If a stable result is found for field two, store the vertical and horizontal shift for field two in a different array that is dedicated to storing field-two results.

The procedure described in the above paragraph is applied to estimate the spatial shift of both field types of each frequencyth, *F*, frame in the Big *YUV* file that contains the processed video. The first temporal uncertainty, *U*, processed frames in the Big *YUV* file are skipped. This sequence of estimates is then used to compute robust estimates of the spatial shift for each field type for the scene being examined. The vertical field-one shift results from each frame are sorted, and the 50th percentile retained as the overall vertical field-one shift. Likewise, the vertical field-two shift results from each frame are sorted, and the 50th percentile retained as the overall vertical field-two shift. The horizontal field-one shift results from each frame are sorted, and the 50th percentile retained as the overall horizontal shift. Any difference between field-one and field-two horizontal shift is most likely due to a sub-pixel horizontal shift (e.g. a horizontal shift of 0.5 pixels). Sub-pixel horizontal shifts will produce estimates that include both of the two closest shifts. Using the 50th percentile point allows the most likely horizontal shift to be chosen, which produces a spatial registration accuracy that is good to the nearest 0.5 pixels[4].

6.1.5.7 Algorithm for one HRC

If several scenes have been passed through the same HRC, the spatial registration results for each scene should be identical. Thus, filtering results obtained from multiple scenes can increase the robustness and accuracy of the spatial shift measurements. The overall HRC spatial registration results can then be used to compensate all of the processed video for that HRC.

---

[4] Spatial registration to the nearest 0.5 pixels is sufficient for the video quality measurements described in this Annex. Sub-pixel spatial registration techniques are beyond the scope of this Annex.

6.1.5.8 Comments on algorithm

Some video scenes are simply not well suited for estimating spatial registration. The described algorithm will sometimes locate a false minimum. Other times, the algorithm will wander between multiple solutions and never reach a stable result. For these reasons, it is advisable to examine multiple images within the same scene and to median filter (i.e. sort results from low to high and select the 50th percentile point) these results across different scenes. The spatial registration by scenes algorithm is an heuristic algorithm that utilizes patterns of spatial shifts that have been observed from a sampling of video systems. These assumptions may be incorrect for some systems, causing the algorithm to find an incorrect spatial shift. However, failure of the algorithm tends to produce spatial shifts that are inconsistent from frame to frame and from scene to scene (i.e. when the algorithm fails, it normally produces a scattering of results). When the algorithm outputs the same or very similar spatial shifts for each scene, a high degree of confidence is indicated. When the individual field results for a scene wander, a low degree of confidence is indicated.

6.1.6 Spatial registration of progressive video

Spatial registration of progressive video follows the same steps as the interlace algorithms, with minor modifications. Where the interlace algorithms operate on field one and field two separately, the progressive algorithm operates on frames. Thus, all mentions of field two are ignored and, with the exception of the fine searches, the range of vertical shifts is doubled.

The modification of the vertical shift range is most important for the broad spatial shift. When doing a broad search for spatial shift (see § 6.1.5.3) the numbers on the vertical axis in Fig. 25 must be doubled (e.g. +8 becoming +16 and –4 becoming –8)[5]. In addition, for progressive CIF and QCIF images, the horizontal and vertical broad spatial search ranges are halved due to the smaller shifts that are typically encountered with these image sizes. For example, using CIF images in Fig. 25, the horizontal axis would stretch from –6 to +6 pixels and the vertical axis would stretch from –8 to +8 frame lines.

The temporal search range, being stated in frames, is largely unchanged. For the broad temporal search in § 6.1.5.2, instead of matching one processed field one to every second original field one, the progressive algorithm compares one processed frame to every second original frame. For the colourbar algorithm, the search examines spatial shifts for one processed frame and one original frame (i.e. no temporal searching).

The only step requiring more complicated changes is the fine search from § 6.1.5.4. Here, the vertical shifts remain unchanged, lying between –2 frame lines and +2 frame lines. Thus, the vertical axis of Fig. 26 is interpreted as referring to frame lines. The temporal extent of this fine search may be set to five original frames centred on the current aligned original frame, instead of the three original frames otherwise implied. A five-frame search extent may improve the speed and efficiency of the fine search when compared to the interlace version of the algorithm, since progressive HRCs are more likely to contain varying video delay than non-zero spatial shifts.

When considering the algorithmic changes for progressive video systems, many of the spatial shift search parameters can be modified without harming the integrity of the algorithm. As an example, consider spatial shifts other than zero pixels and zero lines for the broad temporal search. The spatial shift at zero pixels horizontally and 8 field lines vertically for interlace video could be moved to 16 frame lines for progressive video, as recommended above, or placed at 8 frame lines,

---

[5] In one possible exception to this doubling, the spatial shift associated with zero pixels horizontally and plus or minus one field line vertically could be left at plus or minus one frame line vertically. Spatial shifts very close to (zero, zero) are commonly encountered.

under the assumption that progressive video sequences are unlikely to contain 16 frame lines of vertical shift. Likewise, spatial shift at zero lines vertically and 8 pixels horizontally could be moved to 9 or 10 pixels horizontally without any detrimental effects. As another example, the exact number of repeated fine searches performed could be increased or decreased for specific applications. The exact values recommended here are significantly less important than the actual structure of the search algorithm.

6.2 Valid region

NTSC (525-line) and PAL (625-line) video sampled according to Recommendation ITU-R BT.601 may have a border of pixels and lines that do not contain a valid picture. The original video from the camera may only fill a portion of the Recommendation ITU-R BT.601 frame. A digital video system that utilizes compression may further reduce the area of the picture in order to save transmission bits. If the non-transmitted pixels and lines occur in the over-scan area of the television picture, the typical end-user should not notice the missing lines and pixels. If these non-transmitted pixels and lines exceed the over-scan area, the viewer may notice a black border around the picture, since the system will normally insert black into this non-transmitted picture area. Video systems (particularly those that perform low-pass filtering) may exhibit a ramping up from the black border to the picture area. These transitional effects most often occur at the left and right sides of the image but can also occur at the top or bottom. Occasionally, the processed video may also contain several lines of corrupted video at the top or bottom of the picture that may not be visible to the viewer (e.g. VHS tape recorders corrupt several lines at the bottom of the picture in the over-scan area). To prevent non-picture areas from influencing the VQM measurements, these areas should be excluded from the VQM measurement. The automated valid region algorithm presented here estimates the valid region of the original and processed video streams so that subsequent computations do not consider corrupted lines at the top and bottom of the Recommendation ITU-R BT.601 frame, black border pixels, or transitional effects where the black border meets the picture area.

6.2.1 Core valid region algorithm

This section describes the core valid region algorithm that is applied to a single original or processed image. This algorithm requires three input arguments: an image, a maximum valid region, and the current valid region estimate.

– *Image*: the core algorithm uses the Recommendation ITU-R BT.601 luminance image of a single video frame. When measuring the valid region of a processed video sequence, any spatial shift imposed by the video system must have been removed from the luminance image before applying the core algorithm (see § 6.1).

– *Maximum valid region*: the core algorithm will not consider pixels and lines outside of a maximum valid video region. This provides a mechanism for the user to specify a maximum valid region that is smaller than the entire image area if *a priori* knowledge indicates that the sampled image has corrupted pixels or lines as discussed in § 6.2.

– *Current valid region*: the current valid region is an estimate of the valid region and lies entirely within the maximum valid region. All pixels inside the current valid region are known to contain valid video; pixels outside the current valid region may or may not contain valid video content. Initially, the current valid region is set to the smallest possible area located at the exact centre of the image.

The core algorithm examines the area of video between the maximum valid region and the current valid region. If some of those pixels appear to contain valid video, the current valid region estimate is enlarged. The algorithm will now be described in detail for the left edge of the image.

Step 1: Compute the mean of the left-most column of pixels in the maximum valid region. The left-most column of pixels will be denoted as column J-1 and the mean will be represented by $M_{J\text{-}1}$.

Step 2: Take the mean of the next column of pixels, M_J.

Step 3: Column J is declared invalid video if it is black, ($M_J < 20$) or if the average pixel level of the mean value for successive columns indicates a ramp up from black border to valid picture ($M_J - 2 > M_{J-1}$). If either of these conditions are true, increment J and repeat Steps 2 and 3. Otherwise, go to Step 4.

Step 4: If final column J is within the current valid region, then no new information has been obtained. Otherwise, update the current valid region with J as the left coordinate.

The algorithm for finding the top edge of the image is similar to that given above for the left edge. For the bottom and right edges, J is decremented instead of incremented; otherwise the algorithm is the same. The values produced for top, left, bottom, and right indicate the last valid pixel or line.

The stopping conditions identified in Step 3 can be fooled by scene content. For example, an image that contains genuine black at the left side (i.e. black that is part of the scene) will cause the core algorithm to conclude that the left-most valid column of video is farther toward the middle of the image than it ought to be. For that reason, the core algorithm is applied to multiple images from a video sequence, thereby increasing the accuracy of the valid region estimate.

6.2.2 Applying the core valid region algorithm to a video sequence

6.2.2.1 Original video

The core algorithm is first applied to the original sequence of images. For NTSC video sampled according to Recommendation ITU-R BT.601 (see § 5), the recommended setting for the maximum valid region is top = 6, left = 6, bottom = 482, right = 714. For PAL video sampled according to Recommendation ITU-R BT.601, the recommended setting for the maximum valid region is top = 6, left = 16, bottom = 570, right = 704. The core algorithm is run on the first image in the video sequence, and every frequencyth image thereafter. For example, if the specified frequency were 15, the core algorithm would examine sequence image numbers 0, 15, 30, 45, and so forth. When all images in the sequence have been examined, the current valid region will contain the largest valid area implied by any examined image in the video sequence. Pixels and lines between this final current valid region and the maximum valid region are considered to contain either black or a transitional ramp up from black.

The final valid region must contain an even number of lines and an even number of pixels. Any odd top or left coordinates are incremented by one. Then, if the region contains an odd number of lines, bottom is decremented; likewise, if the region contains an odd number of pixels (e.g. horizontally), right is decremented. This simplifies colour processing for video sampled in accordance with Recommendation ITU-R BT.601, since the colour channels are sub-sampled by 2 when compared to the luminance channel. Also, each interlaced field of video will contain the same number of video lines. This will assure that spatial-temporal sub-regions (from which features will be extracted) always contain valid video with equal contributions from both interlaced fields. The resulting valid region is returned as the original valid region.

6.2.2.2 Processed video

When computing the valid region of the processed video sequence, the maximum valid region setting for the core algorithm is first set equal to the corresponding original valid region found for that scene. This maximum valid region is then reduced in size by any pixels and lines that are considered invalid due to spatially shift correcting the processed video frames. The core algorithm is then run on the first image in the processed video sequence, and every frequencyth image thereafter (i.e. if frequency = F, use images $Y(0)$, $Y(F)$, $Y(2F)$, $Y(3F)$, and so forth).

After the core algorithm has been applied to the processed video sequence, the valid region found by the core algorithm is reduced inward by a safety margin. The recommended safety margin discards one line off the top and bottom, and five pixels off the left and right. The large left and right safety margins ensure that any ramp up or down from black is excluded from the processed valid region.

The final processed valid region must contain an even number of lines and an even number of pixels. Any odd top or left coordinates are incremented by one. Then, if the region contains an odd number of lines, bottom is decremented; likewise, if the region contains an odd number of pixels (e.g. horizontally), right is decremented. The resulting valid region is returned as the processed valid region.

6.2.3 Comments on valid region algorithm

This automated valid region algorithm will work well to estimate the valid region of most scenes. Due to the nearly infinite possibilities for scene content, the algorithm described herein takes a conservative approach to estimation of the valid region. A manual examination of valid region would quite likely choose a larger region. Conservative valid region estimates are more suitable for an automated video quality measurement system, because discarding a small amount of video will have little impact on the quality estimate and in any case this video usually occurs in the over-scan part of the video. On the other hand, including corrupted video in the video quality calculations may have a large impact on the quality estimate.

This algorithm does not contain sufficient artificial intelligence to distinguish between corrupted pixels and lines at the edge of an image and true scene content. A rule of thumb is used instead, stating that such invalid video generally occurs at the extreme edges of the image. Specification of a conservative user-definable maximum valid video region (i.e. the starting point for the automated algorithm) provides a mechanism to exclude these possibly corrupt image edges from consideration.

When the valid region algorithm is applied to video that is not sampled according to Recommendation ITU-R BT.601 (e.g. the common intermediate format, or CIF, used by ITU-T Recommendation H.261), the recommended setting for maximum valid region when examining the original video is the entire image. In these cases, the sampled video does not normally include any corrupted over-scan, so a maximum valid region smaller than the entire image is unnecessary.

6.3 Gain and offset

6.3.1 Core gain and level offset algorithm

This section explains the method for performing gain and level offset calibration. A prerequisite before applying this algorithm is that the original and processed images be spatially registered (see § 6.1). The original and processed images must also be temporally registered, which will be addressed later in § 6.4. Gain and level offset calibration can be performed on either fields or frames as appropriate.

The method presented here makes the assumption that the Recommendation ITU-R BT.601 Y, C_B, and C_R signals each have an independent gain and level offset. This assumption will in general be sufficient for calibrating component video systems (e.g. Y, R-Y, B-Y). However, in composite or S-video systems, it is possible to have a phase rotation of the chrominance information since the two chrominance components are multiplexed into a complex signal vector with amplitude and phase. The algorithm presented here will not properly calibrate video systems that introduce a phase rotation of the chrominance information (e.g. the hue adjustment on a television set).

– As previously noted, this calibration model assumes that there is no cross coupling between any of the three video components. With this assumption, the core calibration algorithm is applied independently to each of the three channels: Y, C_B, and C_R.

– The valid region of the original and processed image plane is first divided into N sub-regions. For each of the sub-regions, the mean *original* and *processed* values are computed (i.e. mean over space). Next, these *original* and *processed* values are represented as N-element column vectors $\underline{O}$ and $\underline{P}$, respectively:

$$\underline{O}_{N\times 1}=\begin{bmatrix} original_1 \\ . \\ . \\ . \\ original_N \end{bmatrix}, \qquad \underline{P}_{N\times 1}=\begin{bmatrix} processed_1 \\ . \\ . \\ . \\ processed_N \end{bmatrix}$$

Calibration involves computing the gain, g, and level offset, l, according to the following model:

$$\underline{P}=g\underline{O}+l$$

Since there are only two unknowns (i.e. g and l) but N equations (i.e. N sub-regions), we must solve the over-determined system of linear equations given by:

$$\underline{\hat{P}}=A\begin{bmatrix} l \\ g \end{bmatrix}$$

where A is an $N \times 2$ matrix given by $A_{N\times 2}=[\underline{1} \quad \underline{O}]$, and $\underline{1}$ is an N-element column vector of "1s" given by:

$$\underline{1}_{N\times 1}=\begin{bmatrix} 1_1 \\ . \\ . \\ . \\ 1_N \end{bmatrix}$$

$\hat{P}$ is the estimate of the processed samples if the gain and level offset correction were applied to the original samples. The least squares solution to this over-determined problem (provided $N > 2$) is given by:

$$\begin{bmatrix} l \\ g \end{bmatrix}=\left(A^{\mathrm{T}}A\right)^{-1}A^{\mathrm{T}}P$$

where the superscript, T, denotes matrix transpose and the superscript, –1, denotes matrix inverse.

When the core gain and offset algorithm is independently applied to each of the three channels, six estimates result: Y gain, Y offset, C_B gain, C_B offset, C_R gain, and C_R offset.

6.3.2 Using scenes

The basic algorithm given in § 6.3.1 can be applied to original and processed video streams provided they have been spatially and temporally registered. This scene-based technique divides the image into abutting blocks with unknown intensity levels. A sub-region size of 16 lines × 16 pixels is recommended for frames (i.e. 8 lines × 16 pixels for one Y NTSC or PAL field; 8 lines × 8 pixels for C_B and C_R due to sub-sampling of the colour image planes). The mean over space of the [Y, C_B, C_R] samples is computed for each corresponding original and processed sub-region, or block, to form a spatially sub-sampled image. All the selected blocks must lie within the PVR.

6.3.2.1 Registering the processed images

For simplicity, we will assume that the best spatial registration has already been found using one of the techniques presented in § 6.1. Before gain and level offset are estimated, each processed image must also be temporally registered. The original image that best aligns with the processed image must be used for the gain and level offset calculation. If the video delay is variable, this temporal registration must be performed for each processed image. If the video delay is constant for the scene, the temporal registration only needs to be performed once.

To temporally register a processed image, first create the spatially sub-sampled original and processed fields (or frames for progressive video) as specified in § 6.3.2, after correcting for the spatial shift of the processed video. Using the sub-sampled *Y* images, apply the search function given in § 6.1.4.3, except one performs this search using all the original images that are within the temporal registration uncertainty, *U*. Use the best resulting temporal registration for all three image planes, *Y*, C_B, and C_R.

6.3.2.2 Gain and level offset of registered images

An iterative least squares solution with a cost function is used to help minimize the weight of outliers in the fit. This is because outliers are normally due to distortions rather than pure level offset and gain changes, and assigning equal weight to these outliers would distort the fit.

The following algorithm is applied separately to the *N* matching original and processed pixels from each of the three spatially sub-sampled images [*Y*, C_B, C_R].

Step 1: Use the normal least squares solution from § 6.3.1 to generate the initial estimate of the level offset and gain: $\begin{bmatrix} l \\ g \end{bmatrix} = \left(A^{\mathrm{T}} A\right)^{-1} A^{\mathrm{T}} \underline{P}$.

Step 2: Generate an error vector, $\underline{E}$, that is equal to the absolute value of the difference between the true processed samples and the fitted processed samples: $\underline{E} = \left|\underline{P} - \underline{\hat{P}}\right|$.

Step 3: Generate a cost vector, $\underline{C}$, that is the element-by-element reciprocal of the error vector, *E*, plus a small epsilon, ε: $\underline{C} = \frac{1}{\underline{E} + \varepsilon}$. The ε prevents division by zero and sets the relative weight of a point that is on the fitted line versus the weight of a point that is off the fitted line. An ε of 0.1 is recommended.

Step 4: Normalize the cost vector *C* for unity norm (i.e. each element of *C* is divided by the square root of the sum of the squares of all the elements of *C*).

Step 5: Generate the cost vector *C* 2 that is the element-by-element square of the cost vector *C* from Step 4.

Step 6: Generate an $N \times N$ diagonal cost matrix, *C* 2, that contains the cost vector's elements, *C* 2, arranged on the diagonal, with zeros everywhere else.

Step 7: Using the diagonal cost matrix, *C* 2, from Step 6, perform cost-weighted least squares fitting to determine the next estimate of the level offset and gain: $\begin{bmatrix} l \\ g \end{bmatrix} = \left(A^{\mathrm{T}} C^2 A\right)^{-1} A^{\mathrm{T}} C^2 \underline{P}$.

Step 8: Repeat Steps 2 through 7 until the level offset and gain estimates converge to four decimal places.

These steps are applied separately to processed field one and processed field two, to obtain two estimates for g and l. Field one and two must be examined separately, because the temporally registered original fields need not correspond to one frame within the original video sequence. For progressive video, the above steps are applied to the entire processed frame at once.

6.3.2.3 Estimating gain and level offset for a video sequence and HRC

The algorithm described above is applied to multiple matching original and processed field pairs distributed every frequencyth frame throughout the scene (for progressive video, original and processed frame pairs). A median filter is then applied to the six time histories of the level offsets and gains to produce average estimates for the scene.

If several scenes have been passed through the same HRC, the level offset and gain for each scene will be considered to be identical. Thus, filtering results obtained from multiple scenes can increase the robustness and accuracy of the level offset and gain measurements. The overall HRC level offset and gain results can then be used to compensate all of the processed video for that HRC.

6.3.3 Applying gain and level offset corrections

The temporal registration algorithms (see § 6.4) and most quality features (§ 7) will specify that the gain calculated herein should be removed. To remove gain and level offset from the Y plane, apply the following formula to each processed pixel:

$$\text{New } Y(i,j,t) = [Y(i,j,t) - l]/g$$

Gain and level offset correction is not performed on the colour planes (i.e. C_B and C_R). Perceptual chrominance errors are instead captured by the colour metrics. The C_B and C_R image planes may be gain and level offset corrected for display purposes.

6.4 Temporal registration

Modern digital video communication systems typically require several tenths of a second to process and transmit the video from the sending camera onto the receiving display. Excessive video delays impede effective two-way communication. Therefore, objective methods for measuring end-to-end video communications delay are important to end-users for specification and comparison of services and to equipment/service providers to optimize and maintain their product offerings. Video delay can depend upon dynamic attributes of the original scene (e.g. spatial detail, motion) and video system (e.g. bit-rate). For instance, scenes with large amounts of motion can suffer more video delay than scenes with small amounts of motion. Thus, video delay measurements should be made in-service to be truly representative and accurate. Estimates of video delay are required to temporally align the original and processed video features shown in Fig. 19 before making quality measurements.

Some video transmission systems may provide time synchronization information (e.g. original and processed frames may be labelled with some kind of a frame numbering scheme). In general, however, time synchronization between the original and processed video streams must be measured. This section presents a technique for estimating video delay based upon the original and processed video frames. The technique is “frame-based” in that it works by correlating lower resolution images, sub-sampled in space and extracted from the original and processed video streams. This frame-based technique estimates the delay of each frame or field (for interlaced video systems). These individual estimates are combined to estimate the average delay of the video sequence.

6.4.1 Frame-based algorithm for estimating variable temporal delays between original and processed video sequences

This section describes a frame-based temporal registration algorithm. To reduce the influence of distortions on temporal registration, images are spatially sub-sampled and normalized to have unit variance. This algorithm temporally registers each processed image separately, locating the most similar original image. Some of these individual temporal registration measurements may be incorrect but those errors will tend to be randomly distributed. When delay measurements from a series of images are combined by means of a voting scheme, the overall estimate for the average delay of a video sequence becomes quite accurate. This temporal registration algorithm does not use still and nearly motionless portions of the scene, since the original images are nearly identical to each other.

6.4.1.1 Constants used by the algorithm

BELOW_WARN: Threshold used when examining correlations for deciding if a secondary correlation maximum is sufficiently large so as to indicate ambiguous temporal registration. A BELOW_WARN of 0.9 is recommended.

BLOCK_SIZE: The sub-sampling factor. Specified in frame lines vertically and pixels horizontally. A BLOCK_SIZE of 16 is recommended.

DELTA: Secondary maximums in the correlation curve that are within DELTA of the maximum (best) correlation are ignored. A DELTA of 4 is recommended.

HFW: Half of the filter width for the filter used to smooth the histogram of frame-by-frame temporal registration values. A HFW of 3 is recommended.

STILL_THRESHOLD: A threshold that is used to detect still video scenes (frame-based temporal registration cannot be used on still video scenes). A STILL_THRESHOLD of 0.002 is recommended.

6.4.1.2 Inputs to the algorithm

A sequence of N original video luminance images: $\boldsymbol{Y_O}(t)$, $0 \le t < N$ [6].

A sequence of N processed video luminance images: $\boldsymbol{Y_P}(t)$, $0 \le t < N$.

Gain and offset correction factors for the processed luminance images.

Spatial registration information: horizontal shift and vertical shift. For interlace video, the vertical shift for each field determines whether the processed video requires reframing.

Valid region of the processed video sequence (i.e. PVR).

Uncertainty (U): a number indicating the accuracy of the initial temporal registration. The initial temporal registration assumption is that the true temporal registration for $\boldsymbol{Y_P}(t)$ is within plus or minus (U – HFW) of $\boldsymbol{Y_O}(t)$, for all $0 \le t < N$.

6.4.1.3 Frames versus fields

The frame-based temporal registration algorithm works for both progressive and interlace video. If the video sequence is progressive, the algorithm aligns frames. If the video sequence is interlaced, the algorithm aligns fields. When aligning interlaced video sequences, either frame or reframed

---

[6] When interlace video requires reframing, the lengths of the original and processed video sequences must be reduced by one to accommodate the reframing. This will reduce the length of the file by one video frame from N as specified in Fig. 20.

alignments are considered but not both. When frame alignments are considered, field one of the processed video is compared to field one of the original video, and field two of the processed video is compared to field two of the original video. When reframed alignments are considered, field one of the processed video is compared to field two of the original video, and field two of the processed video is compared to field one of the original video. The spatial registration values that are input to the algorithm determine whether frame or reframe alignments are considered. The presence of reframing is detected by examining the vertical spatial registration for each field. If the field one vertical shift equals the field two vertical shift, then the processed video is not reframed; only frame alignments are considered. If the field two vertical shift is one greater than the field one vertical shift, only reframe alignments are considered. All other combinations of vertical shifts indicate problems that should be fixed prior to temporal registration.

6.4.1.4 Description of the algorithm

Step 1: Calibrate the video sequences

Correct the processed video sequence, $\boldsymbol{Y_P}(t)$, using the spatial registration and gain-offset information given as inputs to the algorithm.

Step 2: Select the sub-region of video to be used

The sub-region of interest to be used by the algorithm must be a multiple of the BLOCK_SIZE and must fit within the PVR. The largest sub-region that meets these two requirements and is closest to the centre of the image should be selected. All further processing will be limited to video within this selected sub-region of interest.

Step 3: Spatially sub-sample the original and processed images

Spatially sub-sample the region of interest of $\boldsymbol{Y_O}(t)$ and $\boldsymbol{Y_P}(t)$ by a factor of BLOCK_SIZE by computing the mean of each block. For progressive video frames, the sub-sampling will be BLOCK_SIZE horizontally and vertically, while for interlace video fields, the sub-sampling will be BLOCK_SIZE horizontally and BLOCK_SIZE/2 vertically. For example, sub-sampling a progressive video sequence by a BLOCK_SIZE of 16 will take the mean of each 16 pixel by 16 frame line block, while sub-sampling an interlace video sequence by a BLOCK_SIZE of 16 will take the mean of each 16 pixel by 8 field line block. This sub-sampling reduces the impact of impairments on the temporal registration process.

Step 4: Normalize the sub-sampled images

Normalize each sub-sampled image by the standard deviation of that image. Skip this normalization for any image where the standard deviation is less than one (e.g. images containing a flat field of colour)[7]. This normalization will minimize the influence of fluctuations in individual image contrast and energy from influencing the temporal registration results. After this step, the original video and processed video sequences will be denoted as $\boldsymbol{S_O}(t)$ and $\boldsymbol{S_P}(t)$, respectively, to denote that the images have been sub-sampled and normalized.

Step 5: Compare processed images to original images

Compare each processed image, $\boldsymbol{S_P}(t)$, with the original images $\boldsymbol{S_O}(t + d)$, where the valid values of d are: $(-U \leq d \leq +U)$ and the valid values of t are: $(U \leq t < N - U)$. For processed image t and original image $t + d$, these comparisons will be denoted as $\boldsymbol{C}_{td}$ and are computed as the standard deviation over space of the image formed by subtracting processed image t from original image

---

[7] Normalization is skipped when the standard deviation is less than one to prevent amplification of noise and to prevent the possibility of dividing by zero for images that contain a flat or uniform intensity level.

$t + d$: $\boldsymbol{C}_{td} = std_{space}(\boldsymbol{S_O}(t + d) - \boldsymbol{S_P}(t))$. These comparisons, $\boldsymbol{C}_{td}$, correlate the t-th processed image with each original image that is within the registration uncertainty. Lower values of $\boldsymbol{C}_{td}$ indicate that the processed image looks more like the original image since more of the image variance is cancelled. The range for t, $U \leq t < N - U$, covers all processed images for which original images are available for the entire range of temporal registration uncertainty.

Step 6: Perform an overall check for still video

To determine if there is sufficient motion in the video sequence, average $\boldsymbol{C}_{td}$ over time index t for each d:

$$A_d = \frac{1}{N - 2 * U} \cdot \sum_{t=U}^{N-U-1} C_{td} \tag{100}$$

This summation (100) includes the range of processed video images t for which the full uncertainty of original images is available. A_d contains one value for each temporal registration delay d being considered. If (maximum(A_d) – minimum(A_d) < STILL_THRESHOLD), then the scene contains insufficient motion for frame-based temporal registration. The entire scene is still or nearly still. The correlation results from the different video delays are then so similar that any differentiation will be due to random chance rather than reliable measurements. If a still video sequence is detected, the user is given a warning to that effect and the algorithm exits at this point.

Step 7: Temporally register each processed image

For each processed image t ($U \leq t < N - U$), find the d within the temporal uncertainty ($-U \leq d \leq +U$) that minimizes $\boldsymbol{C}_{td}$. In other words, for each processed image t, find $d_{min}(t)$ such that $\boldsymbol{C}_{t\, dmin(t)} \leq \boldsymbol{C}_{td}$, for all d. The best temporal registration of processed image t is given by $d_{min}(t)$. Most of the time, the temporal registration indicated for individual images will be correct or very close to correct. The temporal registration will be incorrect for some images due to various reasons (image distortion, errors, noise, insufficient motion, etc.).

Step 8: Perform a stillness check on each processed image

If for a given processed image t and all values of d ($-U \leq d \leq U$), maximum($\boldsymbol{C}_{td}$) – minimum($\boldsymbol{C}_{td}$) < STILL_THRESHOLD, then $d_{min}(t)$ is undefined for this processed image t. Specifically, there is insufficient motion around image t for frame-based temporal registration to work properly.

Step 9: Form a histogram of all defined temporal registrations

Compute a histogram using all the defined values of $d_{min}(t)$ with $2*U + 1$ bins where each bin represents a different video delay (i.e. from $-U$ to $+U$). Values of $d_{min}(t)$ that are undefined (e.g. still images) are left out of the histogram calculation. This histogram, denoted by H_d, is the histogram of temporal delays for all the processed images that had sufficient motion to perform valid temporal registration. Each bin in the histogram contains the number of processed images with that video delay d, where d can take values from $-U$ to $+U$.

Step 10: Form a smoothed histogram

Histogram H_d is smoothed by convolving it with a low pass filter of length 2*HFW + 1 and defined at index k as:

$$F_k = \frac{0.5 + 0.5 * \cos[\pi * (k - \mathrm{HFW})/(1 + \mathrm{HFW})]}{\sum_{i=0}^{2*\mathrm{HFW}} \{0.5 + 0.5 * \cos[\pi * (i - \mathrm{HFW})/(1 + \mathrm{HFW})]\}} \qquad \text{for } 0 \leq k \leq 2 * \mathrm{HFW} \tag{101}$$

When considering the smoothed histogram (101) SH_d that results from this step, the HFW bins on each end of SH_d are treated as undefined. This restricts the video delays that can be estimated to plus or minus (UNCERTAINTY-HFW). Smoothing of the histogram increases the robustness of the video delay estimates.

Step 11:Examine the histogram information

From the original histogram, H_d, and the smoothed histogram, SH_d, the following three values are determined:

max_H_value: The maximum value of H_d.

max_SH_offset: The offset d that maximizes SH_d.

max_SH_value: The maximum value of SH_d (e.g. at d = max_SH_offset).

Next, the following two checks are performed:

– Was U large enough? Recall that the first and last HFW bins of H_d are missing from SH_d. Examine the values of H_d in these bins. If (H_d > max_H_value ∗ BELOW_WARN), then the temporal registration uncertainty is too small. The algorithm must be re-run with a larger U. The values of d to check are ($-U \le d < -U$ + HFW) and (U – HFW < $d \le U$).

– Does SH_d have one well-defined delay? Examine SH_d, except within DELTA of max_SH_offset. If (SH_d > max_SH_value ∗ BELOW_WARN) for any video delay d where ($-U \le d <$ max_SH_offset − DELTA) or (max_SH_offset + DELTA < $d \le U$), then temporal registration is ambiguous.

If the above two checks pass, then the video delay given by max_SH_offset is chosen as the best average temporal registration for the scene.

6.4.1.5 Observations and conclusions

The frame-based video delay measurement algorithm uses sub-sampled original and processed video sequences. This algorithm is suitable for aligning video in a fully automated out-of-service environment, prior to performing video quality measurements. The frame-based video delay measurement algorithm estimates the temporal registration for each image, forms histograms of those individual estimates, and then uses the most commonly indicated delay as the overall video delay, or temporal registration, for the selected sequence of video frames.

The delay indicated at the final stage of the algorithm (Step 11 of § 6.4.1.4) may be different from the delay a viewer might choose, if aligning the scenes by eye. Viewers tend to focus on motion, aligning the high motion parts of the scene, where the frame-based algorithm chooses the most often observed delay over all of the frames that were examined. These overall delay histograms can be examined to determine the extent and statistics of any variable video delay present in the HRC.

6.4.2 Applying temporal registration correction

All of the quality features will require that the temporal delay calculated herein be removed. For positive delays, remove frames from the beginning of the processed file and the end of the original file. For negative delays, remove frames from the end of the processed file and the beginning of the original file. When reframing interlaced video sequences, the processed sequence is reframed. Thus one field should be removed from the beginning and end of the processed video sequence in addition to the above. Simultaneously, one frame must be removed from either the beginning of the original video file (i.e. –1 field delay overall) or the end of the original video file (i.e. +1 field delay overall).

Correcting for temporal registration will, in effect, shorten the length of available images in the video sequence. For simplicity, all further calculations will be based on the number of video frames available after all calibration corrections have been applied.

7 Quality features

7.1 Introduction

A quality feature is defined as a quantity of information associated with, or extracted from, a spatial-temporal sub-region of a video stream (either original or processed). The feature streams that are produced are a function of space and time. By comparing features extracted from the calibrated processed video with features extracted from the original video, a set of quality parameters (§ 8) can be computed that are indicative of perceptual changes in video quality. This section describes a set of quality features that characterize perceptual changes in the spatial, temporal, and chrominance properties of video streams. Normally, a perceptual filter is applied to the video stream to enhance some property of perceived video quality, such as edge information. After this perceptual filtering, features are extracted from spatial-temporal (S-T) sub-regions using a mathematical function (e.g. standard deviation). Finally, a perceptibility threshold is applied to the extracted features.

For the following discussion, an original feature stream will be denoted as $f_o(s, t)$ and the corresponding processed feature stream will be denoted as $f_p(s, t)$, where s and t are indices that denote the spatial and temporal positions, respectively, of the S-T region within the calibrated original and processed video streams. The features will be assigned lettered subscripts as they are discussed in the following sections, where the subscripted letters are chosen to be indicative of what the feature is measuring. All features operate on frames within a calibrated video sequence (see § 6); any interlace issues are addressed during calibration. All features operate independently of image size (i.e. S-T region size does not change when the image size changes)[8].

In summary, feature calculations perform the following steps. Some features may not require those steps marked [Optional].

Step 1: [Optional] Apply a perceptual filter.

Step 2: Divide the video stream into S-T regions.

Step 3: Extract features, or summary statistics, from each S-T region (e.g. mean, standard deviation).

Step 4: [Optional] Apply a perceptibility threshold.

Some features may utilize two or more different perceptual filters.

7.1.1 S-T regions

In general, features are extracted from localized S-T regions after the original and processed video streams have been perceptually filtered. The S-T regions are positioned to divide the video streams into abutting S-T regions. Since the processed video has been calibrated, for each processed video S-T region there exists an original S-T region spanning the identical spatial and temporal position within the video stream. Features are extracted from each S-T region by calculating summary statistics or some other mathematical function over the S-T region of interest.

Each S-T region describes a block of pixels. S-T region sizes are described by:

– the number of pixels horizontally,

– the number of frame lines vertically, and

---

[8] There is an implicit assumption that the viewing distance as a function of picture height remains fixed (e.g. closer viewing distances are used for smaller images). See Section 9 for further comments regarding the assumed viewing distance.

– the time duration of the region, given in units of equivalent video frames referenced to a 30 fps video system[9].

Figure 27 illustrates a S-T region of 8 horizontal pixels × 8 vertical lines × 6 NTSC video frames, for a total of 384 pixels. When applied to 25 fps video (PAL), this same S-T region spans 8 horizontal pixels × 8 vertical lines × 5 video frames, for a total of 320 pixels.

One fifth of a second is a desirable temporal extent, due to the ease of frame rate conversions (i.e. one fifth of a second results in an integer number of video frames for video systems operating at 10 fps, 15 fps, 25 fps and 30 fps). The general rule for frame rate conversion is to take the length of the S-T region in 30 fps video frames, divide by 30 and multiply by the frame rate of the video system under test. S-T regions that contain one video frame are presumed to always contain one video frame, independent of the frame rate.

The spatial region of interest (SROI, see § 3) encompassed by all S-T regions is identical for the original and calibrated processed video sequences. The SROI must lie entirely within the PVR, possibly with a buffer of pixels as required by any convolutional perceptual filter. The horizontal width of the SROI must be evenly divisible by the S-T region's horizontal extent. Likewise, the vertical height of the SROI must be evenly divisible by the S-T region's vertical extent. A user might further constrain the SROI to encompass a region of particular interest, such as the centre of the video frame.

Temporally, the original and calibrated processed video sequences are divided into an identical number of S-T regions, beginning with the first frame of temporally aligned video. If the number of valid frames available cannot be evenly divided by the S-T region's temporal extent, frames at the end of the clip are dropped from consideration.

For some features such as those presented in § 7.2, the 8 × 8_6F block achieves close to maximum correlation with subjective ratings. It should be noted, however, that the correlation decreases slowly as one moves away from the optimum S-T region size. Horizontal and vertical widths up to 32 or even larger and temporal widths up to 30 frames can be used with satisfactory results, giving the objective measurement system designer considerable flexibility in adapting the features to the available storage or transmission bandwidth [Wolf and Pinson, 2001].

After the video stream has been divided into S-T regions, the temporal axis of the feature (t) no longer corresponds to individual frames. Rather, the temporal axis contains a number of samples equal to the number of valid frames in the calibrated video sequence divided by the temporal extent of the S-T region.

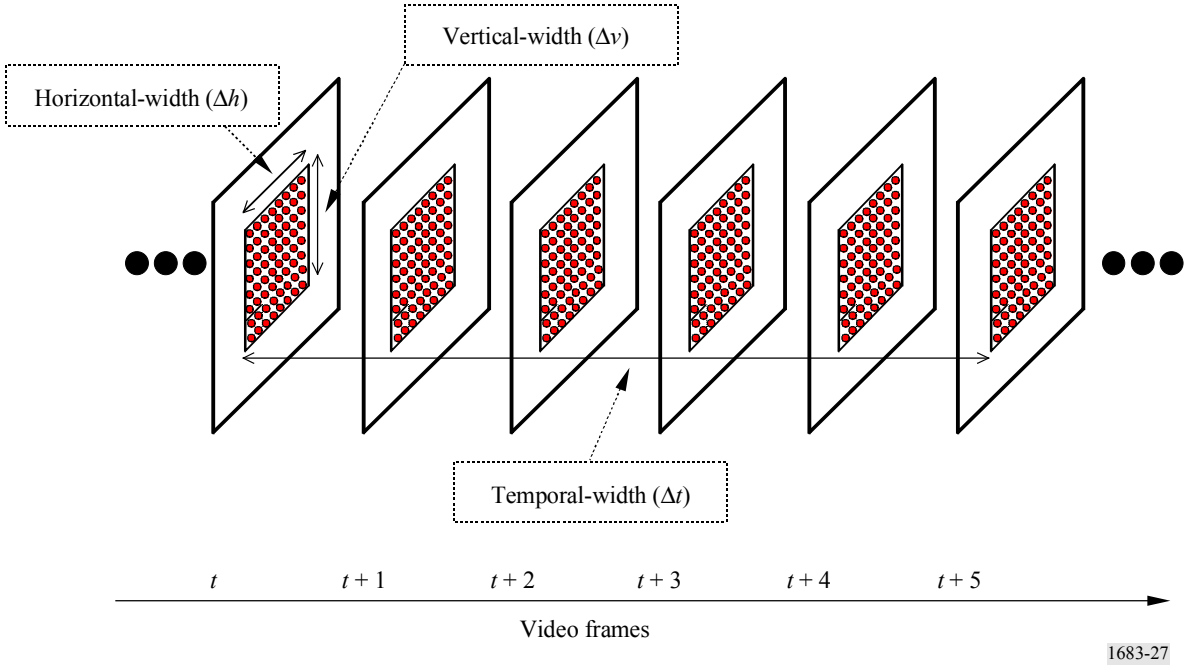
When computing two or more features simultaneously, further considerations become important. Ideally, all features should be calculated for the same SROI.

---

[9] All time durations in this Annex will be referenced to the equivalent number of video frames from a 30 fps video system. Thus, time durations of 6 frames (F) is used to represent both 6 frames from an NTSC system (6/30) and 5 frames from a PAL system (5/25). In addition, 30 fps and 29.97 fps are used interchangeably in this Annex, as this slight difference in frame rate is inconsequential for computation of VQM.

FIGURE 27

Example S-T region size for extracting features

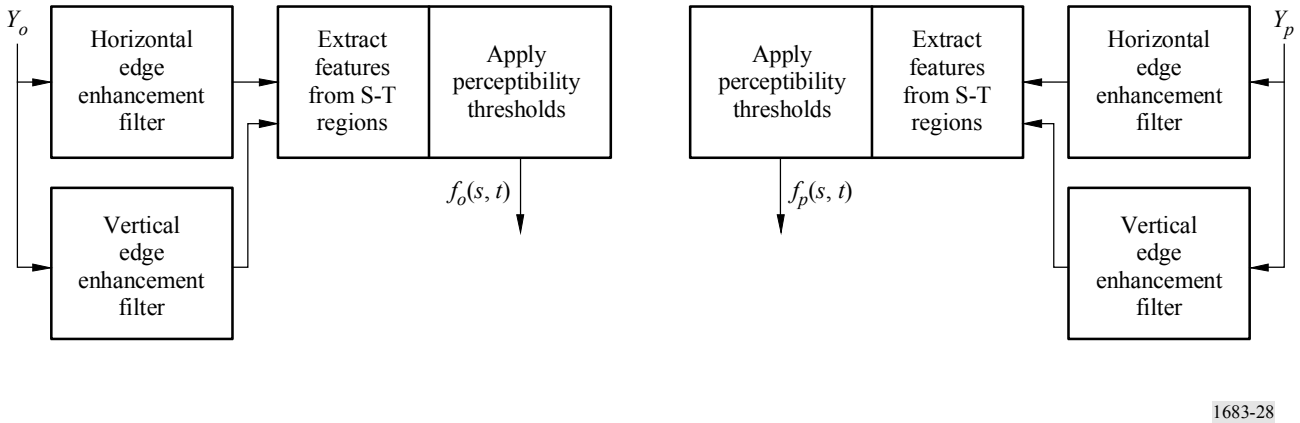


7.2 Features based on spatial gradients

Features derived from spatial gradients can be used to characterize perceptual distortions of edges. For example, a general loss of edge information results from blurring while an excess of horizontal and vertical edge information can result from block distortion or tiling. The *Y* components of the original and processed video streams are filtered using horizontal and vertical edge enhancement filters. Next, these filtered video streams are divided into S-T regions from which features, or summary statistics, are extracted that quantify the spatial activity as a function of angular orientation. Then, these features are clipped at the lower end to emulate perceptibility thresholds. The edge enhancement filters, the S-T region size, and the perceptibility thresholds were selected based on Recommendation ITU-R BT.601 video that has been subjectively evaluated at a viewing distance of six picture heights. Figure 28 presents an overview of the algorithm used to extract features based on spatial gradients.

FIGURE 28

Overview of algorithm used to extract spatial gradient features

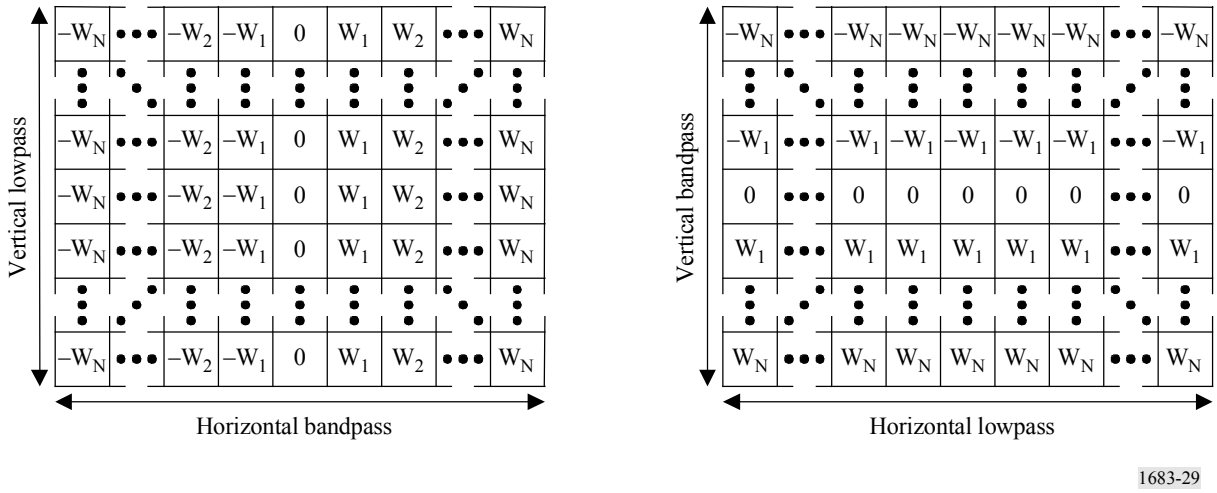


7.2.1 Edge enhancement filters

The original and processed *Y* (luminance) video frames are first processed with horizontal and vertical edge enhancement filters that enhance edges while reducing noise. The two filters shown in Fig. 29 are applied separately, one to enhance horizontal pixel differences while smoothing vertically (left filter), and the other to enhance vertical pixel differences while smoothing horizontally (right filter).

FIGURE 29

Edge enhancement filters



The two filters are transposes of each other, have size 13 × 13, and have filter weights given by

$$w_x = k \cdot \left(\frac{x}{c}\right) \cdot \exp\left\{-\left(\frac{1}{2}\right)\left(\frac{x}{c}\right)^2\right\}$$

where:

x: pixel displacement from the centre of the filter (0, 1, 2, …, N)

c: constant that sets the width of the bandpass filter

k: normalization constant selected such that each filter would produce the same gain as a true Sobel filter [Jain, 1989].

The optimal amount of horizontal bandpass filtering for a viewing distance of six times picture height was found to be given by the c = 2 filter, which has a peak response at about 4.5 cycles/degree. The bandpass filter weights that were used are given by:

[–0.0052625, –0.0173446, –0.0427401, –0.0768961, –0.0957739, –0.0696751, 0, 0.0696751, 0.0957739, 0.0768961, 0.0427401, 0.0173446, 0.0052625]

Note that the filters in Fig. 29 have a flat low-pass response. A flat low-pass response produced the best quality estimate and has the added advantage of being computationally efficient (e.g. for the left filter in Fig. 29, one merely has to sum the pixels in a column and multiply once by the weight).

7.2.2 Description of features f_{SI13} and f_{HV13}

This section describes the extraction of two spatial activity features from S-T regions of the edge-enhanced original and processed video streams from § 7.2.1. These features will be used to detect spatial impairments such as blurring and blocking. The filter shown in Fig. 29 (left) enhances

spatial gradients in the horizontal, H, direction while the transpose of this filter (right) enhances spatial gradients in the vertical, V, direction. The response at each pixel from the *H* and *V* filters can be plotted on a two dimensional diagram such as the one shown in Fig. 30 with the *H* filter response forming the abscissa value and the *V* filter response forming the ordinate value. For a given image pixel located at row *i*, column *j*, and time *t*, the *H* and *V* filter responses will be denoted as $H(i, j, t)$ and $V(i, j, t)$, respectively. These responses can be converted into polar coordinates (R, θ) using the relationships:

$$R(i, j, t) = \sqrt{H(i, j, t)^2 + V(i, j, t)^2}$$

and (102)

$$\theta(i, j, t) = \tan^{-1}\left[\frac{V(i, j, t)}{H(i, j, t)}\right]$$

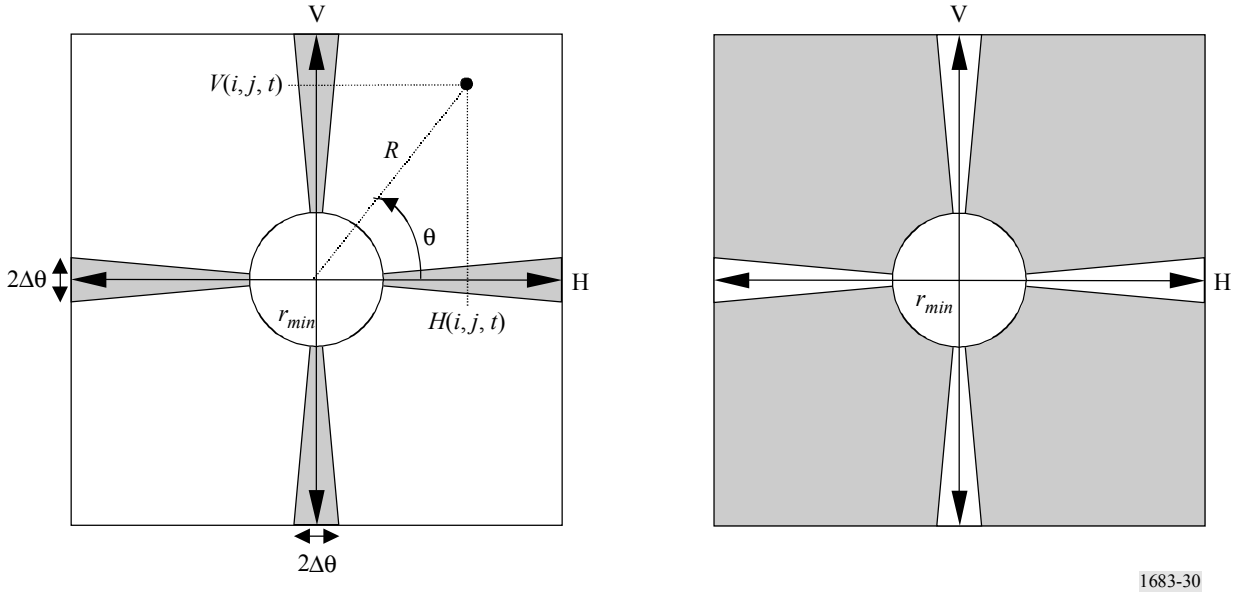
The first feature is a measure of overall SI and hence is denoted as f_{SI13} since images were pre-processed using the 13 × 13 filter masks shown in Fig. 29. This feature is computed simply as the standard deviation (std) over the S-T region of the $R(i, j, t)$ samples, and then clipped at the perceptibility threshold of *P* (i.e. if the result of the std calculation falls below *P*, f_{SI13} is set equal to *P*), namely

$$f_{SI13} = \left\{\text{std}\left[R(i, j, t)\right]\right\}\Big|_P \;/\; i, j, t \in \{\text{S-T region}\} \tag{103}$$

This feature is sensitive to changes in the overall amount of spatial activity within a given S-T region. For instance, localized blurring produces a reduction in the amount of spatial activity, whereas noise produces an increase. The recommended threshold *P* for this feature is 12.

FIGURE 30

Division of horizontal (H) and vertical (V) spatial activity into *HV* (left) and $\overline{HV}$ (right) distributions



The second feature, f_{HV13}, is sensitive to changes in the angular distribution, or orientation, of spatial activity. Complementary images are computed with the shaded spatial gradient distributions shown in Fig. 30. The image with horizontal and vertical gradients, denoted as *HV*, contains the $R(i, j, t)$ pixels that are horizontal or vertical edges (pixels that are diagonal edges are zeroed). The image with the diagonal gradients, denoted as $\overline{HV}$, contains the $R(i, j, t)$ pixels that are diagonal edges

(pixels that are horizontal or vertical edges are zeroed). Gradient magnitudes $R(i, j, t)$ less than r_{min} are zeroed in both images to assure accurate θ computations. Pixels in HV and $\overline{HV}$ can be represented mathematically as:

$$HV(i,j,t)=\begin{cases} R(i,j,t) & \text{if } R(i,j,t)\geq r_{min} \text{ and } m\frac{\pi}{2}-\Delta\theta<\theta(i,j,t)<m\frac{\pi}{2}+\Delta\theta \quad (m=0,1,2,3) \\ 0 & \text{otherwise} \end{cases} \tag{104}$$

and

$$\overline{HV}(i,j,t)=\begin{cases} R(i,j,t) & \text{if } R(i,j,t)\geq r_{min} \text{ and } m\frac{\pi}{2}+\Delta\theta\leq\theta(i,j,t)\leq(m+1)\frac{\pi}{2}-\Delta\theta \quad (m=0,1,2,3) \\ 0 & \text{otherwise} \end{cases} \tag{105}$$

where:

$$i, j, t \in \{\text{S-T region}\}$$

For the computation of HV and $\overline{HV}$ above, the recommended value for r_{min} is 20 and the recommended value for Δθ is 0.225 radians. Feature f_{HV13} for one S-T region is then given by the ratio of the mean of HV to the mean of $\overline{HV}$, where these resultant means are clipped at their perceptibility thresholds P, namely:

$$f_{HV13}=\frac{\{\text{mean}[HV(i,j,t)]\}|_P}{\{\text{mean}[\overline{HV}(i,j,t)]\}|_P} \tag{106}$$

The recommended perceptibility threshold P for the mean of HV and $\overline{HV}$ is 3. The f_{HV13} feature is sensitive to changes in the angular distribution of spatial activity within a given S-T region. For example, if horizontal and vertical edges suffer more blurring than diagonal edges, f_{HV13} of the processed video will be less than f_{HV13} of the original video. On the other hand, if erroneous horizontal or vertical edges are introduced, say in the form of blocking or tiling distortions, then f_{HV13} of the processed video will be greater than f_{HV13} of the original video. The f_{HV13} feature thus provides a simple means to include variations in the sensitivity of the human visual system with respect to angular orientation[10].

7.3 Features based on chrominance information

This section presents a single feature that can be used to measure distortions in the chrominance signals (C_B, C_R). For a given image pixel located at row i, column j, and time t, let C_B (i, j, t) and C_R (i, j, t) represent the Recommendation ITU-R BT.601 C_B and C_R values[11]. The components of a

---

[10] This discussion of f_{HV13}, though true in general, is somewhat simplified. For instance, when encountering some shapes the f_{HV13} filter behaves in a manner that may be counter-intuitive (e.g. a corner formed by the joining of a vertical and horizontal line will result in diagonal energy).

[11] Gain and offset corrections are not applied to the C_B and C_R image planes. See § 6.3.3.

two-dimensional chrominance feature vector, f_{COHER_COLOR}, are computed as the mean, mean over the S-T region of the C_B (i, j, t) and C_R (i, j, t) samples, respectively, giving more perceptual weight to the C_R component:

$$f_{COHER\_COLOR} = \left(\text{mean}\left[C_B(i,j,t)\right], W_R * \text{mean}\left[C_R(i,j,t)\right]\right) / i,j,t \in \{\text{ST region}\}, \text{and} \; W_R = 1.5 \quad (107)$$

Equation (107) performs coherent integration (hence the name f_{COHER_COLOR}) since the phase relationship between C_B and C_R is preserved. If one is familiar with a vectorscope, the value of the chrominance feature when examining colour bar signals is readily apparent. For general-purpose scenes, one can visualize the chrominance feature vector's usefulness for measuring distortions in chrominance for blocks of video that span a range of spatial and temporal extent. However, if S-T region size is too large, then many colours could be included in the calculation, and the usefulness of f_{COHER_COLOR} is reduced. An S-T region size of 8 horizontal pixels × 8 vertical lines × (1 to 3) video frames produces a robust chrominance feature vector (actually 4 horizontal C_B and C_R pixels, since these signals are sub-sampled by two in the horizontal direction for Recommendation ITU-R BT.601 sampling).

7.4 Features based on contrast information

Features that measure localized contrast information are sensitive to quality degradations such as blurring (e.g. contrast loss) and added noise (e.g. contrast gain). One localized contrast feature, f_{CONT}, is easily computed for each S-T region from the Y luminance image as:

$$f_{CONT} = \left\{\text{std}\left[Y(i,j,t)\right]\right\}\Big|_P / i,j,t \in \{\text{ST region}\} \quad (108)$$

The recommended perceptibility threshold P for the f_{CONT} feature is between four and six.

7.5 Features based on ATI

Features that measure distortions in the flow of motion are sensitive to quality degradations such as dropped or repeated frames (motion loss) and added noise (motion gain). An ATI feature, f_{ATI}, is computed for each S-T region by first generating a motion video stream that is the absolute value of the difference between consecutive video frames at time t and $t-1$, and then computing the standard deviation over the S-T region. Mathematically, this process will be represented as:

$$f_{ATI} = \left\{std\left|Y(i,j,t) - Y(i,j,t-1)\right|\right\}\Big|_P / i,j,t \in \{\text{S-T region}\} \quad (109)$$

The recommended perceptibility threshold P for the f_{ATI} feature is between one and three.

The use of a previous frame introduces considerations beyond those required by the other features. When calculating f_{ATI} jointly with another feature (e.g. $f_{CONTRAST\_ATI}$ from § 7.6) or for use in a model (see § 9), the requirement of an extra frame complicates the task of placement of S-T regions (see § 7.1.1).

7.6 Features based on the cross product of contrast and ATI

The perceptibility of spatial impairments can be influenced by the amount of motion that is present. Likewise, the perceptibility of temporal impairments can be influenced by the amount of spatial detail that is present. A feature derived from the cross product of contrast information and absolute

temporal information can be used to partially account for these interactions. This feature, denoted as $f_{CONTRAST\_ATI}$, is computed as the product of the features in § 7.4 and 7.5 [12]. The recommended perceptibility threshold $P = 3$ is applied separately to each feature (f_{CONT} and f_{ATI}) before computing their cross product. Impairments will be more visible in S-T regions that have a low cross product than in S-T regions that have a high cross product. This is particularly true of impairments like noise and error blocks.

The requirement of an extra frame for f_{ATI} complicates $f_{CONTRAST\_ATI}$ slightly, since the S-T regions used by both f_{CONT} and f_{ATI} must be placed identically. Either one frame at the beginning of the video sequence must be left unused for f_{ATI}, or the S-T regions located at the beginning of the video sequence must contain one fewer frame (e.g. given a temporal extent of $6F$, the first f_{ATI} S-T region would use $5F$ instead of $6F$). The parameters and models specified herein presume the second solution will be used.

8 Quality parameters

8.1 Introduction

Quality parameters that measure distortions in video quality due to gains and losses in the feature values are first calculated for each S-T region by comparing the original feature values, $f_o(s, t)$, with the corresponding processed feature values, $f_p(s, t)$ (§ 8.2). Several functional relationships are used to emulate the visual masking of impairments for each S-T region. Next, error-pooling functions across space and time emulate how humans deduce subjective quality ratings. Error pooling across space will be referred to as spatial collapsing (§ 8.3), and error pooling across time will be referred to as temporal collapsing (§ 8.4). Sequential application of the spatial and temporal collapsing functions to the stream of S-T quality parameters produces quality parameters for the entire video clip, which is nominally 5 to 10 s in duration. The final time-collapsed parameter values may be scaled and clipped (§ 8.5) to account for non-linear relationships between the parameter value and perceived quality and to further reduce the parameter's sensitivity.

In summary, parameter calculations perform the following steps. Some features may not require the [Optional] step.

Step 1: Compare original feature values with processed feature values.

Step 2: Perform spatial collapsing.

Step 3: Perform temporal collapsing.

Step 4: [Optional] Perform non-linear scaling and/or clipping.

All parameters are designed to be either all positive or all negative. A parameter value of zero indicates no impairment.

8.2 Comparison functions

The perceptual impairment at each S-T region is calculated using functions that model visual masking of the spatial and temporal impairments. This section presents the masking functions that are used by the various parameters to produce quality parameters as a function of space and time.

---

[12] A standard cross product of the f_{CONT} and f_{ATI} features (i.e. $f_{CONT} * f_{ATI}$) is used for the processed $f_p(s, t)$ and original $f_o(s, t)$ features in the ratio_loss and ratio_gain comparison functions described in § 8.2.1. However, for the *log_loss* and *log_gain* comparison functions, the processed and original features are computed as $\log_{10}[f_{CONT}] * \log_{10}[f_{ATI}]$, and the comparison functions use subtraction (i.e. $f_p(s, t) - f_o(s, t)$ rather than $\log_{10}[f_p(s, t) / f_o(s, t)]$).

8.2.1 Error ratio and logarithmic ratio

Loss and gain are normally examined separately, since they produce fundamentally different effects on quality perception (e.g. loss of spatial activity due to blurring and gain of spatial activity due to noise or blocking). Of the many comparison functions that have been evaluated, two forms have consistently produced the best correlation to subjective ratings. Each of these forms can be used with either gain or loss calculations for a total of four basic S-T comparison functions. The four primary forms are:

$$\text{ratio\_loss}(s,t) = np\left\{\frac{f_p(s,t) - f_o(s,t)}{f_o(s,t)}\right\}$$

$$\text{ratio\_gain}(s,t) = pp\left\{\frac{f_p(s,t) - f_o(s,t)}{f_o(s,t)}\right\}$$

$$\text{log\_loss}(s,t) = np\left\{\log_{10}\left[\frac{f_p(s,t)}{f_o(s,t)}\right]\right\}$$

$$\text{log\_gain}(s,t) = pp\left\{\log_{10}\left[\frac{f_p(s,t)}{f_o(s,t)}\right]\right\}$$

where:

pp: positive part operator (i.e. negative values are replaced with zero)

np: negative part operator (i.e. positive values are replaced with zero).

These visual masking functions imply that impairment perception is inversely proportional to the amount of localized spatial or temporal activity that is present. In other words, spatial impairments become less visible as the spatial activity increases (i.e. spatial masking), and temporal impairments become less visible as the temporal activity increases (i.e. temporal masking). While the logarithmic and ratio comparison functions behave very similarly, the logarithmic function tends to be slightly more advantageous for gains while the ratio function tends to be slightly more advantageous for losses. The logarithm function has a larger dynamic range, and this is useful when the processed feature values greatly exceed the original feature values.

8.2.2 Euclidean distance

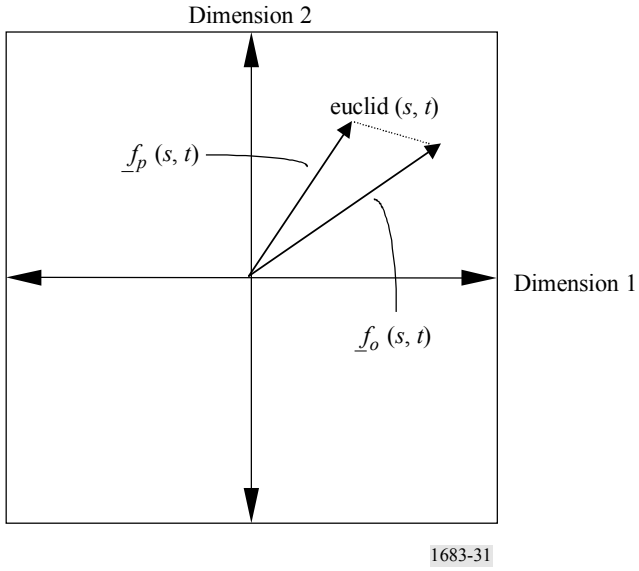
Another useful S-T comparison function is simple Euclidean distance, represented by the length of the difference vector between the original feature vector $f_o(s, t)$ and the corresponding processed feature vector, $f_p(s, t)$:

$$\text{euclid}(s,t) = \left\| \underline{f}_p(s,t) - \underline{f}_o(s,t) \right\| \tag{110}$$

Figure 31 gives an illustration of Euclidean distance for a two-dimensional feature vector extracted from an S-T region (e.g. the f_{COHER_COLOR} feature vector of § 7.3), where s and t are indices that denote the spatial and temporal positions, respectively, of the S-T region within the calibrated original and processed video streams. The dashed line in Fig. 31 shows the Euclidean distance. The Euclidean distance measure can be generalized for feature vectors that have an arbitrary number of dimensions.

FIGURE 31

Illustration of the Euclidean distance euclid (*s*, *t*) for a two-dimensional feature vector



8.3 Spatial collapsing functions

The parameters from the S-T regions (from § 8.2) form three-dimensional matrices spanning one temporal axis and two spatial dimensions (i.e. horizontal and vertical placement of the S-T region). Next, impairments from the S-T regions with the same time index *t* are pooled using a spatial collapsing function. Spatial collapsing yields a time history of parameter values. This time history of parameter values, denoted generically as *p*(*t*), must then be temporally collapsed using a temporal collapsing function given in § 8.4. Table 12 presents a summary of the most commonly used spatial collapsing functions.

Extensive investigation has revealed that the optimal spatial collapsing functions normally involve some form of worst case processing, like the average of the worst 5% of the distortions observed over the spatial index *s* [Wolf and Pinson, 1998, 1999, 2001 and June 2002]. This is because localized impairments tend to draw the focus of the viewer, making the worst part of the picture the predominant factor in the subjective quality decision. For example, the spatial collapsing function above95% is computed at each temporal index *t* for the log_gain(*s*, *t*) function in § 8.2.1 as the average of the most positive 5% of the values over the spatial index *s* [13]. This amounts to sorting the gain distortions from low to high at each temporal index *t* and averaging those distortions that are above the 95% threshold (since more positive values imply greater distortion). Similarly, loss distortions such as those produced by the ratio_loss(*s*, *t*) function in § 8.2.1 would be sorted at each temporal index *t*, but the average of those distortions that are below5% is used (since losses are negative).

8.4 Temporal collapsing functions

The parameter time history results *p*(*t*) output from the spatial collapsing function (from § 8.3) are next pooled using a temporal collapsing function to produce an objective parameter *p* for the video clip, which is nominally 4 to 10 s in length. Viewers seem to use several temporal collapsing

[13] Notice that the time index, *t*, does not indicate individual frames (see § 7.1.1) here. Instead, each value of *t* corresponds to those S-T regions having the same time extent.

functions when subjectively rating video clips that are approximately 10 s in length. The mean over time is indicative of the average quality that is observed during the time period. The 90% and 10% levels over time are indicative of the worst transient quality that is observed for gains and losses, respectively (e.g. digital transmission errors may cause a 1 to 2 s disturbance in the processed video). After temporal collapsing, a given parameter *p* is either all negative or all positive, but not both. Table 13 presents a summary of the most commonly used temporal collapsing functions.

TABLE 12

Spatial collapsing functions and their definitions

Spatial collapsing function	Definition
below5%	For each temporal index *t*, sort the parameter values from low to high. Compute the average of all the parameter values that are less than or equal to the 5% threshold level. For loss parameters, this spatial collapsing function produces a parameter that is indicative of the worst quality over space
above95%	For each temporal index *t*, sort the parameter values from low to high. Compute the average of all the parameter values that are greater than or equal to the 95% threshold level. For gain parameters, this spatial collapsing function produces a parameter that is indicative of the worst quality over space
mean	For each temporal index *t*, compute the average of all the parameter values. This spatial collapsing function produces a parameter that is indicative of the average quality over space
std	For each temporal index *t*, compute the standard deviation of all the parameter values. This spatial collapsing function produces a parameter that is indicative of the quality variations over space
below5%tail	For each temporal index *t*, sort the parameter values from low to high. Compute the average of all the parameter values that are less than or equal to the 5% threshold level, and then subtract the 5% level from this average. For loss parameters, this spatial collapsing function allows one to measure the spread of the worst quality levels over space. It is useful for measuring the perceptual quality effects of spatially localized distortions
above99%tail	For each temporal index *t*, sort the parameter values from low to high. Compute the average of all the parameter values that are greater than or equal to the 99% threshold level, and then subtract the 99% level from this average. For gain parameters, this spatial collapsing function allows one to measure the spread of the worst quality levels over space. It is useful for measuring the perceptual quality effects of spatially localized distortions

TABLE 13

Temporal collapsing functions and their definitions

Temporal collapsing function	Definition
10%	Sort the time history of the parameter values from low to high and select the 10% threshold level. For loss parameters, this temporal collapsing function produces a parameter that is indicative of the worst quality over time. For gain parameters, it produces a parameter that is indicative of the best quality over time
25%	Sort the time history of the parameter values from low to high and select the 25% threshold level
50%	Sort the time history of the parameter values from low to high and select the 50% threshold level
90%	Sort the time history of the parameter values from low to high and select the 90% threshold level. For loss parameters, this temporal collapsing function produces a parameter that is indicative of the best quality over time. For gain parameters, it produces a parameter that is indicative of the worst quality over time
mean	Compute the mean of the time history of the parameter values. This produces a parameter that is indicative of the average quality over time
std	Compute the standard deviation of the time history of the parameter values. This temporal collapsing function produces a parameter that is indicative of the quality variations over time
above90%tail	Sort the time history of the parameter values from low to high and compute the average of all the parameter values that are greater than or equal to the 90% threshold level, and then subtract the 90% level from this average. For gain parameters, this temporal collapsing function allows one to measure the spread of the worst quality levels over time. It is useful for measuring the perceptual quality effects of temporally localized distortions

8.5 Non-linear scaling and clipping

The all-positive or all-negative temporally collapsed parameter p from § 8.4 may be scaled to account for non-linear relationships between the parameter value and perceived quality. It is preferable to remove any non-linear relationships before building the video quality models (§ 9), since a linear least-squares algorithm will be used to determine the optimal parameter weights. The two non-linear scaling functions that might be applied are the square root function, denoted by sqrt, and the square function, denoted by square. If the sqrt function is applied to an all-negative parameter, the parameter is first made all positive (i.e. absolute value taken).

Finally, a clipping function denoted as clip_T, where T is the clipping threshold, might be applied to reduce the sensitivity of the parameter to small impairments. The clipping function replaces any

parameter value between the clipping level and zero with the clipping level, and then the clipping level is subtracted from all resulting parameter values. This is represented mathematically as:

$$\text{clip\_T}(p) = \begin{cases} \max(p, T) - T & \text{if } p \text{ is all positive} \\ \min(p, T) - T & \text{if } p \text{ is all negative} \end{cases}$$

8.6 Parameter naming convention

This section summarizes the technical naming convention used for video quality parameters. This convention assigns to each parameter a lengthy name consisting of identifying words (sub-names) separated by underscores. The technical parameter name summarizes the exact process used to calculate the parameter. Each sub-name identifies one function or step in the process of calculating the parameter. Sub-names are listed in the order in which they occur, from left to right. Table 14 summarizes the sub-names used to create the technical parameter name, listed in the order that they occur. Paragraph 8.6.1 provides examples of technical parameter names and their associated sub-names from Table 14.

TABLE 14

Technical naming convention used for video quality parameters

Sub-name	Definition	Examples
Colour	The colour space image planes used by the parameter	*Y* for luminance image plane. *color* for (C_B, C_R) image planes
Feature Specific	The "Feature Specific" sub-name describes the calculations that make this parameter unique. All other sub-names that follow are generic processes that can be used by many different types of parameters. The "Feature Specific" sub-name is usually the name of the feature that is extracted from the "Colour" plane at this point in the flow, hence the location of this sub-name. However, information not otherwise covered by the naming convention can also be included here. For example, the HV parameter applies the "Block Statistic" sub-name separately to the *HV* and $\overline{HV}$ image planes. The subsequent ratio of *HV* to $\overline{HV}$ is specified by the "Feature Specific" sub-name (i.e. rather than occupying a separate sub-name after the "Block Statistic")	*si13* for the f_{SI13} feature in § 7.2.2. *hv13_angleX.XXX_rmin YY* for the f_{HV13} feature in § 7.2.2, where *X.XXX* is $\Delta\theta$ and *YY* is the r_{min}. *coher_color* for the f_{COHER_COLOR} feature in § 7.3. *cont*: for the f_{CONT} feature in § 7.4. *ati*: for the f_{ATI} feature in § 7.5. *contrast_ati*: for the $f_{CONTRAST\_ATI}$ feature in § 7.6
Block Shift	Present when S-T blocks slide (e.g. overlap in time). When absent, blocks are assumed to abut in time	*sliding*
Full Image	Present when the S-T block size contains the entire valid region of the image. When absent, the "Block Size" sub-name must be present	*image*

TABLE 14

Technical naming convention used for video quality parameters

Sub-name	Definition	Examples
Block Size	Present when the image is divided into S-T blocks (see § 7.1.1). For consistency, block size is always indicated relative to the luminance (*Y*) plane's frame lines and frame pixels. Thus, for 4:2:2 sampled video, colour blocks will actually contain half the specified number of pixels horizontally. When absent, the "Full Image" sub-name must be present	*8 × 8*: for blocks that include 8 frame lines vertically by 8 frame pixels horizontally. *128 × 128*: for blocks that include 128 frame lines vertically by 128 frame pixels horizontally
Block Frames	Indicates the temporal extent of the S-T blocks (see § 7.1.1), referenced to 30 fps video. For example, 6*F* is used to represent one fifth of a second, regardless of the frame rate of the video being measured (e.g. 5 frames from a 25 fps system, 3 frames from a 15 fps system, 2 frames from a 10 fps system)	*1F:* for a temporal extent of one frame. *6F:* for a temporal extent of one fifth of a second
Block Statistic	The statistical function used to extract the feature from each S-T region, producing one number for each S-T block of pixels. Present unless "Block Size" = *1 × 1* (i.e. 1 pixel). Before the Block Statistic has been applied, intermediate results contain time histories of images with one number per pixel (i.e. filtered images); afterward, intermediate results contain one number per each S-T region (i.e. feature images). Parameters that have two image planes (e.g. hv13 and coher_color) will apply the Block Statistic separately to both image planes, producing two feature images	mean is the average of the pixel values. std is the standard deviation of the pixel values. rms is the root mean square of the pixel values
Perceptibility Threshold	The values produced by the "Block Statistic" may be clipped at a perceptibility threshold *P*. Values between zero and this threshold are replaced with the threshold	3 for a minimum feature value of 3.0. 12 for a minimum feature value of 12.0
Comparison Function	The function used to compare features extracted from the original and processed feature streams (see § 8.2). Before the "Comparison Function", the intermediate results contain time histories of original and processed feature images; afterward the intermediate results contain a time history of parameter images	log_gain (see § 8.2.1). ratio_loss (see § 8.2.1). euclid (see § 8.2.2)
Spatial Collapsing Function	See § 8.3. The function is applied to each parameter image (e.g. all S-T regions having the same temporal index) and produces a time history of parameter values. Before spatial collapsing, intermediate results consist of parameter images containing one value for each S-T block; afterward, intermediate results are a time history of numbers (i.e. parameter time history). Must be present for all parameters except "Full Image" parameters	See Table 12

TABLE 14

Technical naming convention used for video quality parameters

Sub-name	Definition	Examples
Temporal Collapsing Function	See § 8.4. The function is applied to the parameter time history and produces one parameter value for the entire video sequence. After temporal collapsing, the parameter contains either all negative values or all positive values, but not both. Zero is associated with no impairment, and parameter values further from zero have higher impairments. Must be present for all parameters	See Table 13
Non-linear Function	See § 8.5. Examination of the parameter's values may indicate that the parameter should be scaled in a non-linear fashion to linearly track the subjective data. The Non-linear Function performs this final scaling. If the sqrt function is applied to an all-negative parameter, the parameter is first made all positive (i.e. absolute value taken)	sqrt for the square root of the temporally collapsed parameter value. square for the square of the temporally collapsed parameter value
Clipping Function	See § 8.5. Final examination of the parameter values may indicate a need to further reduce the sensitivity of the parameter to small impairments (e.g. parameter values near zero). Replace any value between the clipping level *T* and zero with the clipping level, and then subtract the clipping level from all resulting parameter values	clip_0.45 If parameter values are positive, replace all values less than 0.45 with 0.45 and then subtract 0.45 from all the parameter values. If parameter values are negative, replace all values greater than –0.45 with –0.45 and then add 0.45 to all the parameter values

8.6.1 Example parameter names

This section includes five example technical names, and a step-by-step description of the sub-naming procedure given in Table 14.

*Y*_si13_8×8_6F_std_6_ratio_loss_below5%_mean

Y means that the luminance image plane is used. si13 represents filtering of those images with the 13×13 spatial masks in § 7.2.1 in preparation for extraction of the f_{SI13} feature in § 7.2.2. 8×8_6F represents dividing the video stream into S-T regions containing eight frame lines vertically by eight pixels horizontally by one fifth of a second temporally (i.e. 6 NTSC frames, 5 PAL frames). std represents taking the standard deviation of each block. *6* represents application of a perceptibility threshold, replacing all standard deviation values below 6.0 with a value of 6.0. ratio_loss represents comparing the original and processed features from each block using the ratio_loss function. below5% represents spatially collapsing the parameter values at each time index using the below5% function. mean represents temporally collapsing the parameter time history using the mean function.

color_coher_color_8×8_1F_mean_euclid_std_10%_clip_0.8

color represents using the C_B and C_R image planes. coher_color represents preservation of the phase relationship between the C_B and C_R images (by treating them separately) in preparation for extraction of the f_{COHER_COLOR} feature in § 7.3. 8 × 8_1F represents dividing each frame into blocks that are 8 frame lines high by 4 C_B and C_R pixels wide (due to 4:2:2 sub-sampling of the C_B and C_R image planes) by 1 frame in time. mean represents taking the mean value of each block. euclid represents computing the Euclidean distance between original vectors (C_B, C_R) and processed vectors (C_B, C_R) for each S-T block. std represents the std spatial collapsing function. 10% represents the 10% temporal collapsing function. clip_0.8 represents clipping the final parameter value at a minimum of 0.8 (i.e. replacing all values below 0.8 with 0.8, and then subtracting 0.8).

*Y*_hv13_angle0.225_rmin20_8×8_6F_mean_3_ratio_loss_below5%_mean_square_clip_0.05

Y means that the luminance image plane is used. hv*13* represents filtering of the *Y* images with the 13 × 13 spatial masks in § 7.2.1 in preparation for extraction of the f_{HV13} feature in § 7.2.2 (i.e. the *HV* and $\overline{HV}$ images are created and treated separately until after the Perceptibility Threshold). angle0.225 and rmin20 represent a $\Delta\theta$ of 0.225 radians and an r_{min} of 20 for calculation of the f_{HV13} feature. 8 × 8_6F represents dividing the video stream into S-T regions containing eight frame lines vertically by eight pixels horizontally by one-fifth of a second temporally (i.e. 6 NTSC frames, 5 PAL frames). mean represents taking the mean value of each S-T block for *HV* and $\overline{HV}$. 3 represents the application of a perceptibility threshold to these means, replacing all values less than 3 with 3.0. Next, the f_{HV13} feature in § 7.2.2 is calculated as the ratio of clipped means of *HV* to the clipped means of $\overline{HV}$, as specified in hv13_angle0.225_rmin20, the Feature Specific sub-name. ratio_loss represents using the ratio_loss comparison function for each original and corresponding processed f_{HV13} feature extracted from an S-T block. below5% specifies the spatial collapsing function. mean specifies the temporal collapsing function. square specifies the non-linear function for each time-collapsed parameter value. clip_0.05 represents the clipping function, where all values below 0.05 are replaced with 0.05, and then 0.05 is subtracted from the result (recall that the all-negative parameter will become an all-positive parameter due to the non-linear function, square).

*Y*_contrast_ati_4×4_6F_std_3_ratio_gain_mean_10%

Y means the luminance plane is used. contrast_ati represents computing two separate filtered versions of the image in preparation for extraction of the $f_{CONTRAST\_ATI}$ feature in § 7.6. The first filter, contrast, will consider the luminance planes directly (§ 7.4). The second filter, ati, will consider images generated by taking differences between successive luminance planes (§ 7.5). The contrast and ati images are treated separately until after the thresholding. 4 × 4_6F means that the two video streams are divided into S-T regions containing four frame lines vertically by four pixels horizontally by one-fifth of a second temporally (e.g. 6 NTSC frames, 5 PAL frames). The first S-T block of *ati* images will actually contain only 5 images rather than 6 since an ati image cannot be generated for the first frame in the sequence (i.e. there is no earlier image in time available). This exception is specified as part of the Feature Specific sub-name. *std* represents taking the standard deviation of each block. Then, as specified in the Feature Specific sub-name in § 7.6, apply a perceptibility threshold of 3 to both the contrast and ati features (replace all values less than 3 with 3.0). Next, multiply the contrast block-value with the *ati* block-value for each S-T block (see footnote in § 7.6 for special instructions on how to perform this multiplication) and continue calculations with this combined feature image. ratio_gain is the comparison function used to compare each original and processed feature from the S-T blocks. mean is the spatial collapsing function. 10% is the temporal collapsing function.

9 General model

This section provides a full description of the general model VQM (denoted as VQM_G). The general model is optimized to achieve maximum objective to subjective correlation using a wide range of video quality and bit rates. The general model has objective parameters for measuring the perceptual effects of a wide range of impairments such as blurring, block distortion, jerky/unnatural motion, noise (in both the luminance and chrominance channels), and error blocks (e.g. what might typically be seen when digital transmission errors are present). This model consists of a linear combination of video quality parameters whose naming conventions are described in § 8.6. The selection of video quality parameters was determined by the optimization criteria given above. The general model produces output values that range from zero (no perceived impairment) to approximately one (maximum perceived impairment). To place results on the double stimulus continuous quality scale (DSCQS), multiply VQM_G by 100.

The general model was designed based on Recommendation ITU-R BT.601 video that has been subjectively evaluated at a viewing distance of six picture heights. When analysing video sequences for different viewing distances, a scaling factor must be applied to the results. As viewing distance increases, impairments become less visible; as viewing distance decreases, impairments become more visible. Care should be taken when comparing results for video sequences that will be viewed at different viewing distances.

VQM_G consists of a linear combination of seven parameters. Four parameters are based on features extracted from spatial gradients of the Y luminance component (§ 7.2.2), two parameters are based on features extracted from the vector formed by the two chrominance components (C_B, C_R) (§ 7.3), and one parameter is based on contrast and absolute temporal information features, both extracted from the Y luminance component (§ 7.4 and 7.5, respectively). VQM_G is given by:

$$VQM_G = \{-0.2097 * Y\_si13\_8\times8\_6F\_std\_12\_ratio\_loss\_below5\%\_10\%$$

$$+0.5969 * Y\_hv13\_angle0.225\_rmin20\_8\times8\_6F\_mean\_3\_ratio\_loss\_below5\%\_mean\_square\_clip\_0.06$$

$$+0.2483 * Y\_hv13\_angle0.225\_rmin20\_8\times8\_6F\_mean\_3\_log\_gain\_above95\%\_mean$$

$$+0.0192 * color\_coher\_color\_8\times8\_1F\_mean\_euclid\_std\_10\%\_clip\_0.6$$

$$-2.3416 * [Y\_si13\_8\times8\_6F\_std\_8\_log\_gain\_mean\_mean\_clip\_0.004 \mid ^{0.14}]$$

$$+0.0431 * Y\_contrast\_ati\_4\times4\_6F\_std\_3\_ratio\_gain\_mean\_10\%$$

$$+0.0076 * color\_coher\_color\_8\times8\_1F\_mean\_euclid\_above99\%tail\_std\} \mid_{0.0}$$

Remember, that the above features for the general model with a "6F" time extent will actually contain five PAL (625-line) video frames.

The square on the hv_loss parameter is necessary to linearize the parameter response with respect to the subjective data. Note that since the hv_loss parameter becomes positive after the square, a positive multiplying weight is used. Also note that the hv_loss parameter is clipped at 0.06, the colour parameter is clipped at 0.6, and the si_gain parameter is clipped at 0.004. The si_gain parameter is the only quality improvement parameter in the model (since the si_gain parameter is positive, a negative weight results in negative contributions to VQM which produce quality improvements). The si_gain parameter measures improvements to quality that result from edge sharpening or enhancement. Clipping of the parameter at an upper threshold of 0.14 immediately before multiplying by the parameter weight prevents excessive improvements to VQM of more than about 1/3 of a quality unit, which is the maximum improvement observed in the general subjective data set (i.e. an HRC will only be rewarded for a little edge enhancement).

The total VQM (after the contributions of all the parameters are added up) is clipped at a lower threshold of 0.0 to prevent negative VQM numbers. Finally, a crushing function that allows a

maximum of 50% overshoot is applied to VQM values over 1.0 to limit VQM values for excessively distorted video that falls outside the range of the currently available subjective data.

If $VQM_G > 1.0$, then $VQM_G = (1 + c) * VQM_G / (c + VQM_G)$, where $c = 0.5$.

VQM_G computed in the above manner will have values greater than or equal to zero and a nominal maximum value of one. VQM_G may occasionally exceed one for video scenes that are extremely distorted.

10 References

JAIN, A. K. [1989] *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall Inc., 1989, p. 348-357.

PINSON, M. and WOLF, S. [February 2002] Video Quality Measurement User's Manual. NTIA Handbook 02-1. National Telecommunications and Information Administration.

SMPTE [1995a] SMPTE 125M. Television – Component Video Signal 4:2:2 – Bit-Parallel Digital Interface. Society of Motion Picture and Television Engineers, 595 West Hartsdale Avenue, White Plains, NY 10607.

SMPTE [1995b] SMPTE Recommended Practice 187. Center, Aspect Ratio, and Blanking of Video Images. Society of Motion Picture and Television Engineers, 595 West Hartsdale Avenue, White Plains, NY 10607.

SMPTE [1999] SMPTE 170M. SMPTE Standard for Television – Composite Analog Video Signal – NTSC for Studio Applications. Society of Motion Picture and Television Engineers, 595 West Hartsdale Avenue, White Plains, NY 10607.

WOLF, S. and PINSON, M. [12-13 November 1998] In-service performance metrics for MPEG-2 video systems. Proc. Made to Measure 98 – Measurement Techniques of the Digital Age Technical Seminar, technical conference jointly sponsored by the International Academy of Broadcasting (IAB), the ITU, and the Technical University of Braunschweig (TUB), Montreux, Switzerland.

WOLF, S. and PINSON, M. [September 1999] Spatial-temporal distortion metrics for in-service quality monitoring of any digital video system. Proc. SPIE International Symposium on Voice, Video, and Data Communications, Boston, MA.

WOLF, S. and PINSON, M. [July 2001] The relationship between performance and spatial-temporal region size for reduced-reference, in-service video quality monitoring systems. Proc. SCI/ISAS 2001 (Systematics, Cybernetics, and Informatics/Information Systems Analysis and Synthesis), p. 323-328. National Telecommunications and Information Administration.

WOLF, S. and PINSON, M. [June 2002] Video Quality Measurement Techniques. NTIA Report 02-392.

Annex 5a

NTIA VQM raw objective data

This Annex provides a full disclosure of the NTIA VQM raw objective data.

Raw data summary

This General Model developed by the NTIA was originally designed to output values on a nominal 0 to 1 scale, where 0 represents no perceived impairment and 1 represents maximum perceived impairment. However, the binary executable submitted to the VQEG Phase II FR-TV test

transformed the (0, 1) values of the General Model to (0, 100) to match the DSCQS. Since all model values should now be scaled to (0, 1), we have removed the 100 times multiplication factor (i.e. multiplication by 100) to restore the original (0, 1) scale of the General Model.

The General Model values calculated here used the centre 8 s of video in each clip, discarding the 10 extra frames of video at the beginning and end of each video file as described in the VQEG Phase II FR-TV Test Plan. For the calibration routines, an uncertainty of 30 frames and a frequency of 15 frames were used (see § 6 of Annex 5). In addition, the SROI used to calculate the VQM value for each clip was chosen as follows:

Step 1: For 525-line video systems, use a default SROI of 672 pixels × 448 lines centred in the video frame. For 625-line video systems, use a default SROI of 672 pixels × 544 lines centred in the video frame. These SROI defaults may be modified as given in Steps 2 and 3.

Step 2: The model requires 6 additional valid pixels/lines on all sides of the above SROI for the spatial filters to operate properly. If the PVR (calculated automatically as given in § 6.2 of Annex 5) is not large enough to encompass the default SROI plus 6 pixels/lines (Step 1), then the SROI is reduced by multiples of 8 pixels/lines only in the necessary direction (horizontal or vertical).

Step 3: The SROI is always centred horizontally such that the left hand sample starts at Recommendation UIT-R BT.601 luminance/chrominance co-located sampling point. The SROI is centred vertically such that when separated into two fields, the same number of lines is discarded from the top of each field. If the SROI has been reduced in size in Step 2, then perfect centring of the SROI within the video frame may not be possible.

Evaluation software that implements the General Model and its calibration routines may be downloaded from:

http://www.its.bldrdoc.gov/n3/video/vqmsoftware.htm

TABLE 15

525-line raw objective data

Source No.	HRC No.	NTIA: Proponent H
1	1	0.660 [1]
1	2	0.347
1	3	0.286
1	4	0.178
2	1	0.449
2	2	0.246
2	3	0.119
2	4	0.061
3	1	0.321
3	2	0.167
3	3	0.076
3	4	0.049
4	5	0.396
4	6	0.280
4	7	0.222
4	8	0.183
5	5	0.329
5	6	0.217
5	7	0.159
5	8	0.115
6	5	0.542
6	6	0.266
6	7	0.189
6	8	0.139
7	5	0.258
7	6	0.161
7	7	0.108
7	8	0.076
8	9	0.911
8	10	0.717
8	11	0.721
8	12	0.526
8	13	0.424
8	14	0.311
9	9	0.827
9	10	0.453
9	11	0.512
9	12	0.264
9	13	0.188
9	14	0.124
10	9	0.666
10	10	0.250
10	11	0.375
10	12	0.129
10	13	0.078
10	14	0.153
11	9	0.513
11	10	0.534
11	11	0.407
11	12	0.161
11	13	0.148
11	14	0.159
12	9	0.600
12	10	0.410
12	11	0.471
12	12	0.244
12	13	0.171
12	14	0.114
13	9	0.537
13	10	0.425
13	11	0.346
13	12	0.215
13	13	0.188
13	14	0.169

[1] For Source 1, HRC 1, the calibration software submitted to VQEG produced a spatial/temporal registration error that incorrectly estimated the processed video to be reframed (i.e. shifted by one field, see § 6.1.2 in Annex 5). For the other scenes of HRC 1, spatial/temporal registration was correctly estimated. § 6.1.5.7 in Annex 5, recommends median filtering of the calibration results over all scenes of a given HRC as a method to produce more robust calibration estimates for a given HRC. However, the VQEG Phase II test plan specified that all VQM software produce a single quality estimate for each clip independently. Thus, median filtering of calibration numbers over all scenes for a given HRC was not allowed by the test plan. Had median filtering of calibration numbers been allowed, the VQM software would have correctly registered this clip and the raw objective score would have been 0.529.

TABLE 16

625-line raw objective data

Source No.	HRC No.	NTIA: Proponent H
1	2	0.421
1	3	0.431
1	4	0.264
1	6	0.205
1	8	0.155
1	10	0.123
2	2	0.449
2	3	0.473
2	4	0.312
2	6	0.260
2	8	0.226
2	10	0.145
3	2	0.472
3	3	0.506
3	4	0.308
3	6	0.239
3	8	0.183
3	10	0.146
4	2	0.409
4	3	0.458
4	4	0.384
4	6	0.354
4	8	0.280
4	10	0.232
5	2	0.470
5	3	0.521
5	4	0.260
5	6	0.234
5	8	0.132
5	10	0.083
6	2	0.391
6	3	0.364
6	4	0.290
6	6	0.252
6	8	0.181
6	10	0.169
7	4	0.422
7	6	0.385
7	9	0.336
7	10	0.270
8	4	0.345
8	6	0.311
8	9	0.280
8	10	0.242
9	4	0.344
9	6	0.285
9	9	0.246
9	10	0.192
10	4	0.410
10	6	0.355
10	9	0.313
10	10	0.241
11	1	0.739
11	5	0.468
11	7	0.199
11	10	0.201
12	1	0.548
12	5	0.441
12	7	0.367
12	10	0.307
13	1	0.598
13	5	0.409
13	7	0.321
13	10	0.277

Appendix 1

Results of the Video Quality Expert Group FR-TV Phase II test

Introduction

The performance of the perceptual quality models included in this Recommendation was assessed through two parallel evaluations. In the first evaluation, a standard subjective method, the DSCQS method, was used to obtain subjective ratings of quality of video material by panels of human observers. In the second evaluation, objective ratings were obtained by the objective computational models. For each model, several metrics were computed to measure the accuracy and consistency with which the objective ratings predicted the subjective ratings.

This Appendix describes the subjective evaluation portion of the test as well as the results for the objective computational models submitted by the following proponents:

– Model 1 (British Telecom; identified in VQEG FR-TV Phase II as Proponent D);

– Model 2 (Yonsei University/Radio Research Laboratory/SK Telecom; identified in VQEG FR-TV Phase II as Proponent E);

– Model 3 (CPqD; identified in VQEG FR-TV Phase II as Proponent F);

– Model 4 (NTIA; identified in VQEG FR-TV Phase II as Proponent H).

Three independent laboratories conducted the subjective tests. Two laboratories, Communications Research Center (CRC, Canada) and Verizon (United States of America), performed the test with 525/60 Hz sequences and a third lab, Fondazione Ugo Bordoni (FUB, Italy), performed the test with 625/50 Hz sequences.

A detailed description of the Video Quality Expert Group FR-TV Phase II test is provided in the mentionned Document[1].

2 Video materials

The 525/60 or 625/50 line formats test video sequences were in Recommendation ITU-R BT.601 4:2:2 component video format using an aspect ratio of 4:3.

2.1 SRC and HRC

For each of the 525 and 625 tests, thirteen SRCs with different characteristics (e.g. format, temporal and spatial information, colour, etc.) were used (see Tables 17 and 18).

In both tests, the HRCs were chosen to represent typical conditions of secondary distribution of digitally encoded television quality video. In the 625 test, ten HRCs were used; their characteristics are presented in Table 19. In the 525 test, fourteen HRCs were used; their characteristics are presented in Table 20.

In both 625 and 525 tests, SRCs and HRCs were combined into a sparse matrix (see Tables 23-26).

TABLE 17

625/50 format sequences (SRCs)

SRC number	Characteristics
1	View of skyline taken from moving boat; originated as 16:9 film, telecined to 576i/50
2	Dancers on wood floor with fast motion, moderate detail; originally captured in D5 format
3	Indoor men's volleyball match; captured in D5 format
4	Women's soccer game action with fast camera panning; captured in D5
5	12 fps traditional animation; source converted to 24 fps film, then telecined to 576i/50
6	Slowly rotating wireframe globe; captured in DigiBetaCam
7	Rapid in-scene and camera motion, with lighting effects
8	Close-up of guitar being played, with changing light effects
9	Colour, motion, detail
10	High detail, textured background, motion
11	Colour, motion, detail
12	Outdoor rugby match; movement, colour
13	Motion, details, moving water
14 (demo)	Rapid in-scene and camera motion, with lighting effects
15 (demo)	Rapid in-scene and camera motion, with lighting effects
16 (demo)	Facial close-up followed by wide shot of construction site

TABLE 18

525/60 format sequences (SRCs)

SRC number	Characteristics
1	Outdoor football match, with colour, motion, textured background
2	Autumn landscape with detailed colour, slow zooming
3	Animation containing movement, colour and scene cuts
4	Highly detailed park scene with water; down converted from HDTV source
5	Colour and rapid motion; down converted from HDTV
6	Colour, large water surface; down converted from HDTV
7	Neighbourhood soccer match, moderate motion; down converted from HDTV
8	Water amusement park; (DigiBetaCam)
9	Amusement park ride with moderate motion, high detail, slow zoom; (DigiBetaCam)
10	Colour, motion, moderately low illumination; (DigiBetaCam)
11	12 fps traditional animation, converted to 24 fps film and telecined to 480i/60
12	Detailed outdoor fountain with camera zoom; (DigiBetaCam)
13	Scene cuts from close-up of engine ignition, to distant wide shot, and back; film original telecined to 480i/60
14 (demo)	Close-up shot of a rose in light breeze; motion, colour and detail; (DigiBetaCam)
15 (demo)	High detail, low motion; downconverted from HDTV
16 (demo)	Slowly rotating statues, swaying tree branches; (DigiBetaCam)

TABLE 19

625/50 HRCs

HRC number	Bit rate	Resolution	Method	Comments
1	768 kbit/s	CIF	H.263	Full screen (HRC15 from VQEG 1)
2	1 Mbits/s	320H	MPEG2	Proponent encoded
3	1.5 Mbit/s	720H	MPEG2	Encoded by FUB
4	2.5→4 Mbit/s	720H	MPEG2	Cascaded by FUB
5	2 Mbit/s	3/4	MPEG2, sp@ml	HRC13 from VQEG 1
6	2.5 Mbit/s	720H	MPEG2	Encoded by FUB
7	3 Mbit/s	full	MPEG2	HRC9 from VQEG 1
8	3 Mbit/s	704H	MPEG2	Proponent encoded
9	3 Mbit/s	720H	MPEG2	Encoded by FUB
10	4 Mbit/s	720H	MPEG2	Encoded by FUB

TABLE 20

525/60 HRCs

HRC number	Bit rate	Resolution	Method	Comments
1	768 kbit/s	CIF	H.263	Full screen (HRC15 from VQEG 1)
2	2 Mbit/s	3/4	MPEG2, sp@ml	HRC13 from VQEG 1
3	3 Mbit/s	full	MPEG2	HRC9 from VQEG 1
4	5 Mbit/s	720H	MPEG2	Encoded by CRC
5	2 Mbit/s	704H	MPEG2	Encoded by CRC
6	3 Mbit/s	704H	MPEG2	Encoded by CRC
7	4 Mbit/s	704H	MPEG2	Encoded by CRC
8	5 Mbit/s	704H	MPEG2	Encoded by CRC
9	1 Mbit/s	704H	MPEG2	Proponent encoded; low-bit rate combined with high resolution
10	1 Mbit/s	480H	MPEG2	Encoded by CRC; low-bit rate, low resolution
11	1.5 Mbit/s	528H	MPEG2	Proponent encoded; 64-QAM modulation; composite NTSC output converted to component
12	4→2 Mbit/s	720H	MPEG2	Proponent encoded; cascaded encoders
13	2.5 Mbit/s	720H	MPEG2	Encoded by CRC
14	4 Mbit/s	720H	MPEG2	Proponent encoded; using software codec

1 Methodology for the evaluation of objective model performance

The DSCQS method of Recommendation ITU-R BT.500 was used for subjective testing. For the 525 test, difference mean opinion scores (DMOS) were collected for 63 SRC×HRC combinations. For the 625, DMOS were collected for 64 SRC×HRC combinations. For the same SRC×HRC combinations, objective data were also obtained for each objective computational model.

For the purpose of model evaluation, the subjective data were scaled and the objective data were non-linearly transformed to a scale varying from 0 (not distinguishable from the source) to 1. The non-linear transformation was given by:

$$DMOS_p = b1/(1 + \exp(-b2*(VQR - b3)))$$

where:

VQR: actual output value of the objective computational model

$DMOS_p$: non-linearly transformed value.

Performance of the objective models was evaluated with respect to three aspects of their ability to estimate subjective assessment of video quality:

– prediction accuracy – the ability to predict the subjective quality ratings with low error;
– prediction monotonicity – the degree to which the model's predictions agree with the relative magnitudes of subjective quality ratings; and
– prediction consistency – the degree to which the model maintains prediction accuracy over the range of video test sequences, i.e. that its response is robust with respect to a variety of video impairments.

These attributes were evaluated through seven performance metrics which are described below.

Metric 1: The Pearson linear correlation coefficient between $DMOS_p$ and *DMOS*.

Metric 2: Spearman rank order correlation coefficient between $DMOS_p$ and *DMOS*.

The Spearman correlation and the Pearson correlation and all other statistics were calculated across all SRC by HRC combinations simultaneously.

Metric 3: Outlier Ratio of "outlier-points" to total points *N*.

$$\text{Outlier Ratio} = (\text{Total number of outliers})/N$$

where an outlier is a point for which: ABS[Qerror[i]] > 2∗DMOSStandardError[i].

Twice the DMOS Standard Error was used as the threshold for defining an outlier point.

Metric 4, 5, 6: (These metrics were evaluated based on the method described in Report T1.TR.72-2001 [ATIS, 2001]).

4: RMS Error,

5: Resolving Power, and

6; Classification Errors.

Note that evaluation of models using this method omitted the cross-calibration procedure described therein, as it is not relevant to measures of performance of individual models.

Metric 7: This metric is based on the *F*-test. Two *F*-test measures were computed. The first *F* measurement used the Mean Square Error (MSE) computed from individual subject ratings. MSE were obtained for a "null or optimal" model, which corresponded to the observed DMOS and associated residuals, and for each one of the objective models. *F*-tests were performed to compare the MSE of the null model to that of each model, and the MSE of the best performing model to that of the other models. The second *F* measurement was based on the MSE computed from average ratings, i.e. DMOS. Specifically, MSE were computed for each model using the residuals between predicted and observed DMOS. *F*-tests were performed to compare the MSE of the best performing model to that of the other models.

4 Evaluation of results

The results of the metric calculations are presented in Tables 21 and 22, one for the 525-line data and one for the 625-line data.

All seven metrics in the tables agree almost perfectly. An objective model that performs well under one metric does generally so also for the other metrics, and vice versa. Furthermore, the ranking of the objective models by the different metrics is essentially identical within each of the two video formats. However, the results of the two tests (525 and 625) are similar but not identical. There were a few apparent changes in ranking from one experiment to the other.

The subjective scaled data used to compute these metrics are presented in Tables 23-26. The corresponding objective data obtained by the four objective computational models are presented in Annexes 2-5.

5 PSNR data

The PSNR is a simple video quality metric. The performance of the VQMs can be compared to the performance of PSNR. PSNR for the test sequences was calculated by several proponents. Metrics for the highest PSNR are reported in Tables 21 and 22.

TABLE 21

Summary of 525 analyses

Line number	Metric	D525	E525	F525	H525	PSNR525
1	1. Pearson correlation	0.937	0.857	0.835	0.938	0.804
2	2. SPEARMAN CORRELATION	0.934	0.875	0.814	0.936	0.811
3	3. Outlier ratio	33/63 = 0.52	44/63 = 0.70	44/63 = 0.70	29/63 = 0.46	46/63 = 0.73
4	4. RMS error, 63 data points	0.075	0.11	0.117	0.074	0.127
5	5. Resolving power, delta VQM (smaller is better)	0.2177	0.2718	0.3074	0.2087	0.3125
6	6. Percentage of classification errors (Minimum over delta VQM)	0.1889	0.2893	0.3113	0.1848	0.3180
7	7. MSE model/MSE optimal model	1.262	1.59	1.68	1.256	1.795
8	F = MSE model/MSE Proponent H	1.005	1.266	1.338	1	1.429
9	MSE model, 4219 data points	0.02421	0.03049	0.03223	0.02409	0.03442
10	MSE optimal model, 4219 data points	0.01918	0.01918	0.01918	0.01918	0.01918
11	MSE model, 63 data points	0.00559	0.01212	0.01365	0.00548	0.01619
12	F= MSE63 model/MSE63 Prop H	1.02	2.212	2.491	1	2.954

NOTE 1 – Metrics 5 and 6 were computed using the Matlab® code published in T1.TR.72-2001.

NOTE 2 – Metric 5 estimated by eye from scatter plots in output documents.

NOTE 3 – Values of Metric 7 smaller than 1.07 indicate the model is not reliably different from the optimal model.

NOTE 4 – Values in line 8 larger than 1.07 indicate the model has significantly larger residuals than the top proponent model, *H* in this case.

NOTE 5 – Values in line 12 larger than 1.81 indicate the model has significantly larger residuals than the top proponent model, *H* in this case.

TABLE 22

Summary of 625 analyses

Line number	Metric	D625	E625	F625	H625	PSNR625
1	1. Pearson correlation	0.779	0.87	0.898	0.886	0.733
2	2. Spearman correlation	0.758	0.866	0.883	0.879	0.74
3	3. Outlier ratio	28/64 = 0.44	24/64 = 0.38	21/64 = 0.33	20/64 = 0.31	30/64 = 0.47
4	4. RMS error, 64 data points	0.113	0.089	0.079	0.083	0.122
5	5. Resolving power, delta VQM (smaller is better)	0.321	0.281	0.270	0.267	0.313
6	6. Percentage of classification errors (Minimum over delta VQM)	0.305	0.232	0.204	0.199	0.342
7	7. MSE model/MSE null model	1.652	1.39	1.303	1.339	1.773
8	F = MSE model/MSE Proponent F	1.268	1.067	1	1.028	1.361
9	MSE model, 1728 data points	0.02953	0.02484	0.02328	0.02393	0.03168
10	MSE null model, 1728 data points	0.01787	0.01787	0.01787	0.01787	0.01787
11	MSE model, 64 data points	0.0127	0.00786	0.00625	0.00693	0.01493
12	F= MSE64 model/MSE64 Prop F	2.032	1.258	1	1.109	2.389

NOTE 1 – Metrics 5 and 6 were computed using the Matlab® code published in T1.TR.72-2001.

NOTE 2 – Metric 5 estimated by eye from scatter plots in output documents.

NOTE 3 – Values of Metric 7 smaller than 1.12 indicate the model is not reliably different from the optimal model.

NOTE 4 – Values in line 8 larger than 1.12 indicate the model has significantly larger residuals than the top proponent model, *F* in this case.

NOTE 5 – In the case of the 625 data with 1728 observations, the critical value of the *F* statistic is 1.12.

NOTE 6 – Values in line 12 larger than 1.81 indicate the model has significantly larger residuals than the top proponent model, *F* in this case.

TABLE 23

Subjective data for all 525/60 HRC-SRC combinations – (DMOS values)

SRC (image)	HRC													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	0.5402368	0.5483205	0.4024097	0.3063528										
2	0.5025558	0.3113346	0.1881739	0.1907347										
3	0.4682724	0.3088831	0.1300389	0.1293293										
4					0.6742005	0.4250873	0.3762656	0.2972294						
5					0.4682559	0.3203024	0.2071702	0.1652752						
6					0.5690291	0.4370961	0.3591788	0.2482169						
7					0.3796362	0.2276934	0.1644409	0.1819566						
8									0.9513387	0.789748	0.8405916	0.5221555	0.4572049	0.4614104
9									0.8262912	0.660339	0.7100111	0.4921708	0.3656559	0.2960957
10									0.9084171	0.5908784	0.7302376	0.3345703	0.2565459	0.2953144
11									0.6675853	0.7054929	0.5761193	0.32761	0.310495	0.331051
12									0.7883371	0.6295301	0.6809288	0.3651402	0.2714356	0.2782449
13									0.7211194	0.5545722	0.5525494	0.2708744	0.27549	0.2733771

NOTE 1 – The SRC = 6, HRC = 5 value was taken out of the analysis because it exceeded the temporal registration requirements of the test plan.

TABLE 24

Subjective data for all 625/50 HRC-SRC combinations – (DMOS values)

SRC (image)	HRC									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1		0.59461	0.64436	0.40804		0.34109		0.2677		0.26878
2		0.54173	0.70995	0.27443		0.22715		0.21133		0.16647
3		0.73314	0.76167	0.49848		0.38613		0.34574		0.26701
4		0.58528	0.90446	0.62361		0.61143		0.43329		0.26548
5		0.61973	0.68987	0.41648		0.4218		0.27543		0.2022
6		0.38852	0.44457	0.27983		0.28106		0.23726		0.17793
7				0.59953		0.55093			0.45163	0.35617
8				0.32528		0.32727			0.30303	0.26366
9				0.47656		0.49924			0.39101	0.37122
10				0.70492		0.58218			0.49711	0.37854
11	0.79919				0.59256		0.34337			0.30567
12	0.61418				0.6661		0.53242			0.44737
13	0.74225				0.66799		0.42065			0.33381

TABLE 25

Subjective data for all 525/60 HRC-SRC combinations – (standard errors values)

SRC (image)	HRC													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	0.02109499	0.0223858	0.0202654	0.0200377										
2	0.02072424	0.0186353	0.0164296	0.0179823										
3	0.02075164	0.021336	0.0131301	0.0141977										
4					0.0224479	0.0200094	0.0221945	0.0216022						
5					0.0254351	0.0217278	0.0179396	0.0145813						
6						0.0215159	0.0176766	0.0180308						
7					0.0197204	0.0171224	0.0147712	0.0188843						
8									0.010892	0.0180687	0.0185947	0.0249537	0.0272349	0.0258362
9									0.0167711	0.018702	0.0281708	0.0226776	0.0193788	0.0203533
10									0.0144376	0.0263593	0.0171287	0.0202314	0.01996	0.018688
11									0.0186046	0.0189571	0.0213137	0.0188185	0.020292	0.0183653
12									0.0175106	0.0223805	0.0216039	0.0192717	0.0183	0.0202472
13									0.0213225	0.023069	0.0238845	0.0196748	0.0187747	0.0201108

NOTE 1 – To convert to standard deviations, multiply by the square root of the number of observations, 66.

NOTE 2 – The SRC = 6, HRC = 5 value was taken out of the analysis because it exceeded the temporal registration requirements of the test plan.

TABLE 26

Subjective data for all 625/60 HRC-SRC combinations (standard errors values)

SRC (image)	HRC									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1		0.040255	0.039572	0.038567		0.040432		0.040014		0.036183
2		0.038683	0.033027	0.040957		0.038301		0.042618		0.033956
3		0.039502	0.039111	0.039109		0.042553		0.044151		0.036685
4		0.031762	0.024408	0.036375		0.031371		0.02973		0.042911
5		0.034299	0.044757	0.0407		0.03597		0.033742		0.041272
6		0.040602	0.040035	0.03707		0.043341		0.035289		0.040621
7				0.037894		0.032156		0.038034		0.036946
8				0.036819		0.041563		0.036988		0.037467
9				0.040289		0.040265		0.04015		0.039649
10				0.030283		0.038334		0.037966		0.041339
11	0.034761				0.034838		0.041778			0.041516
12	0.037332				0.036964		0.031253			0.035114
13	0.035205				0.038385		0.038371			0.043687

NOTE 1 – To convert to standard deviations, multiply by the square root of the number of observations, 27.

6 References

ATIS [October 2001] Technical Report T1.TR.72-2001 – Methodological Framework for Specifying Accuracy and Cross-Calibration of Video Quality Metrics, Alliance for Telecommunications Industry Solutions, 1200 G Street, NWn Suite 500, Washington DC.