International Telecommunication Union

# ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

**J.144**
(03/2004)

SERIES J: CABLE NETWORKS AND TRANSMISSION OF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS

Measurement of the quality of service

# Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference

Recommendation ITU-T J.144

# Recommendation ITU-T J.144

## Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference

### Summary

Recommendation ITU-T J.144 provides guidelines on the selection of appropriate objective perceptual video quality measurement equipment designed for use in digital cable television applications when the full reference video signal is available. The validation test material did not contain channel errors. This Recommendation defines objective computational models that have been shown to be superior to peak signal to noise ratio (PSNR) as automatic measurement tools for assessing the quality of video.

This revised Recommendation includes four objective methods for the assessment of perceptual video quality in the normative section.

### Source

Recommendation ITU-T J.144 was approved on 15 March 2004 by ITU-T Study Group 9 (2001-2004) under Recommendation ITU-T A.8 procedure.

## FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

## NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g. interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

## INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

© ITU 2009

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

## CONTENTS

## Introduction

Digital television produces new quality of service considerations, with complex relationships between objective parameter measurements and subjective picture quality. While objective measurements with good correlation to subjective quality assessment are desirable in order to attain optimal quality of service in the operation of cable television systems, it must be realized that objective measurements are not a direct replacement for subjective quality assessment.

Subjective quality assessments are carefully designed procedures intended to determine the average opinion of human viewers to a specific set of video sequences for a given application. Results of such tests are valuable in basic system design and benchmark evaluations. Subjective quality assessments for a different application with different test conditions will still provide meaningful results; however, opinion scores for the same set of video sequences are likely to have different values. Objective measurements are intended for use in a broad set of applications producing the same results with a given set of video sequences. The choice of video sequences to use and the interpretation of the resulting objective measurements are some of the factors which vary for a specific application.

Therefore, objective measurements and subjective quality assessment are complementary rather than interchangeable. Where subjective assessment is appropriate for research related purposes, objective measurements are required for equipment specifications and day-to-day system performance measurement and monitoring.

The following terminology convention is adopted for the purpose of this Recommendation:

– The term "subjective assessment" refers to the determination of the quality or impairment of programme-like pictures presented to a panel of human assessors in viewing sessions.

– The term "objective perceptual measurement" refers to the measurement of the performance of a programme chain by the use of programme-like pictures and objective (instrumental) measurement methods to obtain an indication that approximates the rating that would be obtained from a subjective assessment test.

– The term "signal measurement" refers to the measurement of the performance of a programme chain by the use of test signals and objective (instrumental) measurement methods.

In this Recommendation, the terms "objective measurement" and "perceptual measurement" may be used interchangeably to mean objective perceptual measurement.

There are three basic methods to perform objective measurements:

- FR – A method applicable when the full reference video signal is available. This is a double-ended method and is the subject of this Recommendation.
- RR – A method applicable when only reduced video reference information is available. This is also a double-ended method and is the subject of a separate Recommendation (under study).
- NR – A method applicable when no reference video signal or information is available. This is a single-ended method and the subject of a separate Recommendation (under study).

The three methods have different applications, and they provide different degrees of measurement accuracy, expressed in terms of correlation with subjective assessment results.

# Recommendation ITU-T J.144

# Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference

## 1 Scope

This Recommendation provides guidelines on the selection of appropriate perceptual video quality measurement equipment for use in digital cable television applications when the full reference measurement method can be used.

The full reference measurement method can be used when the unimpaired reference video signal is readily available at the measurement point, as may be the case of measurements on individual equipment, or a chain in the laboratory, or in a closed environment such as a cable television head-end. The estimation methods are based on processing 8-bit digital component video as defined by ITU-R Rec. BT.601-5[1]. The encoder can utilize various compression methods (e.g., MPEG, 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. It contained coding degradations and the compression rates were 768 kbit/s-5 Mbit/s.

NOTE – The structure and content of this Recommendation have been organized for case of use by those familiar with the original source material; as such, the usual style of ITU-T Recommendation has not been applied.

### 1.1 Application

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

1) codec evaluation, specification, and acceptance testing, consistent with the limited accuracy as described below;
2) potentially real-time, in-service quality monitoring at the source;
3) remote destination quality monitoring when a copy of the source is available;
4) 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 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.

[1] 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 specifies one method for performing this conversion for NTSC.

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

## 2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

### 2.1 Normative references

– ITU-R Recommendation BT.601-5 (1995), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios.*

### 2.2 Informative references

– ITU-T Recommendation J.140 (1998), *Subjective picture quality assessment for digital cable television systems*.

– ITU-T Recommendation J.143 (2000), *User requirements for objective perceptual video quality measurements in digital cable television*.

– ITU-T Recommendation P.910 (1996), *Subjective video quality assessment methods for multimedia applications*.

– ITU-T Tutorial (2004), *Objective perceptual assessment of video quality: Full reference television*.

– ITU-T Recommendation P.911 (1998), *Subjective audiovisual quality assessment methods for multimedia applications.*

– U. S. Standards Committee T1* Technical Report T1.TR.73-2001, *Video normalization methods applicable to objective video quality metrics utilizing a full reference technique.*

– ITU-R Recommendation BT.500-11 (2002), *Methodology for the subjective assessment of the quality of television pictures*.

## 3 Terms, definitions and acronyms

This Recommendation defines the following terms:

**3.1 subjective assessment (picture)**: The determination of the quality or impairment of programme-like pictures presented to a panel of human assessors in viewing sessions.

**3.2 objective perceptual measurement (picture)**: The measurement of the performance of a programme chain by the use of programme-like pictures and objective (instrumental) measurement methods to obtain an indication that approximates the rating that would be obtained from a subjective assessment test.

**3.3 signal measurement (picture)**: The measurement of the performance of a programme chain by the use of test signals and objective (instrumental) measurement methods.

**3.4 ANOVA**: ANalysis Of VAriance.

* T1 standards are maintained since November 2003 by ATIS.

**3.5** **FRTV**: Full Reference TeleVision.

**3.6** **DSCQS**: Double Stimulus Continuous Quality Scale.

**3.7** **proponent**: An organization or company that proposes a video quality model for validation testing and possible inclusion in an ITU Recommendation.

## 4 User requirements

User requirements for perceptual measurement methods of picture quality are given in ITU-T Rec. J.143.

## 5 Description of the full reference measurement method

The double-ended measurement method with full reference, for objective measurement of perceptual video quality, evaluates the performance of systems by making a comparison between the undistorted input, or reference, video signal at the input of the system, and the degraded signal at the output of the system (Figure 1).

Figure 1 shows an example of application of the full reference method to test a codec in the laboratory.

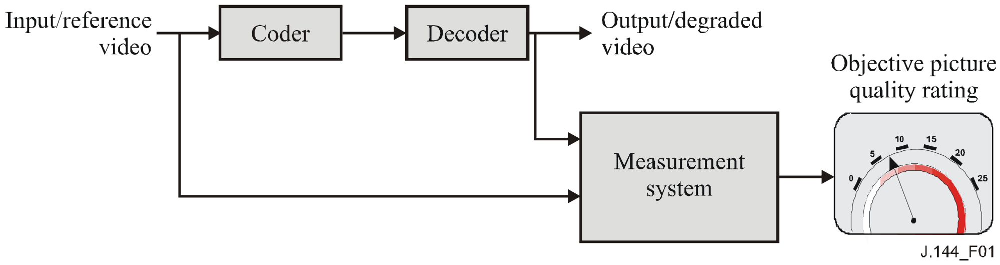


**Figure 1 – Application of the full reference perceptual quality measurement method to test a codec in the laboratory**

The comparison between input and output signals may require a temporal alignment or a spatial alignment process, the latter to compensate for any vertical or horizontal picture shifts or cropping. It also may require correction for any offsets or gain differences in both the luminance and the chrominance channels. The objective picture quality rating is then calculated, typically by applying a perceptual model of human vision.

Alignment and gain adjustment is known as normalization. This process is required because most full reference methods compare reference and processed pictures on what is effectively a pixel-by-pixel basis. An example would be calculation of peak signal to noise ratio (PSNR). Only time-invariant static changes in the video are removed, dynamic changes due to system under test are measured as part of the quality rating calculation. A detailed discussion of the reasons for normalization is available in U.S. Standards Committee T1 Technical Report T1.TR.73-2001, "*Video normalization methods applicable to objective video quality metrics utilizing a full reference technique*". The video quality metrics described in Annexes A through D include associated normalization methods. Alternate normalization methods can be used for the video quality metrics of Annexes A through D and Appendices I and II provided they deliver the required normalization accuracy.

As the video quality metrics are typically based on approximations to human visual responses, rather than on the measurement of specific coding artefacts, they are in principle equally valid for analogue systems and for digital systems. They are also, in principle valid for chains where analogue and digital systems are mixed, or where digital compression systems are concatenated.

Figure 2 shows an example of the application of the full reference method to test a transmission chain.

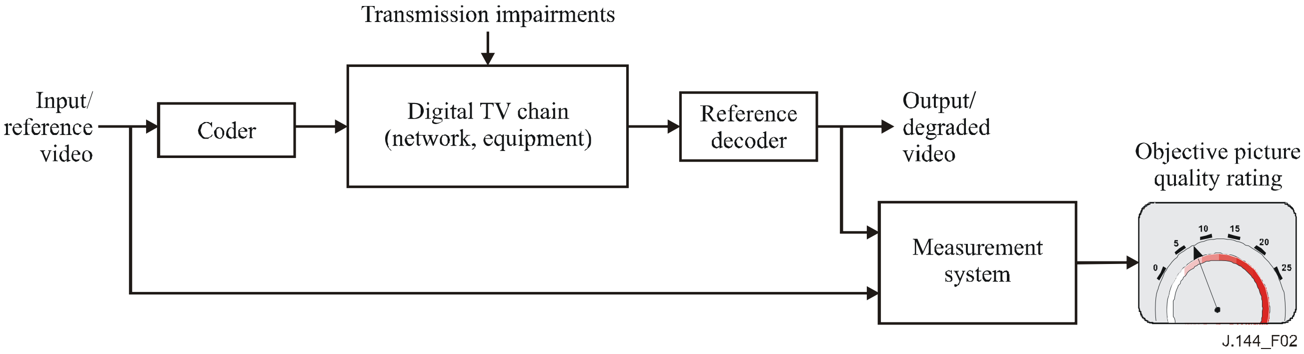


**Figure 2 – Application of the full reference perceptual quality measurement method to test a transmission chain**

In this case, a reference decoder is fed from various points in the transmission chain, e.g., the decoder can be located at a point in the network, as in Figure 2, or directly at the output of the encoder as in Figure 1. If the digital transmission chain is transparent, the measurement of objective picture quality rating at the source is equal to the measurement at any subsequent point in the chain.

It is generally accepted that the full reference method provides the best accuracy for perceptual picture quality measurements. The method has been proven to have the potential for high correlation with subjective assessments made in conformity with the DSCQS methods specified in ITU-R Rec. BT.500-11.

## 6 Findings of the Video Quality Experts Group (VQEG)

Studies of perceptual video quality measurements are conducted in an informal group, called Video Quality Experts Group (VQEG), which reports to ITU-T Study Group 9 and ITU-R Study Group 6. The first task of VQEG was to assess the performance of proposed full reference perceptual video quality measurement algorithms.

VQEG issued a comprehensive final draft report on the first phase of its work in March 2000. VQEG issued a final report of the Phase II Full Reference Television Test in August of 2003.

Readers are advised to study those reports to gain complete insight on the work performed by VQEG. The aim was to check proponent models in terms of:

- prediction accuracy (the model's ability to predict the subjective quality);
- prediction monotonicity (the degree to which the model's predictions agree with the rank ordering of subjective quality ratings);
- prediction consistency (the degree to which the model maintains prediction accuracy over the range of video test sequences and video systems, i.e., that its response is robust with respect to a variety of video impairments).

VQEG's FRTV validation test Phase I did not provide sufficient evidence to identify a method to be recommended for objective assessment of perceptual video quality. VQEG's FRTV validation test Phase II produced results that indicate that four of the methods are appropriate for inclusion in the Normative part of this Recommendation.

In Phase II, strict adherence to BT.500-11 procedures for the Double Stimulus Continuous Quality Scale (DSCQS) method was followed in the subjective evaluation. The subjective and objective test plans included procedures for validation analysis of the subjective scores, and four metrics for comparing the objective data to the subjective results. Further statistical analysis included ANOVA and the F-test.

Based on present evidence, four methods can be recommended to ITU at this time. These are:

Annex A − British Telecom (United Kingdom, VQEG Proponent D);

Annex B − Yonsei University / SK Telecom / Radio Research Laboratory (Republic of Korea, VQEG Proponent E);

Annex C − CPqD (Federative Republic of Brazil, VQEG Proponent F);

Annex D − NTIA (United States of America, VQEG Proponent H).

The technical descriptions of these models can be found in Annexes A through D respectively. Note that the ordering of annexes is purely arbitrary and provides no indication of quality prediction performance. For example, in absolute terms NTIA's model produced the best correlation with subjective ratings in the 525 test.

Tables 1 and 2 provide informative details on the models' performances in the VQEG Phase II FRTV test. For the 525 data, models NTIA and BT performed statistically better than the other models and are statistically equivalent to each other. For the 625 data, three models (CPqD, NTIA, Yonsei/SKT/RRL) are statistically equivalent to each other and are statistically better than the other model. It is also noted that only the NTIA model statistically tied for top performances in both tests.

**Table 1 – Informative description on the models' performances in the VQEG Phase II FRTV test (525 data)**

| Metric | BT | Yonsei/SKT/RRL | CPqD | NTIA | PSNR (Note) |
|---|---|---|---|---|---|
| Annex | A | B | C | D | |
| Pearson correlation | 0.937 | 0.857 | 0.835 | 0.938 | 0.804 |
| RMS error | 0.075 | 0.110 | 0.117 | 0.074 | 0.127 |
| NOTE – The PSNR values reported here are taken from the VQEG Phase II final report. These values were calculated by Yonsei. | | | | | |

**Table 2 – Informative information on the models' performances in the VQEG Phase II FRTV test (625 data)**

| Metric | BT | Yonsei / SKT / RRL | CPqD | NTIA | PSNR |
|---|---|---|---|---|---|
| Annex | A | B | C | D | |
| Pearson correlation | 0.779 | 0.870 | 0.898 | 0.886 | 0.733 |
| RMS error | 0.113 | 0.089 | 0.079 | 0.083 | 0.122 |

## 7 Conclusions

### 7.1 Recommendation general advice

When perceptual video quality measurements are performed, using the full reference method described in this Recommendation, operators should first analyse how their specific application and user requirements translate in terms of measuring equipment characteristics and performance.

Some aspects to be taken into consideration are listed below:

- ownership cost of the perceptual measurement equipment;
- vendor's after-sale support;
- ease of operation;

- reliability;
- size, weight, power requirements;
- real-time or non-real-time measurement speed;
- online (in-service) operation;
- prediction accuracy, monotonicity and consistency.

The four normative methods are recommended because of their high correlation to subjective results in Phase II of VQEG's FRTV testing. However, until those methods are available in commercial test equipment the Tektronix/Sarnoff and KDDI methods are recommended. These two legacy video quality metrics are specified in Appendices I and II. They were validated in VQEG Phase I and are included due to their use in a significant installed base of video quality measurement instruments. The video quality metrics tested in VQEG Phase I were not considered to be accurate enough to be included as normative in an ITU Recommendation. However, it should be noted that VQEG Phase I used a larger variety of test conditions, many with very small degradations, as compared to those of VQEG Phase II. VQEG Phase II contained test conditions utilizing the discrete cosine transform (i.e., MPEG-2 and H.263) with bit-rates between 768 kbit/s to 5 Mbit/s.

### 7.2 Objective video quality models – Pathway to future revisions

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 Study Group (SG) 9. The intention of SG 9 is to eventually recommend only one normative full reference method for cable television.

# Annex A

## British Telecommunications plc

## Full reference video quality model functional description

### A.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 [A-1].

### A.2 BT full-reference model

The BTFR algorithm consists of detection followed by integration as shown in Figure A.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.

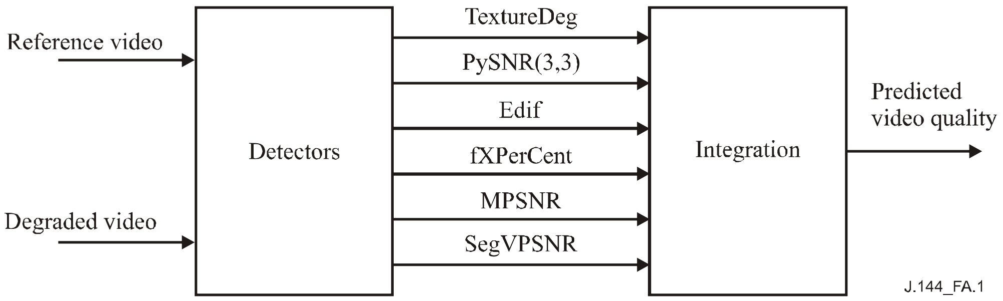


**Figure A.1 – Full-reference video quality assessment model**

Input video of types 625 (720 × 576) interlaced at 50 fields/s and 525 (720 × 486) interlaced at 59.94 fields/s in YUV422 format are supported by the model.

### A.3 Detectors

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

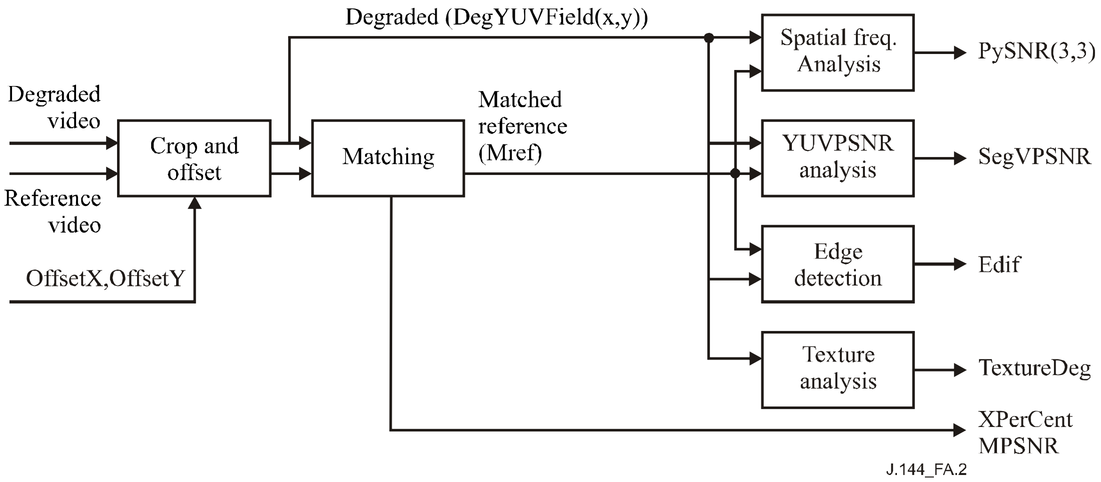


**Figure A.2 – Detection**

### A.3.1 Input conversion

First, the input sequences are converted from YUV422 interlaced format to a block YUV444 deinterlaced format so that each successive field is represented by arrays Re*fY*, Re*fU* and Re*fV*:

$$\mathrm{Re}\, fY(x,y) \quad x=0..X-1, \quad y=0..Y-1 \qquad \text{(A.3.1-1)}$$

$$\mathrm{Re}\, fU(x,y) \quad x=0..X-1, \quad y=0..Y-1 \qquad \text{(A.3.1-2)}$$

$$\mathrm{Re}\, fV(x,y) \quad x=0..X-1, \quad y=0..Y-1 \qquad \text{(A.3.1-3)}$$

where *X* is the number of horizontal pixels within a field and *Y* the number of vertical pixels. For a YUV422 input, each *U* and *V* value must be repeated to give the full resolution arrays (equations A.3.1-2 and A.3.1-3).

### A.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 OffsetX and OffsetY are determined externally and define the number of horizontal and vertical pixels 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 YUV444 format (see A.3.1) in arrays InYField, InUField, and InVField the cropped and offset output is calculated according to equations A.3.2-1 to A.3.2-17.

$$XStart = -OffsetX \qquad \text{(A.3.2-1)}$$

$$if\,(XStart < C_x)\ then \quad XStart = C_x \qquad \text{(A.3.2-2)}$$

$$XEnd = X - 1 - OffsetX \qquad \text{(A.3.2-3)}$$

$$if\,(XEnd > X - C_x - 1)\ then \quad XEnd = X - C_x - 1 \qquad \text{(A.3.2-4)}$$

$$YStart = -OffsetY \qquad \text{(A.3.2-5)}$$

$$if\,(YStart < C_y)\ then \quad YStart = C_y \qquad \text{(A.3.2-6)}$$

$$YEnd = Y - 1 - OffsetY \qquad \text{(A.3.2-7)}$$

$$if\,(YEnd > Y - C_y - 1)\ then\quad YEnd = Y - C_y - 1 \qquad \text{(A.3.2-8)}$$

*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,

$$X = 720,\qquad Y = 288,\qquad C_x = 30,\qquad C_y = 10 \qquad \text{(A.3.2-9)}$$

For 525 sequences,

$$X = 720,\qquad Y = 243,\qquad C_x = 30,\qquad C_y = 10 \qquad \text{(A.3.2-10)}$$

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 A.3.2-11 to A.3.2-12, 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:

$$YField(x,y) = 0 \quad x = 0..XStart - 1, XEnd + 1..X - 1 \quad y = 0..Y - 1 \qquad \text{(A.3.2-11)}$$

$$UField(x,y) = VField(x,y) = 128 \quad x = 0..XStart - 1, XEnd + 1..X - 1 \quad y = 0..Y - 1 \qquad \text{(A.3.2-12)}$$

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

$$YField\,(x,y) = 0 \quad x = XStart\,..XEnd\,,\quad y = 0..YStart\ - 1, YEnd\ + 1..Y - 1 \qquad \text{(A.3.2-13)}$$

$$UField(x,y) = VField(x,y) = 128 \quad x = XStart..XEnd \quad y = 0..YStart - 1, YEnd + 1..Y - 1 \qquad \text{(A.3.2-14)}$$

Finally, the pixel values are copied according to:

$$YField(x,y) = InYField(x + OffsetX, y + OffsetY) \quad x = XStart..XEnd \quad y = YStart..YEnd \qquad \text{(A.3.2-15)}$$

$$UField(x,y) = InUField(x + OffsetX, y + OffsetY) \quad x = XStart..XEnd \quad y = YStart..YEnd \qquad \text{(A.3.2-16)}$$

$$VField(x,y) = InVField(x + XOffset, y + YOffset) \quad x = XStart..XEnd \quad y = YStart..YEnd \qquad \text{(A.3.2-17)}$$

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.

### A.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:

$$\mathrm{Re}\,f\,(N, Px, Py) = \mathrm{Re}\,fYField(Px, Py) \qquad from \quad field \quad N \qquad \text{(A.3.3-1)}$$

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

$$Block\,\mathrm{Re}\,f\,(N, Px, Py) = \mathrm{Re}\,f\,(n, x, y) \quad x = Px - 4..Px + 4,\quad y = Py - 4..Py + 4 \qquad \text{(A.3.3-2)}$$

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:

$$E(N,Px,Py)=MIN((1/81)\sum_{j=-4}^{4}\sum_{k=-4}^{4}(Deg(N,Px+j,Py+k)-\mathrm{Re}\,f(n,x+j,y+k))^2) \qquad \text{(A.3.3-3)}$$

with

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

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

If equation A.3.3-3 determines the best match for BlockDeg($N, Px, Py$) to be BlockRef($n_m, x_m, y_m$), then a matched reference array MRef is updated according to:

$$M\,\mathrm{Re}\,f(N,Px+j,Py+k)=\mathrm{Re}\,f(n_m,x_m+j,y_m+k) \quad j=-4...4, k=-4...4 \qquad \text{(A.3.3-4)}$$

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:

$$Px(h)=16+8\times h \quad h=0..Qx-1 \qquad \text{(A.3.3-5)}$$

and

$$Py(v)=16+8\times v \quad v=0..Qy-1 \qquad \text{(A.3.3-6)}$$

where $Qx$ and $Qy$ define the number of horizontal and vertical analysis blocks. Because the matching process is based on $9 \times 9$ pixel blocks, adjacent matching blocks overlap by one pixel. MRef is updated according to equations A.3.3-4, A.3.3-5, and A.3.3-6 whereby overlapping regions within MRef are overwritten by the results of subsequent calculations.

The matching analysis of the $N$th field, therefore, produces a matched reference sequence described by

$$BlockM\,\mathrm{Re}\,f(N,Px(h),Py(v)) \qquad h=0..Qx-1, \qquad v=0..Qy-1 \qquad \text{(A.3.3-7)}$$

and a set of best match error values

$$E(N,Px(h),Py(v)) \qquad h=0..Qx-1, \qquad v=0..Qy-1 \qquad \text{(A.3.3-8)}$$

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

$$\begin{aligned} BlockM\,\mathrm{Re}\,f(N,Px(h),Py(v)) &= Block\,\mathrm{Re}\,f(MatT(h,v),MatX(h,v),MatY(h,v)) \\ & h=0..Qx-1, \qquad v=0..Qy-1 \end{aligned} \qquad \text{(A.3.3-9)}$$

The matching parameters for 625- and 525- broadcast sequences are given in Table A.1.

**Table A.1 – Search parameters for matching procedure**

| Parameter | 625 | 525 |
|---|---|---|
| *Qx* | 87 | 87 |
| *Qy* | 33 | 28 |

The analysis region defined by equations A.3.3-6 and A.3.3-7 does not cover the complete field size. *M*Re*f* must, therefore, be initialized according to equation A.3.3-9 so that it may be used elsewhere unrestricted.

$$M\,\mathrm{Re}\,f(x,y)=0 \qquad x=0..X-1, \qquad y=0..Y-1 \qquad \text{(A.3.3-10)}$$

### A.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 A.3.3-3, is used in the construction of the histogram *histX* for each field according to:

$$\begin{gathered} histX\left(MatX(h,v)-Px(h)+4\right)=histX\left(MatX(h,v)-Px(h)+4\right)+1 \\ h=0..Qx-1, \quad v=0..Qy-1 \end{gathered} \qquad \text{(A.3.3.1-1)}$$

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

$$fXPerCent=100\times Max(histX(i))/\sum_{j=0}^{8} histX(j) \qquad i=0..8 \qquad \text{(A.3.3.2-2)}$$

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

### A.3.3.2 Matched PSNR

The minimum error, *E*(), for each matched block is used to calculate a matched signal-to-noise ratio according to:

$$\begin{gathered} if\left(\sum_{h=0}^{Qx-1}\sum_{v=0}^{Qy-1} E(N,Px(h),Py(v))\right)>0 \qquad then \\ MPSNR=10\log_{10}(Qx\times Qy\times 255^2/\sum_{h=0}^{Qx-1}\sum_{v=0}^{Qy-1} E(N,Px(h),Py(v))) \end{gathered} \qquad \text{(A.3.3.2-1)}$$

$$if\left(\sum_{h=0}^{Qx-1}\sum_{v=0}^{Qy-1} E(N,Px(h),\; Py(v))\right)=0 \quad then \quad MPSNR=10\log_{10}(255^2) \qquad \text{(A.3.3.2-2)}$$

### A.3.3.3 Matching vectors

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

$$SyncT(h,v)=MatT(h,v)-N \qquad h=0..Qx-1, \qquad v=0..Qy-1 \qquad \text{(A.3.3.3-1)}$$

$$SyncX(h,v)=MatX(h,v)-Px(h) \qquad h=0..Qx-1, \qquad v=0..Qy-1 \qquad \text{(A.3.3.3-2)}$$

$$SyncY(h,v)=MatY(h,v)-Py(h) \qquad h=0..Qx-1, \qquad v=0..Qy-1 \qquad \text{(A.3.3.3-3)}$$

## A.3.4 Spatial frequency analysis

The spatial frequency detector is based on a "pyramid" transformation of the degraded and matched reference sequences (see Figure A.3). 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 signal-to-noise ratio.

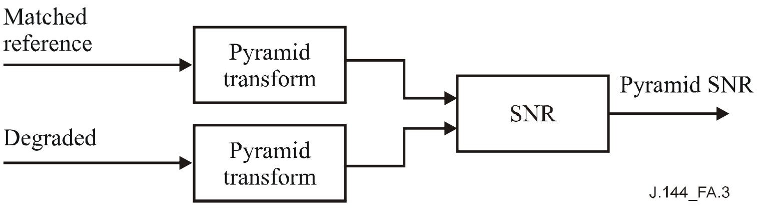


**Figure A.3 – Spatial frequency analysis**

### A.3.4.1 Pyramid transform

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

$$P(x,y)=F(x,y) \qquad x=0..X-1, \quad y=0..Y-1 \qquad \text{(A.3.4.1-1)}$$

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 A.3.4.1-2 to A.3.4.1-6.

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

$$PTemp(x,y)=P(x,y) \qquad x=0..X-1, \qquad y=0..Y-1 \qquad \text{(A.3.4.1-2)}$$

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

$$Tx=X/2^{(stage+1)} \qquad \text{(A.3.4.1-3)}$$

$$Ty=Y/2^{stage} \qquad \text{(A.3.4.1-4)}$$

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

$$P(x,y)=0.5\times(PTemp(2x,y)+PTemp(2x+1,y)) \quad x=0..Tx-1 \quad y=0..Ty-1 \qquad \text{(A.3.4.1-5)}$$

$$P(x+Tx,y)=PTemp(2x,y)-PTemp(2x+1,y) \quad x=0..Tx-1 \quad y=0..Ty-1 \qquad \text{(A.3.4.1-6)}$$

The vertical analysis Vpy(stage) is defined by equations (A.3.4.1-7) to (A.3.4.1-11).

$$PTemp(x,y)=P(x,y) \qquad x=0..X-1, \qquad y=0..Y-1 \qquad \text{(A.3.4.1-7)}$$

$$Tx=X/2^{stage} \qquad \text{(A.3.4.1-8)}$$

$$Ty=Y/2^{(stage+1)} \qquad \text{(A.3.4.1-9)}$$

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

$$P(x,y)=0.5\times(PTemp(x,2y)+PTemp(x,2y+1)) \quad x=0..Tx-1, \quad y=0..Ty-1 \qquad \text{(A.3.4.1-10)}$$

$$P(x,y+Ty)=PTemp(x,2y)-PTemp(x,2y+1) \quad x=0..Tx-1 \quad y=0..Ty-1 \qquad \text{(A.3.4.1-11)}$$

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 A.4:

| | |
|---|---|
| Q(0,0) | Q(0,1) |
| Q(0,2) | Q(0,3) |

Q(0,0) = average of blocks of 4
Q(0,1) = horizontal dif. of blocks of 4
Q(0,2) = vertical dif. of blocks of 4
Q(0,3) = diagonal dif. of blocks of 4

**Figure A.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 A.5:

<table>
<tr><td>Q(1,0)</td><td>Q(1,1)</td><td rowspan="2">Q(0,1)</td></tr>
<tr><td>Q(1,2)</td><td>Q(1,3)</td></tr>
<tr><td colspan="2">Q(0,2)</td><td>Q(0,3)</td></tr>
</table>

**Figure A.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 × 2 pixel analysis, three Q(1,1..3) from the stage 1 4 × 4 analysis and four Q(2,0..3) from the stage 2 8 × 8 analysis.

The 3-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.

### A.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:

$$E(s,q) = (1/(XY)^2) \sum_{x=x1(s,q)}^{x2(s,q)-1} \sum_{y=y1(s,q)}^{y2(s,q)-1} (\Pr ef(x,y) - P\deg(x,y))^2 \quad s=0..2 \quad q=1..3 \qquad \text{(A.3.4.2-1)}$$

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

$$x1(s,1) = X/2^{(s+1)} \quad x2(s,1) = 2 \times x1(s,1) \quad y1(s,1) = 0 \quad y2(s,1) = Y/2^{(s+1)} \qquad \text{(A.3.4.2-2)}$$

$$x1(s,2) = 0 \quad x2(s,2) = X/2^{(s+1)} \quad y1(s,2) = Y/2^{(s+1)} \quad y2(s,2) = 2 \times y1(s,2) \qquad \text{(A.3.4.2-3)}$$

$$x1(s,3) = X/2^{(s+1)} \quad x2(s,3) = 2 \times x1(s,3) \quad y1(s,3) = Y/2^{(s+1)} \quad y2(s,3) = 2 \times y1(s,3) \qquad \text{(A.3.4.2-4)}$$

The results from (A.3.4.2-1) are then used to determine a PSNR measure for each quadrant of each field according to:

$$\begin{array}{l} if\,(E > 0.0) \quad PySNR(s,q) = 10.0 \times \log_{10}(255^2 / E(s,q)) \\ else \quad SNR = 10.0 \times \log_{10}(255^2 \times (XY)^2) \end{array} \qquad \text{(A.3.4.2-5)}$$

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

### A.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 A.3.5-1 to A.3.5-6.

For each field, first a turning-point counter is initialized according to equation A.3.5-1.

$$sum = 0 \qquad \text{(A.3.5-1)}$$

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

$$last\_pos = 0, \qquad last\_neg = 0 \tag{A.3.5-2}$$

$$dif(x) = P(x,y) - P(x+1,y) \tag{A.3.5-3}$$

$$if\,((dif(x) < 0)AND(last\_neg < last\_pos)) \text{ then } sum = sum + 1 \tag{A.3.5-4}$$

$$if\,((dif(x) > 0)AND(last\_neg > last\_pos)) \text{ then } sum = sum + 1 \tag{A.3.5-5}$$

$$if\,(dif(x) > 0) \text{ then } last\_pos = x \tag{A.3.5-6}$$

$$if\,(dif(x) < 0) \text{ then } last\_neg = x \tag{A.3.5-7}$$

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:

$$TextureDeg = sum \times 100 / XY \tag{A.3.5-8}$$

### A.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 (see Figure A.6).

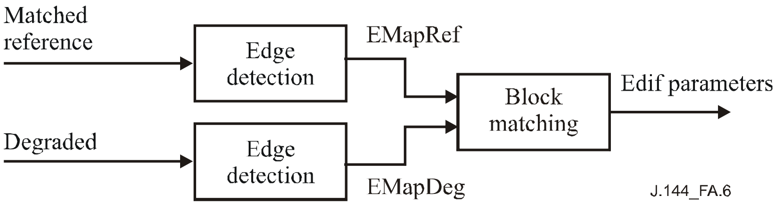


**Figure A.6 – Edge analysis**

#### A.3.6.1 Edge detection

A Canny edge detector [A-2] 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*):

$$EMap(x,y) = 1 \qquad x = 0..X-1, \qquad y = 0..Y-1 \tag{A.3.6.1-1}$$

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

$$EMap(x,y) = 0 \qquad x = 0..X-1, \qquad y = 0..Y-1 \tag{A.3.6.1-2}$$

#### A.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 A.3.6.2-1 to A.3.6.2-5.

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.

$$bref(x,y) = \sum_{i=i1}^{i2} \sum_{j=j1}^{j2} EMap\,\mathrm{Re}\,f(Nx + X1 + i, My + Y1 + j) \quad x = 0..Bh-1, y = 0..Bv-1 \qquad \text{(A.3.6.2-1)}$$

$$BDeg(x,y) = \sum_{i=i1}^{i2} \sum_{j=j1}^{j2} EMapDeg(Nx + X1 + i, My + Y1 + j) \quad x = 0..Bh-1, y = 0..Bv-1 \qquad \text{(A.3.6.2-2)}$$

The summation limits are determined according to:

$$i1 = -(N \;\; div \;\; 2) \qquad i2 = (N-1) \;\; div \;\; 2 \qquad \text{(A.3.6.2-3)}$$

$$j1 = -(M \;\; div \;\; 2) \qquad j2 = (M-1) \;\; div \;\; 2 \qquad \text{(A.3.6.2-4)}$$

where the "div" operator represents an integer division.

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

$$EDif = (1/(N \times M \times Bh \times Bv)) \times (\sum_{x=0}^{Bh-1} \sum_{y=0}^{Bv-1} (B\,\mathrm{Re}\,f(x,y) - BDeg(x,y))^{Q})^{1/Q} \qquad \text{(A.3.6.2-5)}$$

For 720 × 288 pixel fields for 625-broadcast video:

$$N = 4, \;\; X1 = 6, \;\; Bh = 178, \qquad M = 4, \;\; Y1 = 10, \;\; Bv = 69 \;\; Q = 3 \qquad \text{(A.3.6.2-6)}$$

For 720 × 243 pixel fields for 525-broadcast video:

$$N = 4, \;\; X1 = 6, \;\; Bh = 178, \qquad M = 4, \;\; Y1 = 10, \;\; Bv = 58, \;\; Q = 3 \qquad \text{(A.3.6.2-7)}$$

### A.3.7 Matched PSNR analysis

A matched signal-to-noise ratio is calculated for the pixel *V* values by use of the matching vectors defined in equations A.3.3.3-1 to A.3.3.3-3. For each set of matching vectors, an error measure, *VE*, is calculated according to:

$$\begin{aligned} VE(h,v) = (1/81) \sum_{i=-4}^{4} \sum_{j=-4}^{4} & (DegV(N, Px(h)+i, Py(h)+j) - \mathrm{Re}\,fVField(N + SyncT(h,v), \\ & Px(h) + SyncX(h,v) + i, Py(v) + SyncY(h,v) + j))^2 \end{aligned} \qquad \text{(A.3.7-1)}$$

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

$$SegVPSNR = (1/Qx \times Qy) \sum_{h=0}^{Qx-1} \sum_{v=0}^{Qy-1} 10.0 \times \log_{10}(255^2/(VE(h,v)+1)) \qquad \text{(A.3.7-2)}$$

## A.4 Integration

The integration procedure firstly requires the time averaging of the field-by-field detection parameters according to equation A.4-1:

$$AvD(k) = (1/N) \times \sum_{n=0}^{N-1} D(k,n) \qquad k = 0..5 \qquad \text{(A.4-1)}$$

where *N* is the total number of fields in the tested sequences and *D*(*k*, *n*) is 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:

$$PDMOS = Offset + \sum_{k=0}^{5} AvD(k) \times W(k) \qquad \text{(A.4-2)}$$

Tables A.2 and A.3 show the integrator parameters for 625 and 525 sequences respectively.

**Table A.2 – Integration parameters for 625-broadcast video**

| K | Parameter name | W |
|---|---|---|
| 0 | TextureDeg | –0.68 |
| 1 | PySNR(3,3) | –0.57 |
| 2 | EDif | +58913.294 |
| 3 | fXPerCent | –0.208 |
| 4 | MPSNR | –0.928 |
| 5 | SegVPSNR | –1.529 |
| Offset | +176.486 | |
| N | 400 | |

**Table A.3 – Integration parameters for 525-broadcast video**

| K | Parameter name | W |
|---|---|---|
| 0 | TextureDeg | +0.043 |
| 1 | PySNR(3,3) | –2.118 |
| 2 | EDif | +60865.164 |
| 3 | fXPerCent | –0.361 |
| 4 | MPSNR | +1.104 |
| 5 | SegVPSNR | –1.264 |
| Offset | +260.773 | |
| N | 480 | |

## A.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.

## A.6 References

[A-1] ITU-T Tutorial (2004), *Objective perceptual assessment of video quality: Full reference television*.

[A-2] J. CANNY: A computational approach to edge detection, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6): pp. 679-698, 1986.

## A.7 Subjective and objective data

NOTE – The following video files were processed to produce the results shown. The filenames are given for information only. The files are not available publicly. For more information see ITU-T Tutorial.

*525 subjective and objective data*

| Filename | SRC | HRC | Raw mean subjective rating | Raw model predicted rating | Scaled mean subjective rating | Scaled model predicted rating |
|---|---|---|---|---|---|---|
| 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 |
| 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 |

| Filename | SRC | HRC | Raw mean subjective rating | Raw model predicted rating | Scaled mean subjective rating | Scaled model predicted rating |
|---|---|---|---|---|---|---|
| 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 |

*625 subjective and objective data*

| Filename | SRC | HRC | Raw mean subjective rating | Raw model predicted rating | Scaled mean subjective rating | Raw model predicted rating |
|---|---|---|---|---|---|---|
| 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 |
| 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 |

| Filename | SRC | HRC | Raw mean subjective rating | Raw model predicted rating | Scaled mean subjective rating | Raw model predicted rating |
|---|---|---|---|---|---|---|
| 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 B

# Yonsei University/SK Telecom/Radio Research Laboratory

# Full reference video quality model functional description

## B.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 which is easy to implement was developed, fast enough for real-time implementations and robust over a wide range of video impairments. The model is a product of collaborated works from Yonsei University, SK Telecom, and Radio Research Laboratory.

## B.2 Objective measurement of video quality based on edge degradation

### B.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 analyzing 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 edge PSNR 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:

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

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 B.1- B.5 illustrate the above procedure. Figure B.1 shows a source image. Figure B.2 shows a horizontal gradient image $g_{horizontal}(m,n)$, which is obtained by applying a horizontal gradient operator to the source image of Figure B.1. Figure B.3 shows a vertical gradient image $g_{vertical}(m,n)$, which is obtained by applying a vertical gradient operator to the source image of Figure B.1. Figure B.4 shows the magnitude gradient image (edge image) and Figure B.5 shows the binary edge image (mask image) obtained by applying thresholding to the magnitude gradient image of Figure B.4.

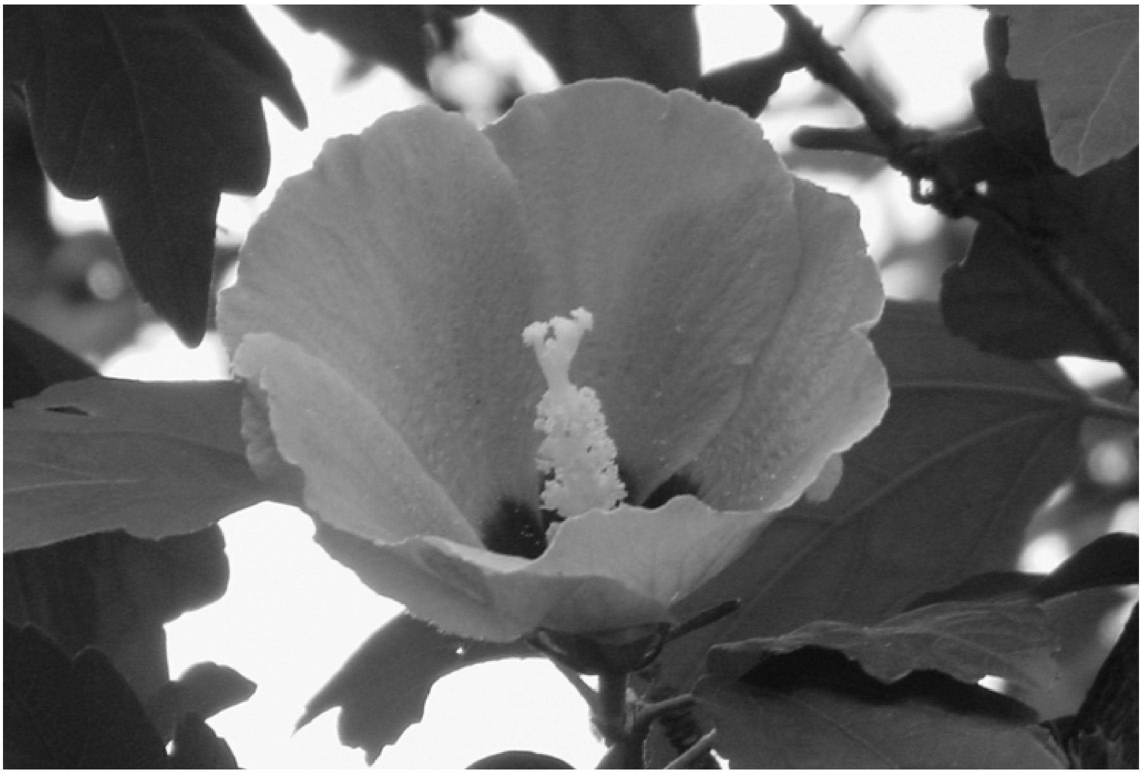


**Figure B.1 – A source image (original image)**

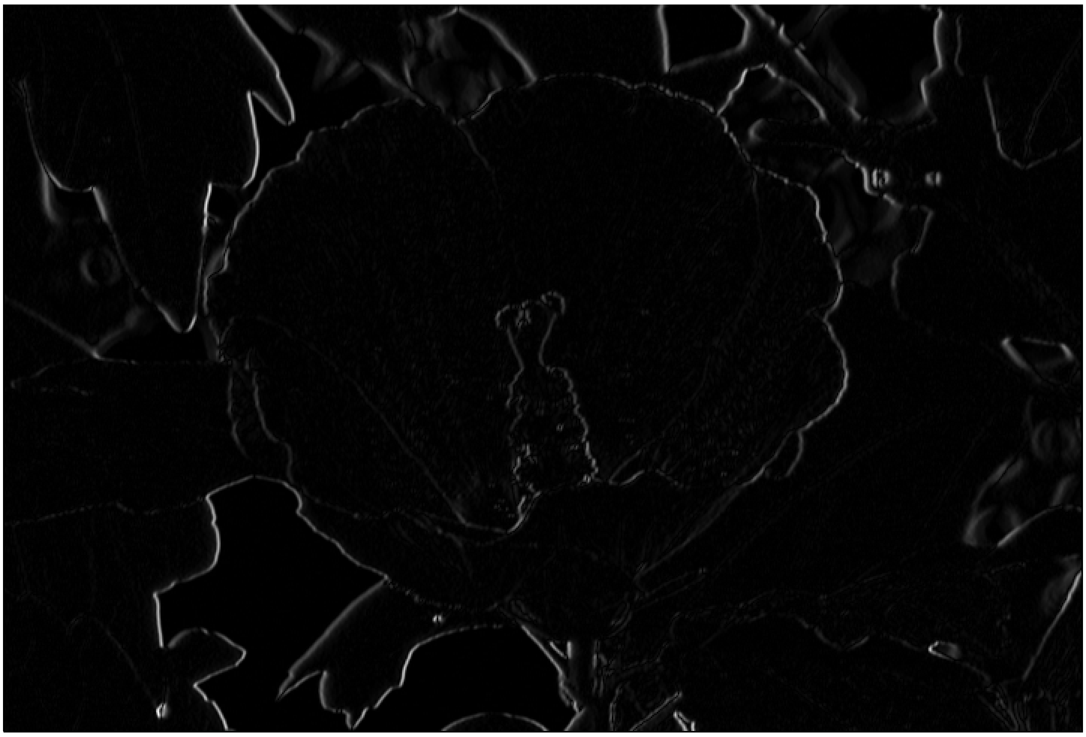


**Figure B.2 – A horizontal gradient image, which is obtained by applying a horizontal gradient operator to the source image of Figure B.1**

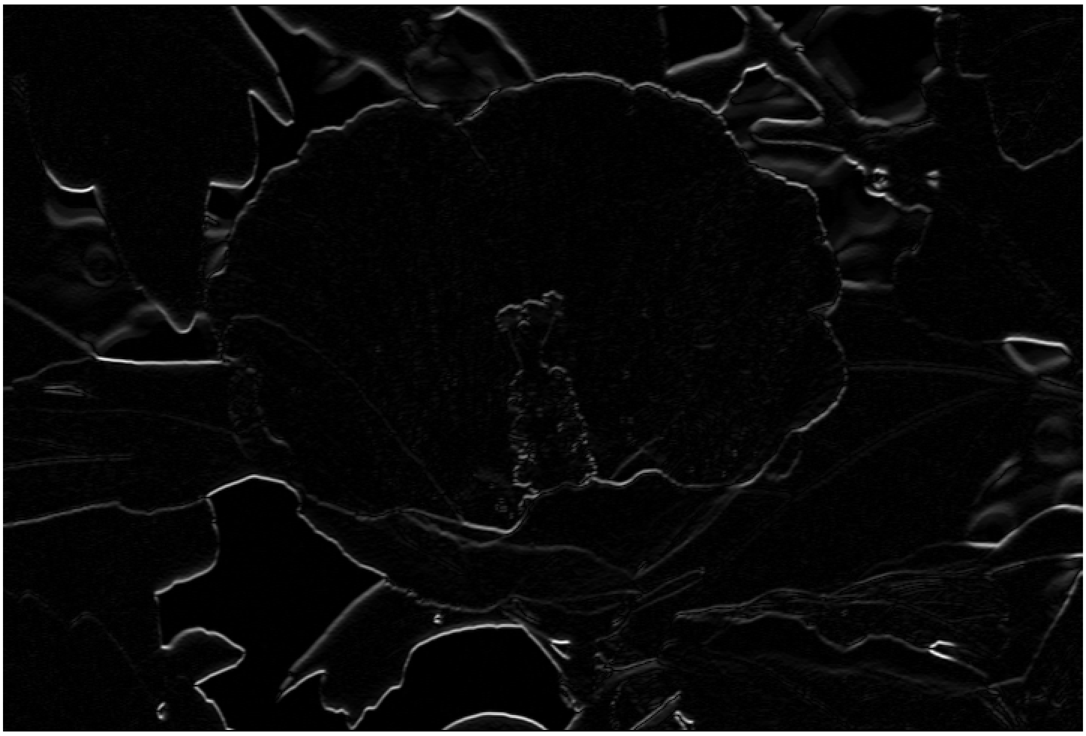


**Figure B.3 – A vertical gradient image, which is obtained by applying a vertical gradient operator to the source image of Figure B.1**

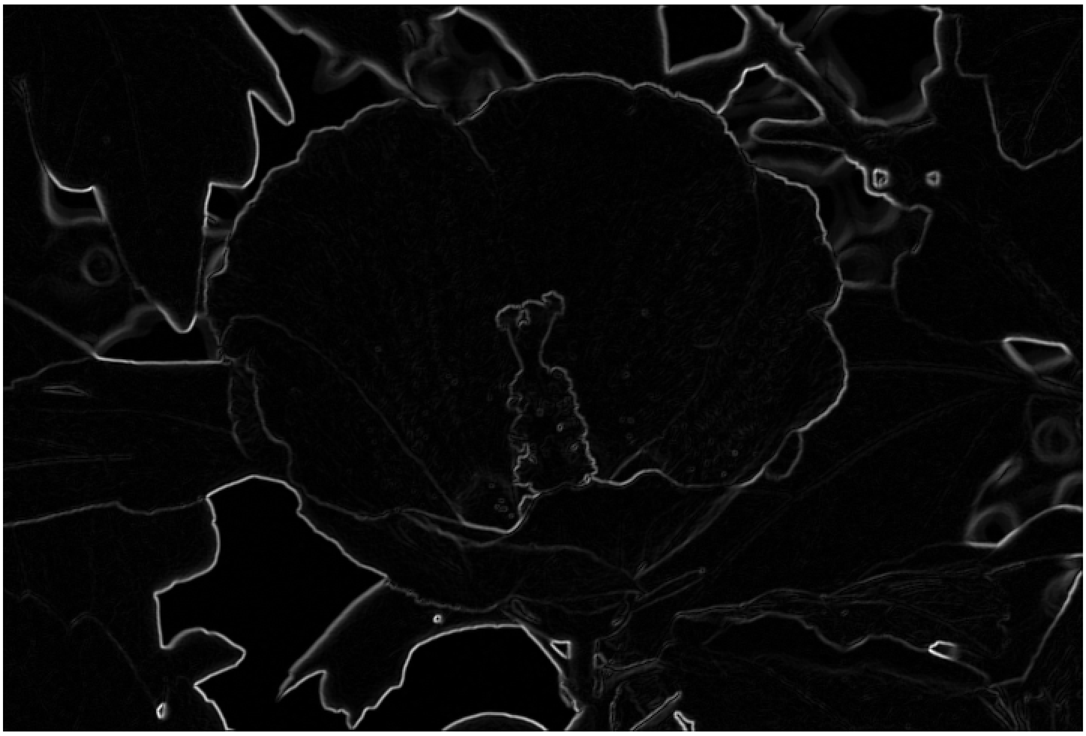


**Figure B.4 – A magnitude gradient image**

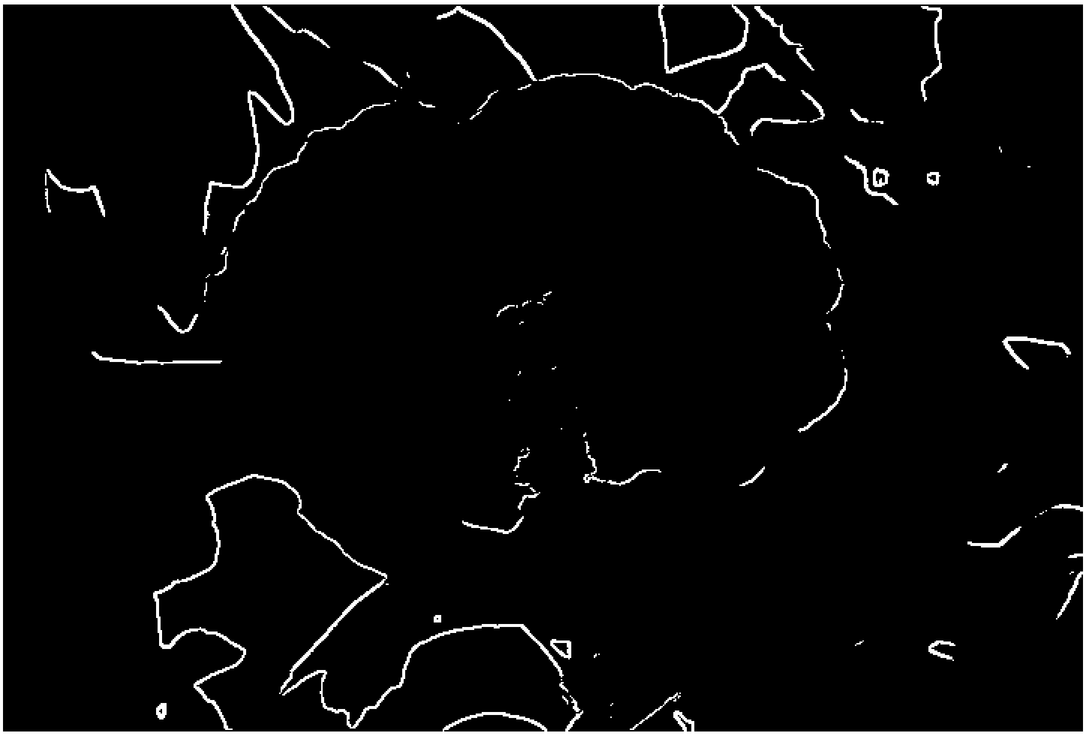


**Figure B.5 – A binary edge image (mask image) obtained by applying thresholding to the magnitude gradient image of Figure B.4**

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 B.6-B.9 illustrate the modified procedure. Figure B.6 shows a vertical gradient image $g_{vertical}(m,n)$, which is obtained by applying a vertical gradient operator to the source image of Figure B.1. Figure B.7 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 Figure B.6. Figure B.8 shows the binary edge image (mask image) obtained by applying thresholding to the modified successive gradient image of Figure B.7.

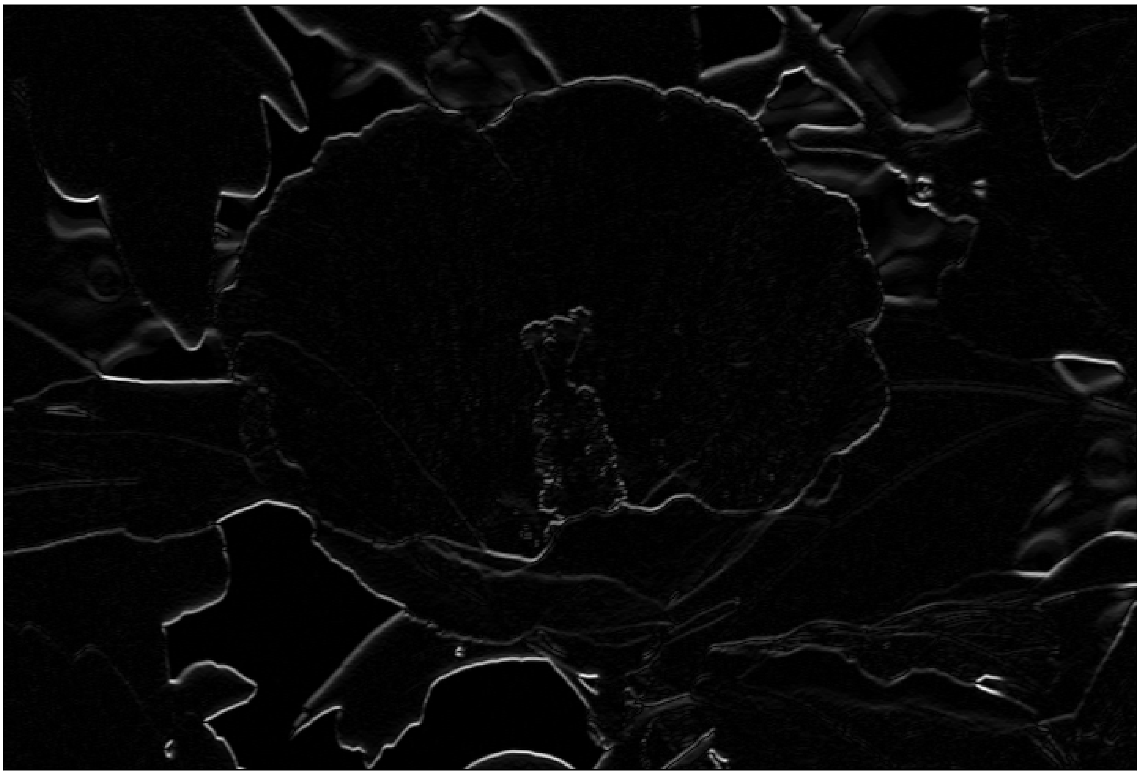


**Figure B.6 – A vertical gradient image, which is obtained by applying a vertical gradient operator to the source image of Figure B.1**

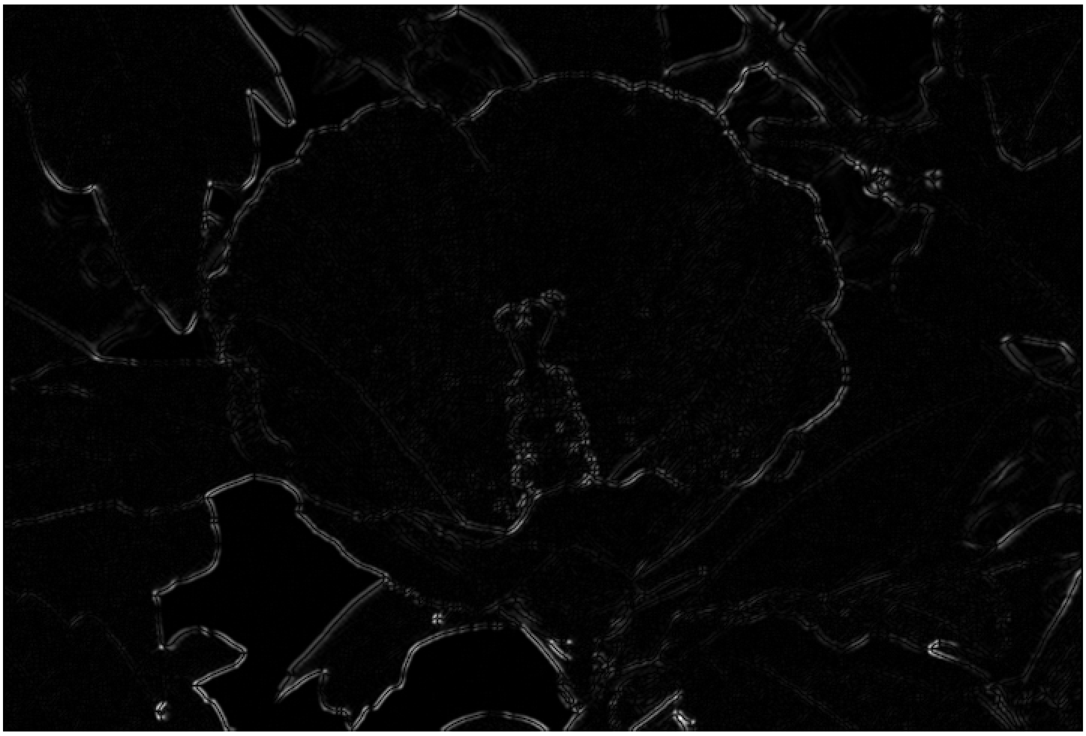


**Figure B.7 – A modified successive gradient image (horizontal and vertical gradient image), which is obtained by applying a horizontal gradient operator to the vertical gradient image of Figure B.6**

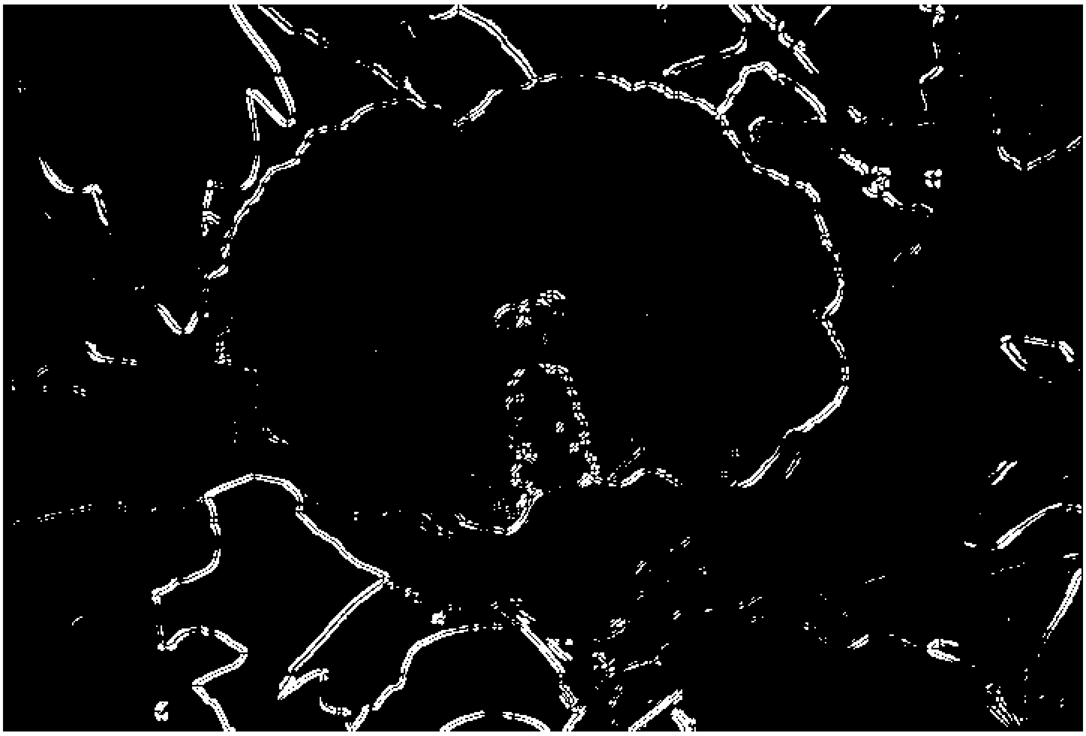


**Figure B.8 – A binary edge image (mask image) obtained by applying thresholding to the modified successive gradient image of Figure B.7**

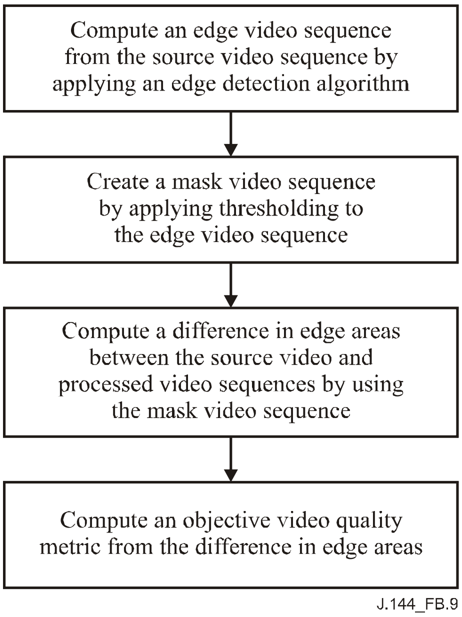

Compute an edge video sequence from the source video sequence by applying an edge detection algorithm
Create a mask video sequence by applying thresholding to the edge video sequence
Compute a difference in edge areas between the source video and processed video sequences by using the mask video sequence
Compute an objective video quality metric from the difference in edge areas
J.144_FB.9

**Figure B.9 – A block diagram of EPSNR**

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 (Figures B.4 and B.7). Then, a mask image (binary edge image) is produced by applying thresholding to the edge image (Figures B.5 and B.8). 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 B.5 and B.8 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:

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

where $S^l(i,j)$ is the *l*-th image of the source video sequence, $P^l(i,j)$ is the *l*-th image of the processed video sequence, $R^l(i,j)$ is the *l*-th image of the mask video sequence, *M* is the number of rows, and *N* is the 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:

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

where $Q^l(i,j)$ is the $l$-th image of the edge video sequence and $t_e$ is a threshold. Although a sum of squared error is used in equation B-1 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:

$$mse_e = \frac{1}{K}\sum_{l=1}^{L} se_e^l \qquad \text{(B-3)}$$

where $L$ is the number of images (frames or fields) and $K$ is the total number of pixels of the edge areas. Finally, the PSNR of the edge areas is computed as follows:

$$EPSNR = 10\log_{10}(\frac{P^2}{mse_e}) \qquad \text{(B-4)}$$

where $P$ is the peak pixel value. In the model, this edge PSNR (EPSNR) is used as a basic objective video quality score. In the model, $P$=255 is used. Figure B.9 shows a block diagram of computing the EPSNR.

### B.2.2 Post adjustments

#### B.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:

$$EPSNR = \begin{cases} EPSNR & if\ \ 0 \le EPSNR \le 35 \\ EPSNR \times 0.9 & if\ \ 35 \le EPSNR \le 40 \\ EPSNR \times 0.8 & if\ \ EPSNR > 40 \end{cases} \qquad \text{(B-5)}$$

#### B.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:

$$MEPSNR = \begin{cases} EPSNR - 60 \times (0.1225 - (\frac{EP_{common}}{EP_{src}})^2) & if\ \ EPSNR < 25\ \ and\ \ \frac{EP_{common}}{EP_{src}} < 0.35\ \ and\ \frac{EP_{hrc}}{EP_{src}} < 0.13 \\ EPSNR & otherwise \end{cases} \qquad \text{(B-6)}$$

where:

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

$EP_{hrc}$: the total number of edge pixels in the processed video sequence (HRC)

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

$MEPSNR$: modified EPSNR.

For some video sequences, $EP_{src}$ can be very small. In the worst scenario, $EP_{src}$ can be zero (blank images or very low frequency images), causing a zero-division error. In order to prevent such cases, the following modification is recommended: If $EP_{src}$ is smaller than 10 000 pixels (about 10 000/240 = 41.7 pixels per frame for the 8-seconds 525 videos and about 10 000/200 = 50 pixels per frame for the 8-seconds 625 videos), the user may reduce threshold $t_e$ in equation B-2 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 B-6 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 B-6.

#### B.2.2.3 Scaling

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

$$VQM = 1 - MEPSNR \times 0.02 \qquad \text{(B-7)}$$

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

### B.2.3 Recommended registration accuracy for the model

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 or better is strongly recommended to calculate sub-pixel values.

### B.2.4 The block diagram of the model

Figure B.10 shows the complete block diagram of the model. On the other hand, Figure B.11 shows a modified block-diagram, which prevents the zero-division error.

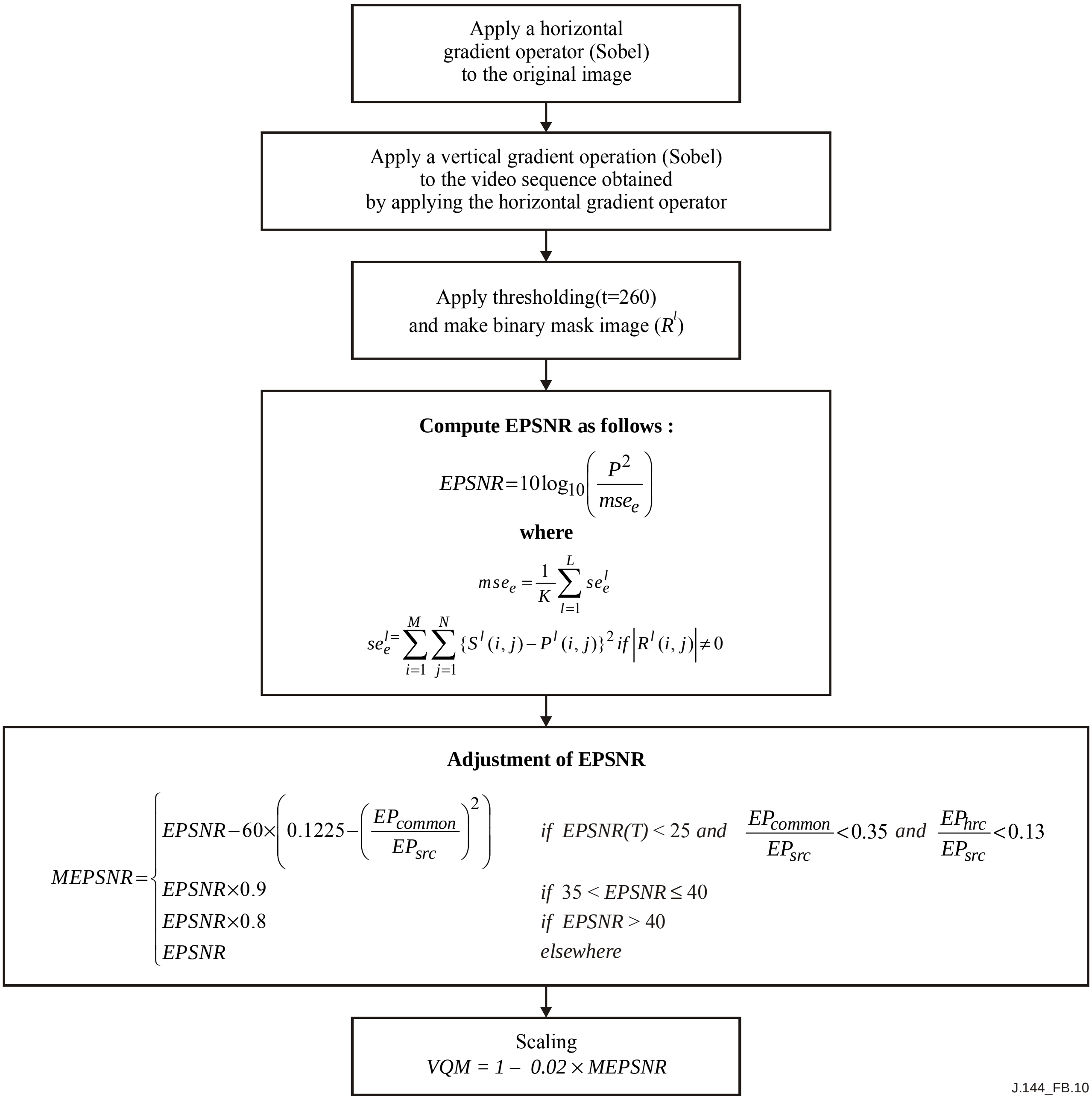

Apply a horizontal gradient operator (Sobel) to the original image
Apply a vertical gradient operation (Sobel) to the video sequence obtained by applying the horizontal gradient operator
Apply thresholding(t=260) and make binary mask image ($R^l$)
**Compute EPSNR as follows :**
$EPSNR = 10\log_{10}\left(\frac{P^2}{mse_e}\right)$
**where**
$mse_e = \frac{1}{K}\sum_{l=1}^{L} se_e^l$
$se_e^l = \sum_{i=1}^{M}\sum_{j=1}^{N}\{S^l(i,j) - P^l(i,j)\}^2 \; if \left|R^l(i,j)\right| \neq 0$
**Adjustment of EPSNR**
$MEPSNR = \begin{cases} EPSNR - 60\times\left(0.1225 - \left(\frac{EP_{common}}{EP_{src}}\right)^2\right) & if\ EPSNR(T) < 25\ and\ \frac{EP_{common}}{EP_{src}} < 0.35\ and\ \frac{EP_{hrc}}{EP_{src}} < 0.13 \\ EPSNR \times 0.9 & if\ 35 < EPSNR \leq 40 \\ EPSNR \times 0.8 & if\ EPSNR > 40 \\ EPSNR & elsewhere \end{cases}$
Scaling
$VQM = 1 - 0.02 \times MEPSNR$
J.144_FB.10

**Figure B.10 – The complete block-diagram of the model (*P*=255 is used in the model)**

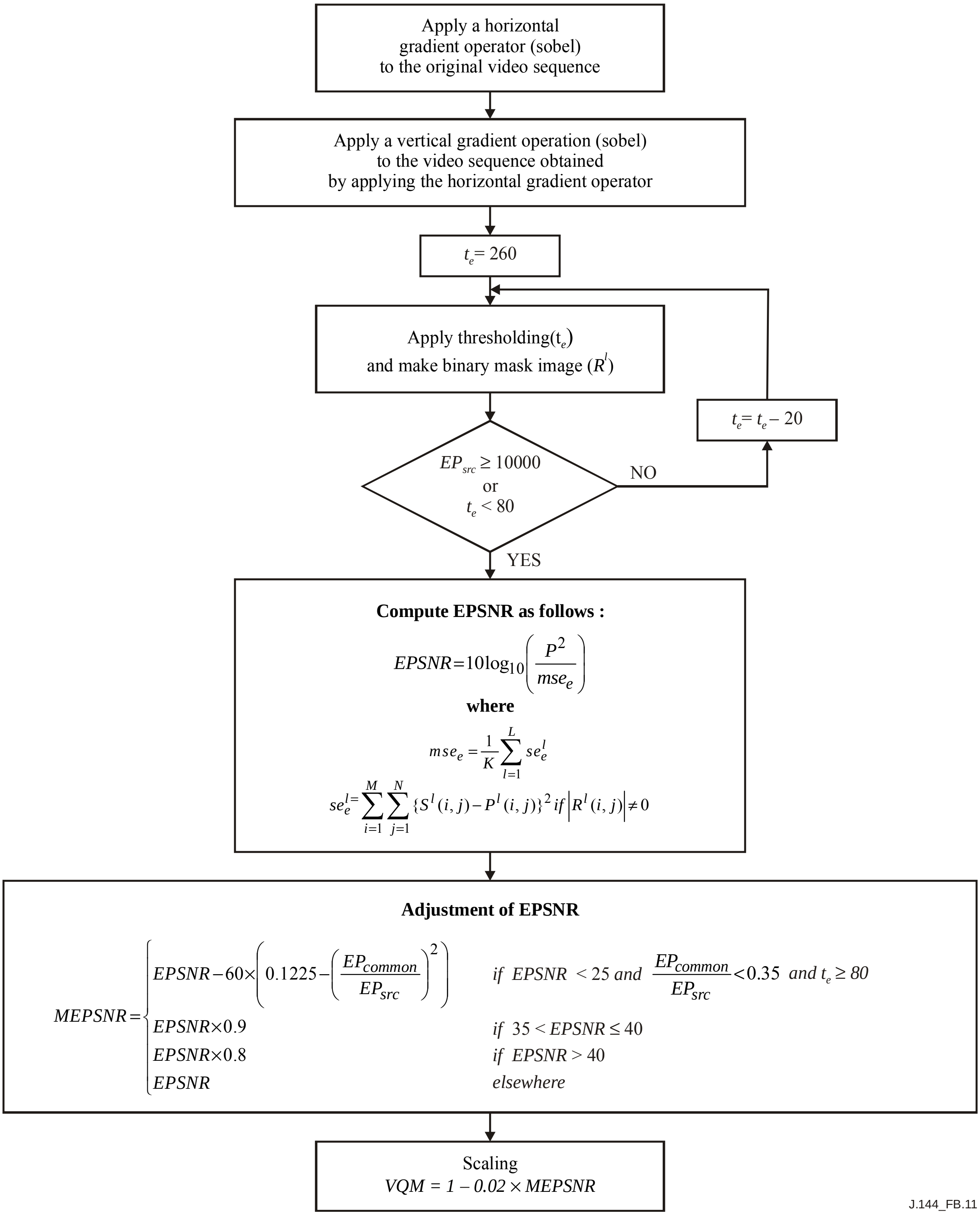

Apply a horizontal gradient operator (sobel) to the original video sequence
Apply a vertical gradient operation (sobel) to the video sequence obtained by applying the horizontal gradient operator
$t_e = 260$
Apply thresholding($t_e$) and make binary mask image ($R^l$)
$t_e = t_e - 20$
$EP_{src} \geq 10000$ or $t_e < 80$
NO
YES
Compute EPSNR as follows :
$EPSNR = 10\log_{10}\left(\frac{P^2}{mse_e}\right)$
where
$mse_e = \frac{1}{K}\sum_{l=1}^{L} se_e^l$
$se_e^l = \sum_{i=1}^{M}\sum_{j=1}^{N}\{S^l(i,j) - P^l(i,j)\}^2 \; if \left|R^l(i,j)\right| \neq 0$
Adjustment of EPSNR
$MEPSNR = \begin{cases} EPSNR - 60\times\left(0.1225 - \left(\frac{EP_{common}}{EP_{src}}\right)^2\right) & if\ EPSNR < 25\ and\ \frac{EP_{common}}{EP_{src}} < 0.35\ and\ t_e \geq 80 \\ EPSNR\times 0.9 & if\ 35 < EPSNR \leq 40 \\ EPSNR\times 0.8 & if\ EPSNR > 40 \\ EPSNR & elsewhere \end{cases}$
Scaling
$VQM = 1 - 0.02 \times MEPSNR$
J.144_FB.11

**Figure B.11 – A modified block-diagram, which prevents the zero-division error (*P*=255 is used in the model)**

## B.3 Registration

### B.3.1 Video registration

Video registration is necessary to find the best match between two video sequences. In video quality assessment, one needs to find how much the processed video is shifted in the spatial or temporal directions. If the translation is represented as displacement vector $D = [d_1, d_2, d_3]^{\mathrm{T}}$, the mean square error between a source video sequence and a processed video sequence that is translated by $D$ is computed by

$$MSE(d_1, d_2, d_3) = \frac{1}{LMN} \sum_{l}^{L} \sum_{m}^{M} \sum_{n}^{N} (U(m,n,l) - V(m + d_1, n + d_2, l + d_3))^2 \qquad \text{(B-8)}$$

where $U$ and $V$ represent video sequences. The best estimate of the displacement vector that provides the best matching is obtained by minimizing the $MSE$ :

$$\hat{D} = \arg \min_{(d_1, d_2, d_3)} MSE(d_1, d_2, d_3) \qquad \text{(B-9)}$$

The precision of the vertical and horizontal component of displacement vector may be integer or fractional pixel. If it is fractional, one needs to use an interpolation technique such as the bilinear interpolation or the cubic spline interpolation. Generally, the precision of the temporal component of the displacement vector is one video frame for a progressively scanned video sequence as shown in Figure B.12. For an interlaced video sequence, it is one field shift: 1/50 sec for 50 Hz interlaced videos and 1/60 sec for 60 Hz interlaced videos as shown in Figure B.13. In the interlaced format, one needs to construct a complete frame from each field to find the spatial displacement of interlaced videos.

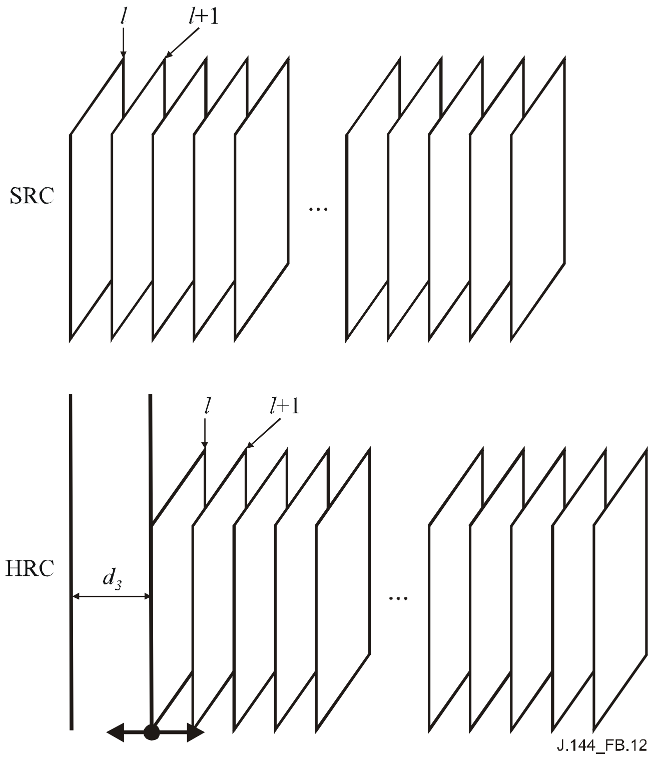


**Figure B.12 – Temporal registration of the progressive video sequence**
**The unit of $d_3$ is one video frame**

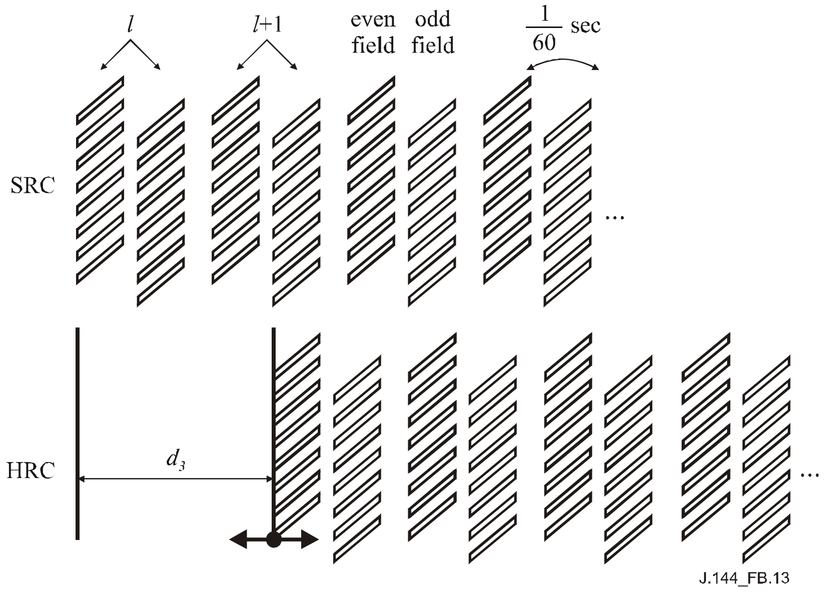


**Figure B.13 – Temporal registration of interlaced video sequences**
**The unit of $d_3$ is one field (1/60 sec for 60 Hz interlaced video sequence)**

### B.3.2 Video registration based on region of interest

Typically, it requires a very long processing time to find the best match using equations B-8 and B-9. If accurate registration is required, one may use all the frames of a video sequence to find *MSE*. However, it would take a long processing time and it might not be done in real time. In order to reduce the processing time in the model, a smaller number of subregions (regions of interests) are selected from the video sequence. Then, the model finds the best match between two video sequences by computing *MSE* in the regions of interest (ROI):

$$MSE_{\mathrm{ROI}}(d_1,d_2,d_3)=\frac{1}{K}\sum_{(m,n,l)\in ROI}(U(m,n,l)-V(m+d_1,n+d_2,l+d_3))^2 \qquad \text{(B-10)}$$

where *K* represents the number of pixels in ROI.

It is observed that rapidly changing areas of video sequences provide valuable information for use in image registration. Therefore, the model locates such areas and uses them for video registration. In order to find rapidly changing scenes, the frame mean squared error (*fMSE*) of the *l*-th frame of a source video sequence is computed as follows:

$$fMSE(l)=\frac{1}{MN}\sum_{m}^{M}\sum_{n}^{N}(U(m,n,l)-U(m,n,l+1))^2 \qquad \text{(B-11)}$$

After computing *fMSE*(*l*) for the entire frames of the source video sequence, the model selects five reference frames that have maximum *fMSE*s. These frames can be considered as the most rapidly changing scenes in the temporal direction. Furthermore, the model has an additional restriction that the intervals between the reference frames should exceed a certain amount of time period. For the purpose, the model uniformly divides the entire video sequence into five sub-sequences. From each sub-sequence, the model selects a frame which has the largest *fMSE* of equation B-11.

Among the five selected frames, the model chooses the frame which has the largest frame MSE. Then, the model applies a 2-D wavelet transform to the frame and areas where high frequency coefficients are dominant are considered as rapidly changing areas in the spatial domain. The 2-D wavelet transform is performed by applying separate 1-D wavelet transforms in the horizontal and vertical directions. The 1-D wavelet transform is computed by

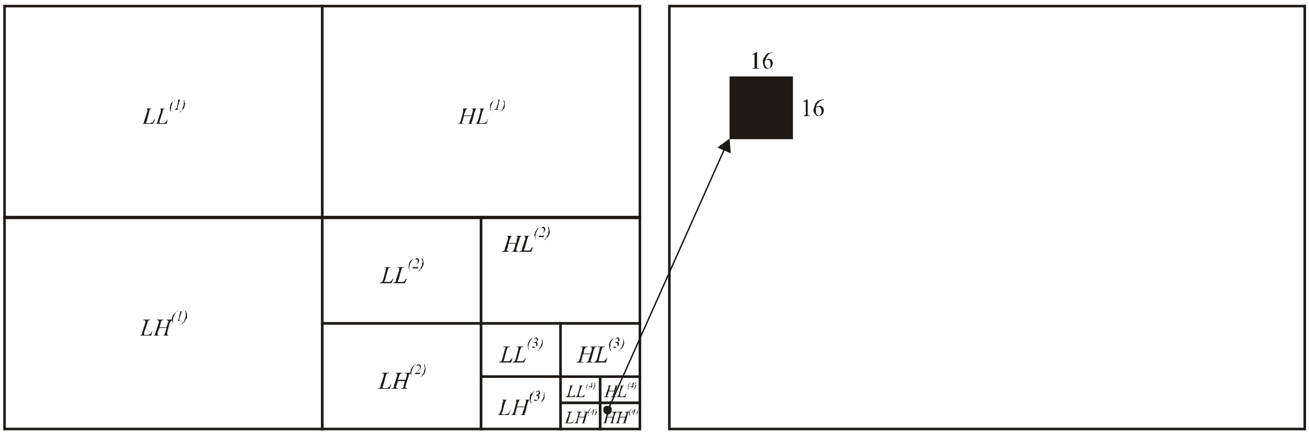
$$y_L^{(1)}[n] = \sum_k x[k]h_0[2n-k] \qquad \text{(B-12)}$$

$$y_H^{(1)}[n] = \sum_k x[k]h_1[2n-k] \qquad \text{(B-13)}$$

where $x[n]$ is an 1-D input signal and $h_0[n]$, $h_1[n]$ are the impulse responses of lowpass and highpass analysis filters (Harr filters). The terms $y_L^{(1)}[n]$, $y_H^{(1)}[n]$ represent output signals of the filters. The model repeatedly applies the wavelet transform to the high frequency subband as follows:

$$y_H^{(l+1)}[n] = \sum_k y_H^{(l)} h_1[2n-k] \qquad \text{(B-14)}$$

Figure B.14 shows the recursive transform of the high frequency subband for the 2-D image. Then, the model selects twelve maximum coefficients in the highest frequency subband and the corresponding subregions are selected as ROIs as illustrated in Figure B.14. In the model, the decomposition level is four. Thus, the size of ROIs in the original image is 16×16 at $(2^4 m, 2^4 n)$. In this way, the model selects twelve 16×16 ROIs which have high spatial frequencies.



J.144_FB.14

**Figure B.14 – Wavelet transformed frame and ROI for fast video registration**

In order to select ROIs which have high temporal frequencies, the frame which has the largest *fMSE* is divided into a number of 16×16 blocks (Figure B.15). Then, the model chooses twelve blocks which have the largest absolute block difference. The absolute block difference (*ABD*) is computed as follows:

$$ABD = \frac{1}{256} \sum_{(m,n) \in k-th\,block} \left| U(m,n,l) - U(m,n,l+1) \right| \qquad \text{(B-15)}$$

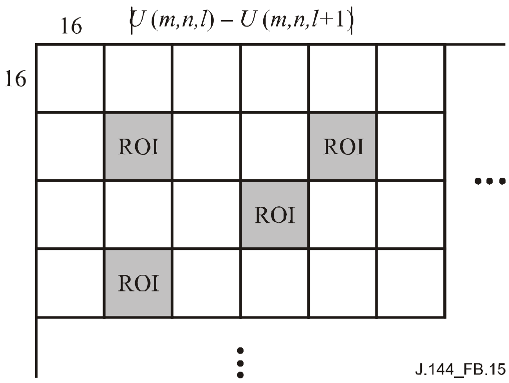


**Figure B.15 – The model chooses 12 blocks which have the largest ABDs from the frame which has the largest frame MSE**

Thus, the model selects twelve blocks which have high spatial frequencies and twelve blocks which have the largest *ABDs* from the frame which has the largest frame MSE. These 24 blocks are used as ROID. It is noted that the blocks (24×4=96 blocks) of the remaining four frames, which are located in the same locations are also used as ROID. Thus, the total of 120 blocks whose size is 16×16 are used as ROID and equation B-10 is used to find the best displacement vector. It is also noted that the 24 blocks of the frame which has the largest frame MSE have double weighting when compared with the blocks in the remaining four frames. During the registration process, a quarter pixel registration was performed using the cubic spline interpolation.

## B.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.

## B.5 References

[B-1] ITU-T Tutorial (2004), *Objective perceptual assessment of video quality: Full reference television*.

## Table B.1 – The 525 VQM matrix (raw data)[2]

| SRC (Image) | HRC=1 | | HRC=2 | | HRC=3 | | HRC=4 | | HRC=5 | | HRC=6 | | HRC=7 | | HRC=8 | | HRC=9 | | HRC=10 | | HRC=11 | | HRC=12 | | HRC=13 | | HRC=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 |

[2] After the model was submitted, registration and operator errors were found. The objected data presented in this annex are 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 B.1.

### Table B.2 – The 625 VQM matrix (raw data)[3]

| SRC (Image) | HRC=1 | | HRC=2 | | HRC=3 | | HRC=4 | | HRC=5 | | HRC=6 | | HRC=7 | | HRC=8 | | HRC=9 | | HRC=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 |

[3] After the model was submitted, registration and operator errors were found. The objected data presented in this annex are 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 B.2.

## Table B.3 – The 525 VQM matrix (scaled data)[4]

| SRC (Image) | HRC=1 | | HRC=2 | | HRC=3 | | HRC=4 | | HRC=5 | | HRC=6 | | HRC=7 | | HRC=8 | | HRC=9 | | HRC=10 | | HRC=11 | | HRC=12 | | HRC=13 | | HRC=14 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0.727 | 4 | 0.490 | 7 | 0.467 | 10 | 0.304 | | | | | | | | | | | | | | | | | | | | |
| 2 | 2 | 0.324 | 5 | 0.224 | 8 | 0.162 | 11 | 0.195 | | | | | | | | | | | | | | | | | | | | |
| 3 | 3 | 0.549 | 6 | 0.359 | 9 | 0.192 | 12 | 0.153 | | | | | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | 13 | 0.715 | 17 | 0.588 | 21 | 0.546 | 25 | 0.509 | | | | | | | | | | | | |
| 5 | | | | | | | | | 14 | 0.523 | 18 | 0.418 | 22 | 0.344 | 26 | 0.289 | | | | | | | | | | | | |
| 6 | | | | | | | | | 15 | 0.665 | 19 | 0.404 | 23 | 0.341 | 27 | 0.292 | | | | | | | | | | | | |
| 7 | | | | | | | | | 16 | 0.386 | 20 | 0.298 | 24 | 0.239 | 28 | 0.199 | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | 29 | 0.823 | 35 | 0.783 | 41 | 0.788 | 47 | 0.537 | 53 | 0.481 | 59 | 0.512 |
| 9 | | | | | | | | | | | | | | | | | 30 | 0.854 | 36 | 0.550 | 42 | 0.774 | 48 | 0.435 | 54 | 0.377 | 60 | 0.383 |
| 10 | | | | | | | | | | | | | | | | | 31 | 0.539 | 37 | 0.354 | 43 | 0.520 | 49 | 0.206 | 55 | 0.156 | 61 | 0.241 |
| 11 | | | | | | | | | | | | | | | | | 32 | 0.634 | 38 | 0.662 | 44 | 0.669 | 50 | 0.400 | 56 | 0.393 | 62 | 0.442 |
| 12 | | | | | | | | | | | | | | | | | 33 | 0.580 | 39 | 0.515 | 45 | 0.581 | 51 | 0.390 | 57 | 0.332 | 63 | 0.353 |
| 13 | | | | | | | | | | | | | | | | | 34 | 0.542 | 40 | 0.568 | 46 | 0.476 | 52 | 0.273 | 58 | 0.247 | 64 | 0.293 |

[4] After the model was submitted, registration and operator errors were found. The objected data presented in this annex are 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 B.3.

### Table B.4 – The 625 VQM matrix (scaled data)[5]

| SRC (Image) | HRC=1 | | HRC=2 | | HRC=3 | | HRC=4 | | HRC=5 | | HRC=6 | | HRC=7 | | HRC=8 | | HRC=9 | | HRC=10 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | 4 | 0.625 | 10 | 0.429 | 16 | 0.204 | | | 29 | 0.164 | | | 42 | 0.169 | | | 52 | 0.082 |
| 2 | | | 5 | 0.467 | 11 | 0.454 | 17 | 0.202 | | | 30 | 0.170 | | | 43 | 0.177 | | | 53 | 0.062 |
| 3 | | | 6 | 0.542 | 12 | 0.538 | 18 | 0.327 | | | 31 | 0.275 | | | 44 | 0.272 | | | 54 | 0.134 |
| 4 | | | 7 | 0.605 | 13 | 0.686 | 19 | 0.502 | | | 32 | 0.475 | | | 45 | 0.414 | | | 55 | 0.255 |
| 5 | | | 8 | 0.609 | 14 | 0.641 | 20 | 0.335 | | | 33 | 0.312 | | | 46 | 0.185 | | | 56 | 0.091 |
| 6 | | | 9 | 0.586 | 15 | 0.395 | 21 | 0.284 | | | 34 | 0.248 | | | 47 | 0.226 | | | 57 | 0.143 |
| 7 | | | | | | | 22 | 0.551 | | | 35 | 0.496 | | | | | 48 | 0.430 | 58 | 0.339 |
| 8 | | | | | | | 23 | 0.371 | | | 36 | 0.335 | | | | | 49 | 0.283 | 59 | 0.217 |
| 9 | | | | | | | 24 | 0.356 | | | 37 | 0.298 | | | | | 50 | 0.243 | 60 | 0.168 |
| 10 | | | | | | | 25 | 0.623 | | | 38 | 0.590 | | | | | 51 | 0.549 | 61 | 0.455 |
| 11 | 1 | 0.741 | | | | | | | 26 | 0.592 | | | 39 | 0.359 | | | | | 62 | 0.290 |
| 12 | 2 | 0.672 | | | | | | | 27 | 0.499 | | | 40 | 0.482 | | | | | 63 | 0.322 |
| 13 | 3 | 0.698 | | | | | | | 28 | 0.406 | | | 41 | 0.279 | | | | | 64 | 0.179 |

[5] After the model was submitted, registration and operator errors were found. The objected data presented in this annex are 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 B.4.

# Annex C

## Telecommunications Research and Development Center (CPqD)

## Image Evaluation based on Segmentation (IES) technical description

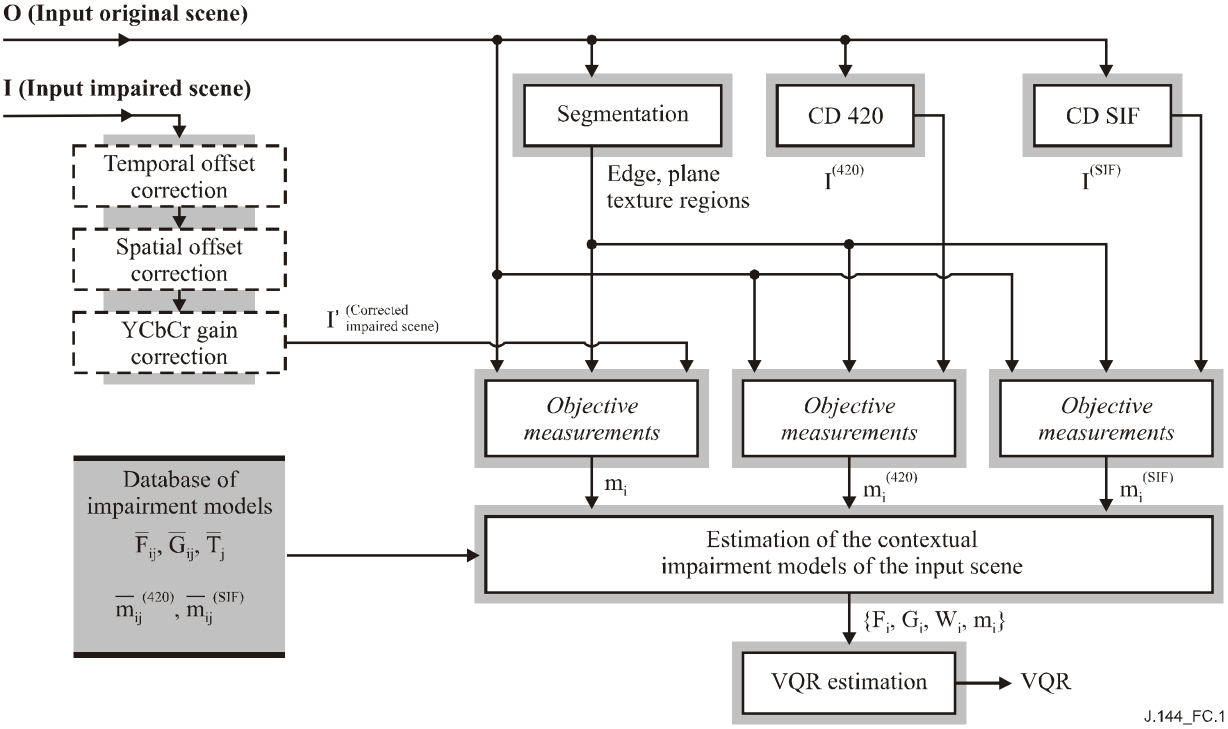
### C.1 Introduction

This annex provides a full description of the Image Evaluation based on Segmentation (IES) algorithm proposed by Telecommunications Research and Development Center (CPqD). The CPqD-IES algorithm is a methodology for video quality assessment using objective impairment measures, computed on plane, edge, and texture regions resulting from an image segmentation process. It provides predictions to human viewer judgement (Mean Opinion Score (MOS)), being an effective and efficient alternative to the costly and time-consuming subjective test methodologies in current use [C-1] and [C-2].

The CPqD-IES software prototype was submitted to the VQEG Phase II tests [C-3] and the raw objective scaled data obtained are shown in Tables C.2 and C.3.

### C.2 General description of the CPqD-IES algorithm

Figure C.1 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* (clause C.3).



O (Input original scene)
I (Input impaired scene)
Temporal offset correction
Spatial offset correction
YCbCr gain correction
I' (Corrected impaired scene)
Segmentation
Edge, plane texture regions
CD 420
$I^{(420)}$
CD SIF
$I^{(SIF)}$
Objective measurements
Objective measurements
Objective measurements
$m_i$
$m_i^{(420)}$
$m_i^{(SIF)}$
Database of impairment models
$\overline{F}_{ij}, \overline{G}_{ij}, \overline{T}_j$
$\overline{m}_{ij}^{(420)}, \overline{m}_{ij}^{(SIF)}$
Estimation of the contextual impairment models of the input scene
$\{F_i, G_i, W_i, m_i\}$
VQR estimation
VQR
J.144_FC.1

**Figure C.1 – General overview of the CPqD-IES algorithm**

Input scenes ***I*** and ***O*** to CPqD-IES algorithm are in YCbCr4:2:2 format according to ITU-R Rec. BT.601-5 [C-4].

The Y component of each frame $f$ of ***O*** is segmented into three categories: texture, edge, and plane regions (clause C.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 Y, Cb and Cr, forming a set of 9 objective measures $\{m_1, m_2, ..., m_9\}$ for each frame $f$ (clause C.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:

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

where $F_i$ and $G_i$ are two parameters computed (clause C.7) based on a database of impairment models (clause C.6), spatial $S$ and temporal $T$ attributes (clause C.5), and on the objective measures $m_i^{(420)}$ and $m_i^{(\mathrm{SIF})}$ for frame $f$, resulting from the codec operations CD420 and CDSIF applied to ***O*** (clause C.7). The two reference impairment codecs, CD420 (coder/decoder MPEG-2 4:2:0) and CDSIF (coder/decoder MPEG-1 SIF), are totally based on the routines extracted directly from MPEG-2 [C-5] and MPEG-1 [C-6], available at 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 CDSIF do not introduce offset and gain differences with respect to ***O***.

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:

$$VQR_f = \sum_{i=1}^{9} W_i . L_i \tag{C-2}$$

where $W_i$ is the weight of the impairment level $L_i$ for this particular natural scene, which is computed as described in clause C.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 among $VQR_{f-1}$, $VQR_f$, and $VQR_{f+1}$. 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 video quality rating ($VQR$) is then the average of the $VQR_f^{\prime}$ values.

$$VQR = \frac{1}{n-2} . \sum_{f=1}^{n-2} VQR'_f \tag{C-3}$$

Equations C-1 and C-2, 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 clauses complete the description of the method by presenting the details inside the remaining blocks of Figure C.1.

## C.3 Correction of offset and gain

### C.3.1 Temporal offset

The temporal offset $dt$ is an integer ranging from –2 to 2 frames. The positive values of $dt$ mean that the impaired scene $\boldsymbol{I}$ is delayed from original scene $\boldsymbol{O}$**,** while negative values of $dt$ tell the opposite: the original scene $\boldsymbol{O}$ is delayed from the impaired scene $\boldsymbol{I}$. Input scenes with temporal offsets out of this range are not considered by the model. Let $\boldsymbol{I}_{dt}$ be the impaired scene $\boldsymbol{I}$ with a displacement of $dt$ frames. A dissimilarity coefficient between original scene $\boldsymbol{O}$ and each displaced scene $\boldsymbol{I}_{dt}$ is calculated. The displacement with the lowest dissimilarity coefficient is used as temporal offset, and the output $\boldsymbol{I}_{dt}$ is then the impaired scene $\boldsymbol{I}$ 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:

```
$\xi_T \leftarrow 0$;
For each $f$ from 1 to $n$, do
    Compute $S_b$;
    Compute $S_b'$;
    Compute $D_b$;
    Compute $\mu$, the mean value of the pixels in $D_b$;
    $\xi_T \leftarrow \xi_T + (\mu/n)$
Return $\xi_T$ (dissimilarity coefficient between $\boldsymbol{O}$ and $\boldsymbol{I}_{dt}$).
```

where:

$S_b$ = the magnitude of the Sobel's gradient [C-7] of the component $Y$ of the $f$-th frame of $\boldsymbol{O}$

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

$D_b$ = the pixelwise absolute difference between $S_b$ and $S_b'$.

### C.3.2 Spatial offset

The spatial offset $(d_x, d_y)$ is one of the following integer horizontal and vertical displacements $d_x = -6, -5, \ldots 6$ pixels and $d_y = -6, -5, \ldots, 6$ pixels. Negative displacement values mean that a frame of the impaired scene $\boldsymbol{I}$ is displaced when referenced to a frame of the original scene $\boldsymbol{O}$, to the left and up directions. Positive displacement values mean that a frame of the impaired scene $\boldsymbol{I}$ is displaced when referenced to a frame of the original scene $\boldsymbol{O}$, to the right and down directions. The model does not consider input scenes with spatial offsets out of this range.

Consider $\boldsymbol{I}_{dx\,,dy}$ the impaired scene $\boldsymbol{I}_{dt}$ with all frames displaced of $(d_x, d_y)$ pixels. A dissimilarity coefficient between the original scene $\boldsymbol{O}$ and the displaced impaired scene $\boldsymbol{I}_{dx,\,dy}$ is calculated. The spatial displacement which leads to the lowest dissimilarity coefficient is used as spatial offset, and the output $\boldsymbol{I}_{dx,\,dy}$ is then the scene $\boldsymbol{I}_{dt}$ displaced by this offset, used for gain correction.

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

```
$\xi_S \leftarrow 0$ ; $c \leftarrow 0$;
For each $f$ from 1 to $n$, do
    For $x$ from $x0$ to $(x0 + w/4)$ do
        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 $\xi_S$ (dissimilarity coefficient between $\boldsymbol{O}$ and $\boldsymbol{I}_{dx,dy}$);
```

where:

$w \times h$ is an image portion with size of *w* columns and *h* lines presented both in a frame *f* of the original scene ***O*** and in a correspondent frame of the impaired scene $\boldsymbol{I}_{dx,dy}$

$Y(x, y)$, $Cb(x, y)$, $Cr(x, y)$ are the values in the image components of a frame *f* of ***O*** for a pixel $(x, y)$

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

### C.3.3 Gain

The amplitude gain between ***O*** and $\boldsymbol{I}_{dx,dy}$ is calculated for each image component *Y, Cb* and *Cr*, separately. The algorithm computes the average of the gains over all *n* frames and corrects each image component accordingly. The output ***I´*** is the impaired scene used for all afterward 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 ***O*** is obtained by blurring both images *C´* and *C*, using a Gaussian filter [C-7] with kernel

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

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

## C.4 Image segmentation

Initially, the segmentation algorithm classifies each pixel in the component *Y* of a given frame *f* of the original scene ***O*** into plane and non-plane regions. The algorithm also applies to *Y* an edge detector and the edge regions are defined by edges that fall within the boundary of the plane regions. The texture regions are composed by the remaining pixels of the image *Y* (see Figure C.2).

The segmentation is computed over the *Y* component of each frame of the input original scene ***O***. The segmented regions for the *Cb* and *Cr* components are derived subsampling *Y* component pixel position by a factor of 2 in the horizontal direction.

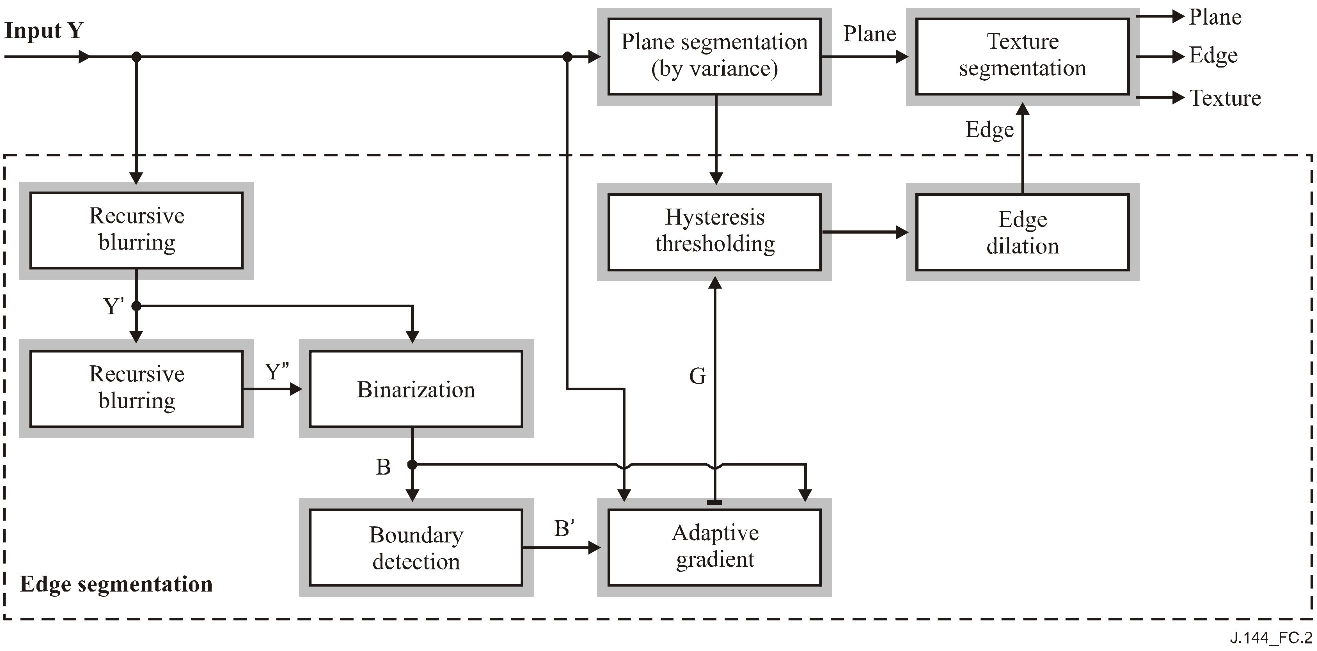


**Figure C.2 – Block diagram of the segmentation process**

### C.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 of variance is thresholded such that pixels with variance value below $25^2$ are classified as belonging to the plane regions. This process creates small pixel components misclassified within the texture regions. 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 [C-7]. This operation corresponds to the following equation applied to an input binary image *A* to create a dilated binary image *A'*.

$$A'(x,y) = \max\{A(x',y')\} \; \textit{for all pixels} \; (x',y') \in N_{81}(x,y) \qquad \text{(C-4)}$$

where $N_{81}(x,y)$ is the set of the 81 closest pixels of pixel *(x, y)*.

### C.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 *Y* of one frame of the input scene ***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)         $Y(x, y – 1) \leftarrow Y(x, y) + 0.7.[Y(x, y–1) – Y(x, y)]$;
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 $Y''$:

$$B(x,y)=\begin{cases}1, & \text{if } Y'(x,y)\geq Y''(x,y),\\ 0, & \text{otherwise}\end{cases} \tag{C-5}$$

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

$$B'(x,y)=\begin{cases}1, & \text{if } B(x,y)=1 \text{ and } B(x',y')=0 \ \text{ for any pixel } (x',y')\in \mathrm{N}_9(x,y)\\ 0, & \text{otherwise}\end{cases} \tag{C-6}$$

where $N_9(x, y)$ is the set of the 9 closest pixels of $(x, y)$.

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

$$G(x,y)=\begin{cases}\left|\mu_1-\mu_0\right|, & \text{if } B'(x,y)=1,\\ 0, & \text{otherwise,}\end{cases} \tag{C-7}$$

where:

$\mu_1$ = mean value of $Y(x',y')$, for all $(x',y')\in \mathrm{N}_9(x,y)$ such that $B(x',y')=1$

$\mu_0$ = mean value of $Y(x',y')$, for all $(x',y')\in \mathrm{N}_9(x,y)$ such that $B(x',y')=0$.

Note that the algorithm uses $B$ instead of $B'$ to compute the mean values $\mu_1$ and $\mu_0$.

A hysteresis thresholding [C-8] is applied to $G$ restricted to pixels, which have been classified in C.4.1 as belonging to the plane regions. The lower threshold is 30 and the upper threshold is 40. The algorithm first identifies as edge all pixels $(x, y)$ 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, i.e., using the set $N_{13}(x, y)$ of the 13 closest pixels of pixel (x, y), similarly to equation C-4, ignoring the restriction to the plane regions. The pixels with value 1 in this dilation are classified as belonging to the edge regions.

### C.4.3 Texture regions

The texture regions consist of the pixels in $Y$, which were neither classified as belonging to the edge regions nor to the plane regions in the above sections.

## C.5 Objective measurement

Consider $S_b$, the image of magnitude of the Sobel's gradient [C-7] computed for a given component ($Y$, $Cb$ or $Cr$) 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 pixelwise absolute difference between $S_b$ and $S'_b$ is computed and the region R of pixels in $D_b$ that belong to a given context (plane, edge, or texture) is considered. The Absolute Sobel 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 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 $\{m_1^{(SIF)}, m_2^{(SIF)}, ..., m_9^{(SIF)}\}$, for the frame $f$ with respect to the MPEG-2 4:2:0 and MPEG-1 SIF reference codec operations over $\boldsymbol{O}$ (Figure C.1). These measures are used as references together with spatial $S$ and temporal $T$ attributes in order to determine a contextual impairment model for $\boldsymbol{I'}$ (clause C.7). The temporal attribute $T$ is the mean value of the pixelwise absolute difference between the segmentations of frames $f$ and $f - 1$, normalized within [0, 1]. The spatial attribute $S$ is defined as the ratio $m_7^{(SIF)}/m_7^{(420)}$, normalized within [0,1], where $m_7^{(SIF)}$ and $m_7^{(420)}$ are the corresponding ASDs for the texture regions in the component $Y$ of the frame $f$.

## C.6 Database of impairment models

The CPqD-IES algorithm uses a database of impairment models for scenes different from the original 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, $\left\{\overline{m}_{1,j}^{(420)}, \overline{m}_{2,j}^{(420)}, ..., \overline{m}_{9,j}^{(420)}\right\}$, and $\left\{\overline{m}_{1,j}^{(SIF)}, \overline{m}_{2,j}^{(SIF)}, ..., \overline{m}_{9,j}^{(SIF)}\right\}$ were computed over the frames of each scene $j$, $j = 1, 2, ..., 12$.

The values of $\overline{T}_j = \{27.01, 25.33, 45.54, 36.40, 32.02, 12.63, 28.38, 10.19, 0.01, 7.26, 7.60, 14.27\}$ were calculated as the average of the temporal attributes, computed as described in clause C.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 within [0%, 100%] for each scene $j$.

Each objective measure $\overline{m}_{i,j}$, $i = 1, 2, ..., 9$ and $j = 1, 2, ..., 12$, extracted from the database scenes (original and impaired) is related to a contextual impairment level $\overline{L}_{i,j}$, according to equation C-1. The values of $F_{i,j}$ and $G_{i,j}$ in equation C-1 were found for each scene $j$ by minimizing the mean squared error $E\left[\left(\overline{SL}_j - \overline{L}_{i,j}\right)^2\right]$.

Finally, the database of impairment models consists of five sets $\overline{F}_{i,j}, \overline{G}_{i,j}, \overline{T}_j, \overline{m}_{i,j}^{(420)}, \overline{m}_{i,j}^{(SIF)}$, $i = 1, 2, ..., 9$, of parameters for each scene $j$, $j = 1, 2, ..., 12$.

Table C.1 contains the values of $\overline{F}_{i,j}, \overline{G}_{i,j}, \overline{m}_{i,j}^{(420)}, \overline{m}_{i,j}^{(SIF)}$, where Y, Cb and Cr are the components of a frame and the suffixes P, E and T mean plane regions, edge regions and texture regions, respectively.

### Table C.1 – Impairment measures for 12 database scenes: $\overline{m}_{i,j}^{(420)}$, $\overline{m}_{i,j}^{(SIF)}$, $\overline{F}_{i,j}$, $\overline{G}_{i,j}$

| Scene j | $\overline{m}_{1,j}^{(420)}$ | $\overline{m}_{2,j}^{(420)}$ | $\overline{m}_{3,j}^{(420)}$ | $\overline{m}_{4,j}^{(420)}$ | $\overline{m}_{5,j}^{(420)}$ | $\overline{m}_{6,j}^{(420)}$ | $\overline{m}_{7,j}^{(420)}$ | $\overline{m}_{8,j}^{(420)}$ | $\overline{m}_{9,j}^{(420)}$ |
|---|---|---|---|---|---|---|---|---|---|
| | **YP** | **CbP** | **CrP** | **YE** | **CbE** | **CrE** | **YT** | **CbT** | **CrT** |
| 1 | 10.89 | 7.08 | 7.41 | 22.73 | 24.21 | 24.68 | 22.93 | 20.94 | 19.25 |
| 2 | 11.69 | 5.82 | 5.12 | 20.06 | 15.80 | 12.22 | 21.55 | 12.41 | 9.76 |
| 3 | 8.14 | 5.36 | 4.32 | 13.77 | 12.54 | 11.21 | 14.43 | 10.89 | 10.43 |
| 4 | 14.18 | 6.04 | 5.44 | 21.89 | 15.57 | 12.19 | 21.50 | 12.55 | 10.80 |
| 5 | 6.87 | 5.44 | 4.50 | 19.42 | 17.24 | 14.40 | 20.36 | 19.18 | 16.98 |
| 6 | 8.96 | 4.31 | 4.17 | 16.67 | 8.08 | 10.56 | 15.58 | 6.38 | 6.15 |
| 7 | 14.25 | 8.10 | 6.99 | 22.69 | 17.65 | 18.62 | 21.66 | 16.80 | 16.57 |
| 8 | 7.06 | 3.36 | 3.92 | 21.40 | 17.45 | 22.31 | 22.44 | 30.12 | 22.21 |
| 9 | 8.80 | 6.61 | 7.19 | 20.65 | 17.24 | 13.81 | 21.02 | 15.94 | 12.51 |
| 10 | 16.04 | 15.92 | 10.28 | 19.58 | 20.93 | 12.59 | 19.57 | 25.38 | 14.39 |
| 11 | 6.70 | 4.41 | 4.98 | 17.48 | 11.97 | 13.50 | 17.42 | 15.27 | 16.44 |
| 12 | 12.10 | 7.09 | 8.14 | 21.49 | 25.22 | 22.38 | 21.58 | 19.83 | 19.18 |
| **Scene j** | $\overline{m}_{1,j}^{(SIF)}$ | $\overline{m}_{2,j}^{(SIF)}$ | $\overline{m}_{3,j}^{(SIF)}$ | $\overline{m}_{4,j}^{(SIF)}$ | $\overline{m}_{5,j}^{(SIF)}$ | $\overline{m}_{6,j}^{(SIF)}$ | $\overline{m}_{7,j}^{(SIF)}$ | $\overline{m}_{8,j}^{(SIF)}$ | $\overline{m}_{9,j}^{(SIF)}$ |
| | **YP** | **CbP** | **CrP** | **YE** | **CbE** | **CrE** | **YT** | **CbT** | **CrT** |
| 1 | 21.65 | 9.88 | 10.97 | 83.35 | 35.03 | 35.40 | 72.68 | 28.46 | 27.22 |
| 2 | 17.85 | 6.97 | 5.90 | 68.56 | 20.50 | 15.08 | 53.11 | 15.88 | 11.67 |
| 3 | 12.57 | 7.05 | 5.20 | 32.24 | 17.38 | 14.42 | 32.80 | 13.54 | 12.48 |
| 4 | 21.77 | 7.17 | 6.20 | 75.52 | 20.10 | 15.20 | 61.95 | 15.88 | 13.17 |
| 5 | 11.79 | 6.07 | 5.24 | 88.69 | 22.84 | 19.79 | 84.11 | 25.62 | 23.44 |
| 6 | 11.85 | 4.63 | 4.49 | 43.21 | 9.83 | 13.09 | 26.98 | 7.04 | 6.79 |
| 7 | 21.35 | 8.62 | 8.13 | 110.79 | 20.89 | 23.41 | 88.45 | 20.36 | 21.23 |
| 8 | 9.97 | 3.82 | 4.48 | 70.29 | 22.30 | 30.25 | 72.19 | 40.05 | 28.45 |
| 9 | 18.27 | 7.98 | 8.20 | 61.46 | 21.79 | 16.82 | 54.04 | 19.81 | 15.33 |
| 10 | 27.18 | 22.33 | 12.85 | 42.67 | 30.69 | 15.35 | 39.09 | 36.27 | 19.24 |
| 11 | 8.66 | 4.70 | 5.73 | 51.38 | 14.96 | 18.01 | 42.91 | 19.17 | 21.01 |
| 12 | 13.99 | 10.33 | 11.11 | 76.35 | 43.82 | 35.30 | 62.17 | 31.69 | 31.35 |

**Table C.1 – Impairment measures for 12 database scenes: $\overline{m}_{i,j}^{(420)}$, $\overline{m}_{i,j}^{(SIF)}$, $\overline{F}_{i,j}$, $\overline{G}_{i,j}$**

| **Scene j** | $\overline{F}_{1,j}$ | $\overline{F}_{2,j}$ | $\overline{F}_{3,j}$ | $\overline{F}_{4,j}$ | $\overline{F}_{5,j}$ | $\overline{F}_{6,j}$ | $\overline{F}_{7,j}$ | $\overline{F}_{8,j}$ | $\overline{F}_{9,j}$ |
|---|---|---|---|---|---|---|---|---|---|
| | **YP** | **CbP** | **CrP** | **YE** | **CbE** | **CrE** | **YT** | **CbT** | **CrT** |
| 1 | 19.67 | 9.60 | 10.34 | 64.69 | 35.11 | 34.99 | 62.74 | 30.48 | 28.92 |
| 2 | 16.80 | 7.02 | 5.80 | 50.94 | 20.85 | 15.31 | 50.94 | 20.85 | 15.31 |
| 3 | 16.25 | 8.43 | 5.97 | 48.49 | 19.54 | 15.89 | 50.17 | 15.57 | 13.89 |
| 4 | 20.59 | 7.04 | 6.01 | 52.95 | 20.70 | 15.58 | 49.29 | 16.91 | 13.90 |
| 5 | 10.64 | 6.03 | 5.39 | 58.91 | 23.39 | 19.79 | 60.79 | 27.51 | 24.66 |
| 6 | 11.01 | 4.48 | 4.36 | 29.93 | 9.27 | 12.43 | 24.51 | 6.84 | 6.58 |
| 7 | 20.56 | 8.49 | 7.91 | 69.41 | 20.68 | 23.91 | 60.70 | 20.06 | 22.14 |
| 8 | 10.18 | 3.88 | 4.52 | 58.20 | 22.37 | 30.64 | 61.92 | 43.33 | 31.30 |
| 9 | 24.49 | 8.92 | 9.02 | 70.80 | 24.17 | 19.21 | 63.14 | 23.06 | 18.00 |
| 10 | 22.55 | 20.91 | 12.45 | 32.29 | 29.62 | 15.01 | 32.45 | 36.43 | 19.03 |
| 11 | 8.03 | 4.68 | 5.61 | 32.73 | 14.48 | 16.94 | 33.55 | 19.15 | 20.70 |
| 12 | 13.04 | 9.30 | 10.01 | 44.95 | 40.64 | 32.98 | 45.45 | 30.93 | 30.62 |
| **Scene j** | $\overline{G}_{1,j}$ | $\overline{G}_{2,j}$ | $\overline{G}_{3,j}$ | $\overline{G}_{4,j}$ | $\overline{G}_{5,j}$ | $\overline{G}_{6,j}$ | $\overline{G}_{7,j}$ | $\overline{G}_{8,j}$ | $\overline{G}_{9,j}$ |
| | **YP** | **CbP** | **CrP** | **YE** | **CbE** | **CrE** | **YT** | **CbT** | **CrT** |
| 1 | 1.85 | 4.17 | 3.80 | 1.27 | 4.38 | 4.61 | 1.63 | 5.36 | 4.99 |
| 2 | 3.52 | 8.17 | 10.04 | 1.50 | 8.26 | 8.80 | 2.36 | 8.63 | 10.54 |
| 3 | 3.69 | 6.52 | 9.67 | 2.09 | 6.73 | 7.65 | 2.35 | 8.25 | 9.32 |
| 4 | 2.84 | 10.70 | 8.95 | 1.15 | 5.56 | 5.24 | 1.31 | 5.23 | 5.06 |
| 5 | 5.25 | 14.15 | 12.98 | 2.00 | 10.02 | 8.41 | 3.37 | 13.05 | 12.69 |
| 6 | 4.07 | 19.74 | 11.09 | 1.26 | 10.68 | 7.10 | 1.63 | 11.86 | 7.54 |
| 7 | 4.42 | 8.98 | 8.96 | 1.81 | 9.17 | 6.63 | 2.08 | 7.32 | 6.29 |
| 8 | 2.19 | 8.71 | 10.86 | 1.69 | 8.74 | 8.73 | 2.64 | 8.97 | 9.20 |
| 9 | 3.11 | 6.45 | 7.26 | 2.33 | 5.69 | 8.05 | 2.69 | 5.66 | 5.83 |
| 10 | 6.49 | 15.92 | 10.79 | 2.02 | 9.03 | 8.06 | 2.57 | 7.88 | 7.96 |
| 11 | 5.50 | 4.71 | 5.78 | 1.50 | 2.27 | 3.80 | 1.78 | 3.70 | 3.86 |
| 12 | 13.04 | 9.30 | 10.01 | 44.95 | 40.64 | 32.98 | 45.45 | 30.93 | 30.62 |

## C.7 Estimation of impairment models

The contextual impairment models for a given frame *f* of ***I´*** consist of the parameters {$F_i$, $G_i$, $W_i$} of equations C-1 and C-2, *i* = 1, 2, ..., 9. This clause describes how to compute these parameters using the $I^{(420)}$ and $I^{(SIF)}$ impaired scenes as references.

### C.7.1 Computation of $W_i$

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

$$D_{i,j} = \frac{1}{2}.\left(\left|L_{i,j}^{(420)} - \overline{L}_{i,j}^{(420)}\right| + \left|L_{i,j}^{(SIF)} - \overline{L}_{i,j}^{(SIF)}\right|\right) \qquad \text{(C-8)}$$

where:

$$\begin{cases} \overline{L}_{i,j}^{(420)} = 100 / \left[ 1 + \left( \overline{F}_{i,j} / \overline{m}_i^{(420)} \right)^{\overline{G}_{i,j}} \right] \\ \overline{L}_{i,j}^{(SIF)} = 100 / \left[ 1 + \left( \overline{F}_{i,j} / \overline{m}_i^{(SIF)} \right)^{\overline{G}_{i,j}} \right] \\ L_{i,j}^{(420)} = 100 / \left[ 1 + \left( \overline{F}_{i,j} / m_i^{(420)} \right)^{\overline{G}_{i,j}} \right] \\ L_{i,j}^{(SIF)} = 100 / \left[ 1 + \left( \overline{F}_{i,j} / m_i^{(SIF)} \right)^{\overline{G}_{i,j}} \right] \end{cases} \quad \text{(C-9)}$$

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:

$$a_k = \begin{cases} 1, & \text{if (scene } k) \in \Omega, \\ 0, & \text{otherwise.} \end{cases} \quad \text{(C-10)}$$

$$W_{i,j} = \frac{a_j . D_{i,j}^{-1}}{\sum_{k=1}^{12} a_k . D_{i,k}^{-1}} \quad \text{(C-11)}$$

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

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

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

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

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

$$W_i = \frac{\kappa_i}{\tau} . \frac{1}{E_i} \quad \text{(C-12)}$$

### C.7.2 Computation of $F_i$ and $G_i$

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

$$L_i^{(420)} = \frac{1}{\gamma} . \sum_{j=1}^{12} W_{i,j} . L_{i,j}^{(420)} \quad \text{(C-13)}$$

$$L_i^{(SIF)} = \frac{1}{\gamma} . \sum_{j=1}^{12} W_{i,j} . L_{i,j}^{(SIF)} \quad \text{(C-14)}$$

where $\gamma$ is a factor restricted into [1/2, 2], which is computed based on the vector distances $D_j$ between the spatial and temporal attributes, $(S_j, T_j)$ and $(\bar{S}_j, \bar{T}_j)$, of the input scene and each database scene, respectively. The spatial attributes $\bar{S}_j$ of the database of impairment models are computed (see clause C.5) on the fly based on $\bar{}_{mi,j}(420)$, $\bar{}_{mi,j}(SIF)$, $i$=1, 2,..., 9 and $j$=1, 2,..., 12.

$$D_j = \left(S - \bar{S}_j\right)^2 + (T - \bar{T}_j)^2 \tag{C-15}$$

$$w_j = \frac{D_j^{-1}}{\sum_{k=1}^{12} D_k^{-1}}$$

$$a = \sum_{j=1}^{12} w_j \cdot \left[ \frac{\bar{S}_j \cdot \bar{T}_j}{2} + (1 - \bar{T}_j^2) \cdot \left( 1 - \frac{\bar{S}_j^2}{2} \right) \right] \tag{C-16}$$

$$b = \frac{S \cdot T}{2} + (1 - T^2) \cdot \left( 1 - \frac{S^2}{2} \right)$$

$$\gamma = 1 + a - b$$

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

$$L_i^{(420)} = 100 / \left[ 1 + \left( \frac{F_i}{m_i^{(420)}} \right)^{G_i} \right] \tag{C-17}$$

$$L_i^{(SIF)} = 100 / \left[ 1 + \left( \frac{F_i}{m_i^{(SIF)}} \right)^{G_i} \right] \tag{C-18}$$

## C.8 References

[C-1] Recommendation ITU-R BT.500-11 (2002), *Methodology for the subjective assessment of the quality of television pictures*.

[C-2] Recommendation ITU-R BT.802-1 (1994), *Test pictures and sequences for subjective assessments of digital codecs conveying signals produced according to Recommendation ITU-R BT.601*.

[C-3] ITU-T Tutorial (2004), *Objective perceptual assessment of video qality: Full reference television*.

[C-4] Recommendation ITU-R BT.601-5 (1995), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*.

[C-5] ITU-T Recommendation H.262 (2000) | ISO/IEC 13818-2:2000, *Information technology – Generic coding of moving pictures and associated audio information: Video*.

[C-6] ISO/IEC 11172-1:1993, *Information technology – Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s – Part 1: Systems*.

[C-7] Gonzalez, R.C. and Woods, R.E. (1992), *Digital Image Processing*, Addison-Wesley.

[C-8] Trucco, E. and Verri A. (1998), *Introductory Techniques for 3-D Computer Vision*, Prentice-Hall.

## C.9 Objective results in VQEG-phase II tests

### Table C.2 – 625/60 raw objective data

| 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 C.3 – 525/60 raw objective data

| 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* | 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 |

* 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 D

# National Telecommunications and Information Administration (NTIA)

# Video Quality Metric (VQM) technical description

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 ½ pixel and a temporal registration accuracy of plus or minus one interlaced field.

## D.1 Introduction

This annex provides a complete technical description of the National Telecommunications and Information Administration (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-2 tests only evaluated the performance of the General Model on MPEG-2 and 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 ½ pixel and a temporal registration accuracy of plus or minus one interlaced field.

NTIA has indicated its willingness to provide to all interested parties a software that implements the General Model and its associated automatic calibration techniques. Interested parties can find it here: www.its.bldrdoc.gov/n3/video/vqmsoftware.htm.

Disclaimer: In no event shall the ITU be liable for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the identified software. The ITU disclaims all warranties, express or implied, including but not limited to, warranties of merchantability and fitness for a particular purpose.

## D.2 References

### D.2.1 Normative References

– ITU-R Recommendation BT.601-5 (1995), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*.

## D.3 Definitions

**D.3.1 4:2:2**: A Y, Cb, Cr image sampling format where chrominance planes (Cb and Cr) are sampled horizontally at half the luminance (Y) plane's sampling rate. See ITU-R Rec. BT.601-5 (clause D.2).

**D.3.2 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 D.7.5 for the precise mathematical definition.

**D.3.3 big YUV**: The binary file format used for storing clips that have been sampled according to ITU-R Rec. BT.601-5. 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 ITU-R Rec. BT.601-5 sampling. The Y represents the luminance channel information, the U represents the blue color difference channel (i.e., $C_B$ in ITU-R Rec. BT.601-5), and the V represents the red color difference channel (i.e., $C_R$ in ITU-R Rec. BT.601-5). The pixel ordering in the binary file is the same as that specified in SMPTE 125M [D-7]. The full specification of the Big YUV file format is given in clause D.5 and software routines for reading and displaying Big YUV files are given in [D-14].

**D.3.4 clip**: Digital representation of a scene that is stored on computer media.

**D.3.5 clip VQM**: The VQM of a single clip of processed video.

**D.3.6 chrominance (C, $C_B$, $C_R$)**: The portion of the video signal that predominantly carries the color information (C), perhaps separated further into a blue color difference signal ($C_B$) and a red color difference signal ($C_R$).

**D.3.7 codec**: Abbreviation for a coder/decoder or compressor/decompressor.

**D.3.8 Common Intermediate Format (CIF)**: A video sampling structure used for video teleconferencing where the luminance channel is sampled at 352 pixels by 288 lines [D-2].

**D.3.9 feature**: A quantity of information associated with, or extracted from, a spatial-temporal subregion of a video stream (either an original video stream or a processed video stream).

**D.3.10 field**: One half of a frame, containing all of the odd or even lines.

**D.3.11 frame**: One complete television picture.

**D.3.12 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.

**D.3.13 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.

**D.3.14 general model**: The video quality model, or VQM, that is the subject of this annex (clause D.9). The General Model was submitted to the phase-2 tests performed by the Video Quality Experts Group (VQEG). The VQEG Phase-2 final report describes the performance of the General Model (see [D-15], proponent H).

**D.3.15 H.261**: Abbreviation for ITU-T Recommendation H.261 [D-2].

**D.3.16 Hypothetical Reference Circuit (HRC)**: A video system under test such as a codec or digital video transmission system.

**D.3.17 input Video**: Video before being processed or distorted by an HRC (see Figure D.1). Input video may also be referred to as Original Video.

**D.3.18 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.

**D.3.19 International Telecommunication Union (ITU)**: An international organization within the United Nations System where governments and the private sector coordinate global telecommunication networks and services. The ITU includes the Radiocommunication Sector (ITU-R), the Telecommunication Standardization Sector (ITU-T) and the Telecommunication Development Sector (ITU-D).

**D.3.20 luminance (Y)**: The portion of the video signal that predominantly carries the luminance information (i.e., the black and white part of the picture).

**D.3.21 Mean Opinion Score (MOS)**: The average subjective quality judgment assigned by a panel of viewers to a processed video clip.

**D.3.22 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).

**D.3.23 National Television Systems Committee (NTSC)**: The 525-line analog color video composite system [D-8].

**D.3.24 offset or level offset**: An additive factor applied by the hypothetical reference circuit (HRC) to all pixels of an individual image plane (e.g., luminance, chrominance). Offset of the luminance signal is commonly known as brightness.

**D.3.25 Original Region of Interest (OROI)**: A Region of Interest (ROI) extracted from the original video, specified in Rectangle Coordinates.

**D.3.26 original video**: Video before being processed or distorted by an HRC (see Figure D.1). Original video may also be referred to as input video since this is the video input to the digital video transmission system.

**D.3.27 Original Valid Region (OVR)**: The Valid Region of an original video clip, specified in Rectangle Coordinates.

**D.3.28 output video**: Video that has been processed or distorted by an HRC (see Figure D.1). Output video may also be referred to as Processed Video.

**D.3.29 over-scan**: The portion of the video that is not normally visible on a standard television monitor.

**D.3.30 Phase-Altering Line (PAL)**: The 625-line analog color video composite system.

**D.3.31 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.

**D.3.32 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.

**D.3.33 processed video**: Video that has been processed or distorted by an HRC (see Figure D.1). Processed video may also be referred to as output video since this is the video output from the digital video transmission system.

**D.3.34 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 original valid region (OVR). The region between the PVR and the OVR is that portion of the video that was blanked or corrupted by the HRC.

**D.3.35 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 ITU-R Rec. BT.601-5 sampled video, the Production Aperture is 720 pixels × 486 lines for 525-line systems and 720 pixels × 576 lines for 625-line systems [D-9].

**D.3.36 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 [D-2].

**D.3.37 BT.601-5**: Abbreviation for ITU-R Recommendation BT.601-5 (clause D.2), a common 8-bit video sampling standard that samples the luminance (Y) channel at 13.5 MHz, and the blue and red color difference channels ($C_B$ and $C_R$) at 6.75 MHz. See clause D.5 for more information.

**D.3.38 rectangle coordinates**: A rectangular-shaped image subregion that is completely contained within the production aperture and 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 D.5.3.

**D.3.39 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 [D-2]. 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.

**D.3.40 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 D.6.1.2.

**D.3.41 Region of Interest (ROI)**: An image lattice (specified in Rectangle Coordinates) that is used to denote a particular subregion of a field or frame of video. Also see SROI.

**D.3.42 scene**: A sequence of video frames.

**D.3.43 Spatial Information (SI)**: A feature based on statistics that are extracted from the spatial gradients (i.e., edges) of an image or video scene. Reference [D-4] provides a definition of SI based on statistics extracted from 3 × 3 Sobel-filtered images [D-6] while D.7.2.2 provides a definition of SI based on statistics extracted from much larger 13 × 13 edge-filtered images (Figure D.11).

**D.3.44 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 BT.601-5 sampled video, the recommended SROI is 672 pixels × 448 lines for 525-line systems and 672 pixels × 544 lines for 625-line systems, centered 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.

**D.3.45 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.

**D.3.46 Spatial-Temporal (S-T) subregion**: 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 Figure D.9.

**D.3.47 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.

**D.3.48 Temporal Information (TI)**: A feature based on statistics that are extracted from the temporal gradients (i.e., motion) of a video scene. Reference [D-4] and D.7.5 all provide definitions of TI based on statistics extracted from simple frame differences.

**D.3.49 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.

**D.3.50 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 D.6.4.1).

**D.3.51 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.

**D.3.52 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 D.6.4.

**D.3.53 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.

**D.3.54 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 International Telecommunication Union (ITU) and may be used as the basis for international video quality measurement recommendations.

**D.3.55 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.

## D.4 Overview of the Video Quality Metric (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 [D-1]. Figure D.1 gives an overview diagram of the processes required to compute the General VQM. These processes include sampling of the original and processed video streams (clause D.5), calibration of the original and processed video streams (clause D.6), extraction of perception-based features (clause D.7), computation of video quality parameters (clause D.8), and calculation of the General Model (clause D.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 [D-3]. These reduced reference parameters utilize features extracted from spatial-temporal (S-T) regions of the video sequence (see D.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 Figure D.1.

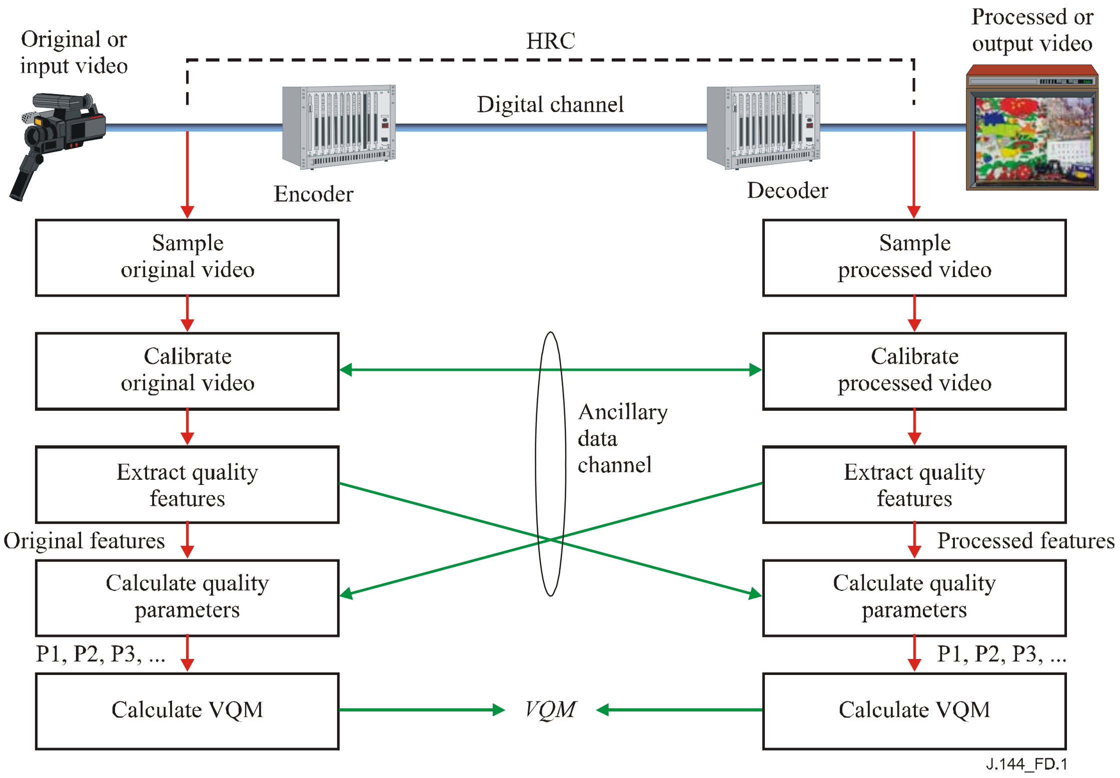


**Figure D.1 – Steps required to compute VQM**

## D.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 analog format, one of the most widely used digital sampling standards is BT.601-5 (clause D.2). Composite video such as NTSC and PAL must first be converted into component video that contains the following three signals: luminance (Y), blue color difference ($C_B$), and red color difference ($C_R$). BT.601-5 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. BT.601-5 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 BT.601-5 sampled Y video frame will be 720 pixels by 486 lines. Likewise, when 625-line PAL video is sampled according to BT.601-5, the Y video frame will contain 720 pixels by 576 lines. If 8 bits are used to uniformly sample the Y signal, BT.601-5 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 analog 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".

### D.5.1 Temporal indexing of original and processed video files

A luminance video frame that results from BT.601-5 sampling will be denoted as $\mathbf{Y}(t)$. 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 Figure D.2, $t = 0$ denotes the first frame that was sampled and $t$ = (N–1) denotes the last frame that was sampled.

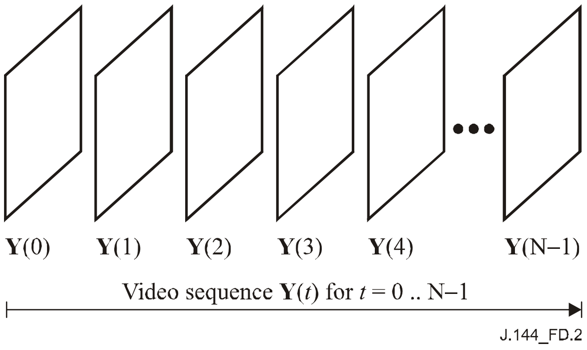


**Figure D.2 – Temporal indexing of frames in Big YUV files**

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 Figure D.1. Therefore, the uncertainty that remains in the video delay estimate will be denoted as plus or minus **U**.

### D.5.2 Spatial indexing of original and processed video frames

The coordinate system used for the sampled luminance frames is shown in Figure D.3. 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 $Y(i, j, t)$.

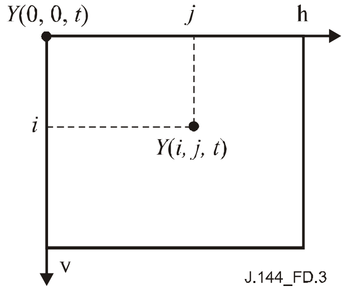


**Figure D.3 – Coordinate system used for sampled luminance Y frames**

A clip of video sampled according to BT.601-5 is stored in "Big YUV" file format, where the Y denotes the BT.601-5 luminance information, the U denotes the blue color-difference information (i.e., $C_B$ in BT.601-5), and the V denotes the red color-difference information (i.e., $C_R$ in BT.601-5). 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 [D-7].

### D.5.3 Specifying rectangular subregions

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 subregions will use rectangle coordinates defined by the four quantities top, left, bottom, and right. Figure D.4 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 subregion definition used in Figure D.4 defines the grid used to display these abutted subregions and the math used to extract features from each abutted subregion.

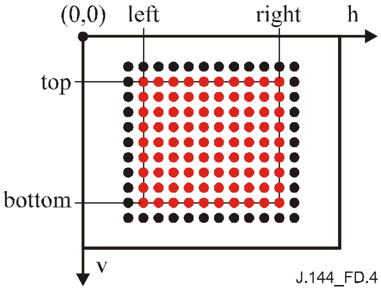


**Figure D.4 – Rectangle coordinates for specifying image subregions**

### D.5.4 Considerations for video sequences longer than 10 seconds

The video quality measurements in this annex were based upon subjective test results that utilized 8 to 10-second 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.

## D.6 Calibration

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

1) spatial registration estimation and correction;

2) valid region estimation to limit the extraction of features to those pixels that contain picture information;

3) gain and level offset estimation and correction (commonly known as contrast and brightness); and

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 [D-5] 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 (clauses D.7 and D.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 D.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.

### D.6.1 Spatial registration

#### D.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 D.6.1.6.

Spatial registration must be determined before processed valid region (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.

### D.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 Figure D.5, 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.

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.

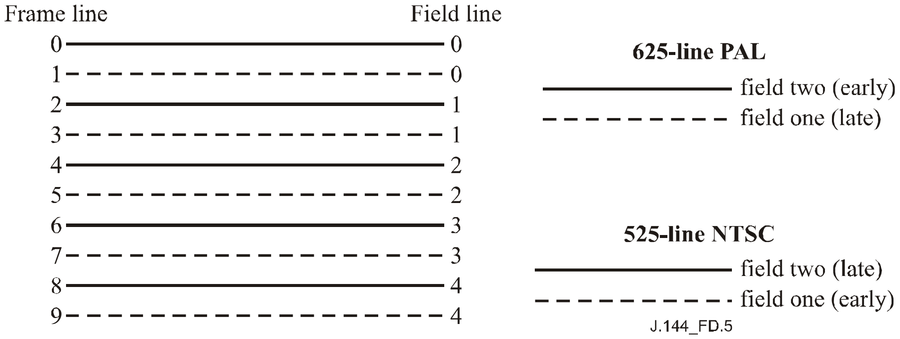


**Figure D.5 – Diagram depicting interlaced fields and frame/field line numbering scheme**

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.

#### D.6.1.3 Required inputs to the spatial registration algorithm

This clause 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 miss-guesses the search uncertainties by a small amount, the undesirable side effect is to dramatically increase run time when the user miss-guesses by a large amount. Alternatively, the search algorithm may fail to find the correct spatial shift in this case.

##### D.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 ITU-R Rec. BT.601-5, 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 BT.601-5 (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.

##### D.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).

##### D.6.1.3.3 Processed Valid Region (PVR) guess

The processed valid region (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 BT.601-5, 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 BT.601-5, 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.

#### D.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.

##### D.6.1.4.1 Region of Interest (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 processed valid region (PVR) for all possible spatial shifts that will be examined.

– The OROI is centered within the original image.

##### D.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 D.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 is unaffected by level offsets (see D.6.1.4.3).

##### D.6.1.4.3 Formulae used to compare PROI with OROI

After correcting the PROI for gain[6] (see D.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] 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 D.6.1.5.2 and D.6.1.5.3).

### D.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 D.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 clause 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.

#### D.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 frequency$^{th}$ 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 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.

#### D.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 Figure D.5) 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 Figure D.6). In Figure D.6, 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 D.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 (see D.6.1.5.3). According to the coordinate system in Figure D.3, 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.

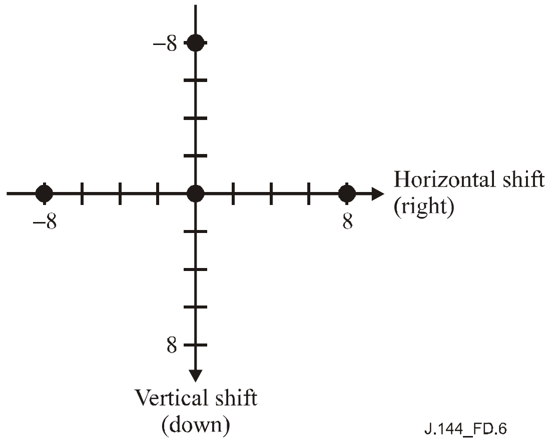


**Figure D.6 – Spatial shifts considered by the broad search for the temporal shift**

### D.6.1.5.3 Broad search for the spatial shift

Using the temporal registration found by the broad search for temporal shift (see D.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 D.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 Figure D.7. Notice that fewer downward shifts are considered (as in D.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 D.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 D.6.1.5.4.

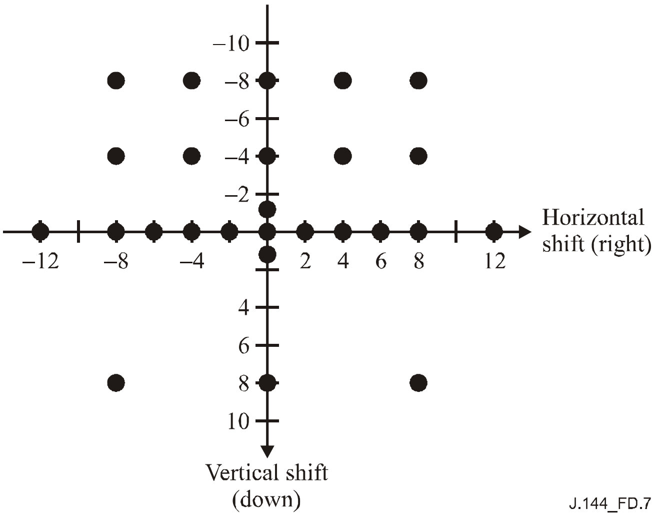


**Figure D.7 – Spatial shifts considered by the broad search for the spatial shift**

### D.6.1.5.4 Fine search for the spatial-temporal shift

The fine search includes a much smaller set of shifts centered around the current spatial registration estimate and just five fields centered 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 Figure D.8). In the example shown in Figure D.8, 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 Figure D.8 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 D.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 D.6.1.5.5.

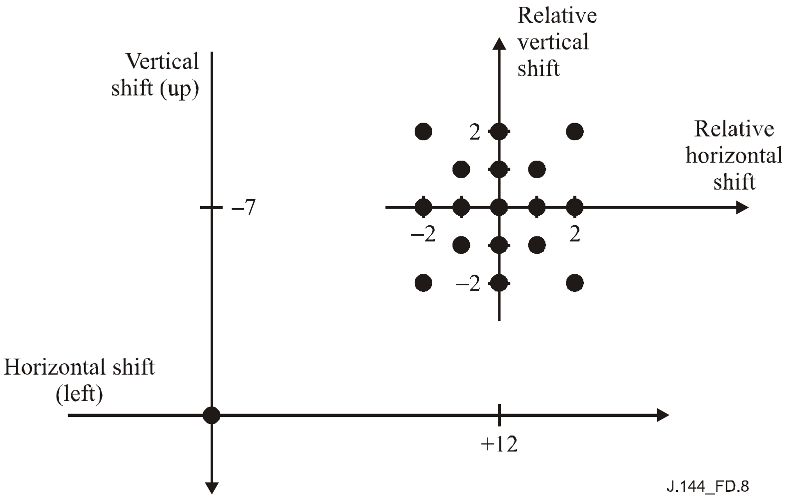


**Figure D.8 – Spatial shifts considered by the fine search for the spatial shift**

### D.6.1.5.5 Repeated fine searches

Iteration through the fine search of D.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 D.6.1.4.2).

Repeated fine searches are performed on the processed field (see D.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 (see D.6.1.5.6), and multiple scenes are examined for each HRC (see D.6.1.5.7).

#### D.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 D.6.1.5.2). Notice that this broad search will search the first $\mathbf{U} \times 2 + 1$ frames of the original video sequence for a field type one that best aligns. Then, a broad search for the spatial shift is performed centered on this best-aligned original field (see D.6.1.5.3). Next, perform up to five fine spatial-temporal searches to fine-tune the spatial and temporal estimates (see D.6.1.5.4 and D.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 frequency$^{th}$ (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 frequency$^{th}$ (**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.[7]

#### D.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.

#### D.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 a 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.

### D.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 D.6.1.5.3) the numbers on the vertical axis in Figure D.7 must be doubled (e.g., +8 becoming +16 and –4 becoming –8).[8] 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 Figure D.7, 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 D.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 colorbar 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 D.6.1.5.4. Here, the vertical shifts remain unchanged, lying between –2 frame lines and +2 frame lines. Thus, the vertical axis of Figure D.8 is interpreted as referring to frame lines. The temporal extent of this fine search may be set to five original frames centered on the current aligned original frame, instead of

[7] 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.

[8] 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.

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, 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.

### D.6.2 Valid Region

NTSC (525-line) and PAL (625-line) video sampled according to BT.601-5 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 BT.601-5 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 BT.601-5 frame, black border pixels, or transitional effects where the black border meets the picture area.

#### D.6.2.1 Core valid region algorithm

This clause 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 BT.601-5 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 spatial registration clause D.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 clause D.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.

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 "$M_{J-1}$".

2) Take the mean of the next column of pixels, "$M_J$".

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 is true, increment J and repeat steps 2 and 3. Otherwise, go to step 4.

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.

### D.6.2.2 Applying the core valid region algorithm to a video sequence

#### D.6.2.2.1 Original video

The core algorithm is first applied to the original sequence of images. For NTSC video sampled according to BT.601-5 (see clause D.5), the recommended setting for the maximum valid region is top = 6, left = 6, bottom = 482, right = 714. For PAL video sampled according to BT.601-5, 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 frequency$^{th}$ 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, the bottom one is decremented; likewise, if the region contains an odd number of pixels (e.g., horizontally), the right is decremented. This simplifies color processing for video sampled in accordance with BT.601-5, since the color 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 subregions (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.

#### D.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 frequency$^{th}$ image thereafter (i.e., if frequency = F, use images $\mathbf{Y}(0)$, $\mathbf{Y}(F)$, $\mathbf{Y}(2F)$, $\mathbf{Y}(3F)$, 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, the bottom one is decremented; likewise, if the region contains an odd number of pixels (e.g., horizontally), the right is decremented. The resulting valid region is returned as the processed valid region.

### D.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 BT.601-5 (e.g., the common intermediate format, or CIF, used by ITU-T Rec. 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.

## D.6.3 Gain and offset

### D.6.3.1 Core gain and level offset algorithm

This clause 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 D.6.1). The original and processed images must also be temporally registered, which will be addressed later in D.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 BT.601-5 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 subregions. For each of the subregions, 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:

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

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 subregions), we must solve the over-determined system of linear equations given by:

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

where $A$ is an N × 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.

### D.6.3.2 Using scenes

The basic algorithm given in D.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 subregion 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 color image planes). The mean over space of the [Y, $C_B$, $C_R$] samples is computed for each corresponding original and processed subregion, or block, to form a spatially sub-sampled image. All the selected blocks must lie within the processed valid region (PVR).

#### D.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 D.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 D.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 D.6.1.4.3, except perform 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 $C_R$.

#### D.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, $C_B$, $C_R$].

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

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: $\underline{E} = \left|\underline{P} - \underline{\hat{P}}\right|$.

3) Generate a cost vector ($\underline{C}$) that is the element-by-element reciprocal of the error vector ($\underline{E}$) plus a small epsilon ($\varepsilon$): $\underline{C} = \dfrac{1}{\underline{E} + \varepsilon}$. The $\varepsilon$ 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 $\varepsilon$ of 0.1 is recommended.

4) Normalize the cost vector $\underline{C}$ for unity norm (i.e., each element of $\underline{C}$ is divided by the square root of the sum of the squares of all the elements of $\underline{C}$).

5) Generate the cost vector $\underline{C}^2$ that is the element-by-element square of the cost vector $\underline{C}$ from step 4.

6) Generate an N x N diagonal cost matrix ($C^2$) that contains the cost vector's elements ($\underline{C}^2$) arranged on the diagonal, with zeros everywhere else.

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: $\begin{bmatrix} l \\ g \end{bmatrix} = \left(A^{\mathrm{T}} C^2 A\right)^{-1} A^{\mathrm{T}} C^2 \underline{P}$.

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.

#### D.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 frequency$^{th}$ 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.

### D.6.3.3 Applying gain and level offset corrections

The temporal registration algorithms (see D.6.4) and most quality features (clause D.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 color planes (i.e., $C_B$ and $C_R$). Perceptual chrominance errors are instead captured by the color metrics. The $C_B$ and $C_R$ image planes may be gain and level offset corrected for display purposes.

## D.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 Figure D.1 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 clause 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.

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

This clause 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.

#### D.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.

#### D.6.4.1.2 Inputs to the algorithm

A sequence of N original video luminance images: $\mathbf{Y_O}(t)$, $0 \le t < \mathrm{N}$.[9]

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

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 $\mathbf{Y_P}(t)$ is within plus or minus (U – HFW) of $\mathbf{Y_O}(t)$, for all $0 \le t < \mathrm{N}$.

#### D.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 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

[9] 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 Figure D.2.

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.

#### D.6.4.1.4 Description of the algorithm

1) *Calibrate the video sequences*

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

2) *Select the subregion of video to be used*

   The subregion of interest to be used by the algorithm must be a multiple of the BLOCK_SIZE and must fit within the PVR. The largest subregion 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 subregion of interest.

3) *Spatially sub-sample the original and processed images*

   Spatially sub-sample the region of interest of $\mathbf{Y_O}(t)$ and $\mathbf{Y_P}(t)$ 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.

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 color).[10] 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 $\mathbf{S_O}(t)$ and $\mathbf{S_P}(t)$, respectively, to denote that the images have been sub-sampled and normalized.

5) *Compare processed images to original images*

   Compare each processed image, $\mathbf{S_P}(t)$, with the original images $\mathbf{S_O}(t+d)$, where the valid values of $d$ are: ($-U \le d \le +U$) and the valid values of $t$ are: ($U \le t < N - U$). For processed image $t$ and original image $t+d$, these comparisons will be denoted as $\mathbf{C}_{td}$ and are computed as the standard deviation over space of the image formed by subtracting processed image $t$ from original image $t+d$: $\mathbf{C}_{td}=std_{\text{space}}(\mathbf{S_O}(t+d)-\mathbf{S_P}(t))$. These comparisons, $\mathbf{C}_{td}$, correlate the $t^{\text{th}}$ processed image with each original image that is within the registration uncertainty. Lower values of $\mathbf{C}_{td}$ indicate that the processed image looks more like the original image since more of the image variance is cancelled. The range for $t$, $U \le t < N - U$, covers all processed images for which original images are available for the entire range of temporal registration uncertainty.

[10] 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.

6) *Perform an overall check for still video*

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

$$A_d = \frac{1}{N-2\times U}\sum_{t=U}^{N-U-1}\mathbf{C}_{td}$$

This summation 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.

7) *Temporally register each processed image*

For each processed image $t$ ($U \le t < N - U$), find the $d$ within the temporal uncertainty ($-U \le d \le +U$) that minimizes $\mathbf{C}_{td}$. In other words, for each processed image $t$, find $d_{\min}(t)$ such that $\mathbf{C}_{t\ d\min(t)} \le \mathbf{C}_{td}$, 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.).

8) *Perform a stillness check on each processed image*

If, for a given processed image $t$ and all values of $d$ ($-U \le d \le U$), maximum($\mathbf{C}_{td}$) – minimum($\mathbf{C}_{td}$) < 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.

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.

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:

$$F_k = \frac{0.5+0.5\times\cos[\pi\times(k-\mathrm{HFW})/(1+\mathrm{HFW})]}{\sum_{i=0}^{2\times\mathrm{HFW}}\{0.5+0.5\times\cos[\pi\times(i-\mathrm{HFW})/(1+\mathrm{HFW})]\}},\quad 0\le k\le 2\times\mathrm{HFW}$$

When considering the smoothed histogram $SH_d$ 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.

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 ($-U \leq d < -U + HFW$) and ($U - HFW < d \leq U$).
- 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 ($-U \leq d <$ max_SH_offset – DELTA) or (max_SH_offset + DELTA $< d \leq U$), 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.

### D.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 D.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.

### D.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.

## D.7 Quality features

### D.7.1 Introduction

A quality *feature* is defined as a quantity of information associated with, or extracted from, a spatial-temporal subregion 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* (clause D.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) subregions 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 clauses, 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 clause D.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).[11]

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

1) [Optional] Apply a perceptual filter.

2) Divide the video stream into S-T regions.

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

4) [Optional] Apply a perceptibility threshold.

Some features may utilize two or more different perceptual filters.

#### D.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:

1) the number of pixels horizontally;

2) the number of frame lines vertically; and

[11] 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 clause D.9 for further comments regarding the assumed viewing distance.

3) the time duration of the region, given in units of equivalent video frames referenced to a 30 fps video system.[12]

Figure D.9 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 clause D.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 clause D.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 [D-12].

[12] 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.

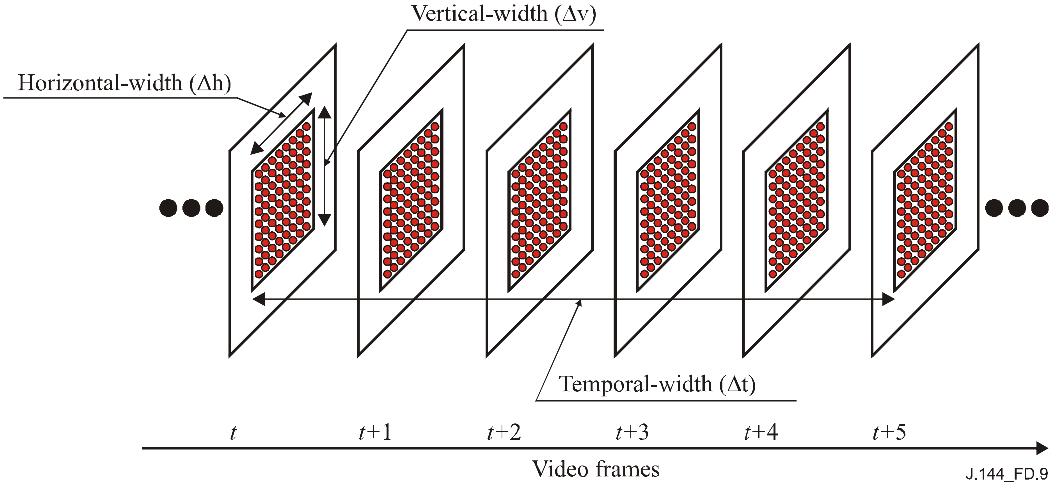


**Figure D.9 – Example spatial-temporal (S-T) region size for extracting features**

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.

### D.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 spatial-temporal (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 BT.601-5 video that has been subjectively evaluated at a viewing distance of six picture heights. Figure D.10 presents an overview of the algorithm used to extract features based on spatial gradients.

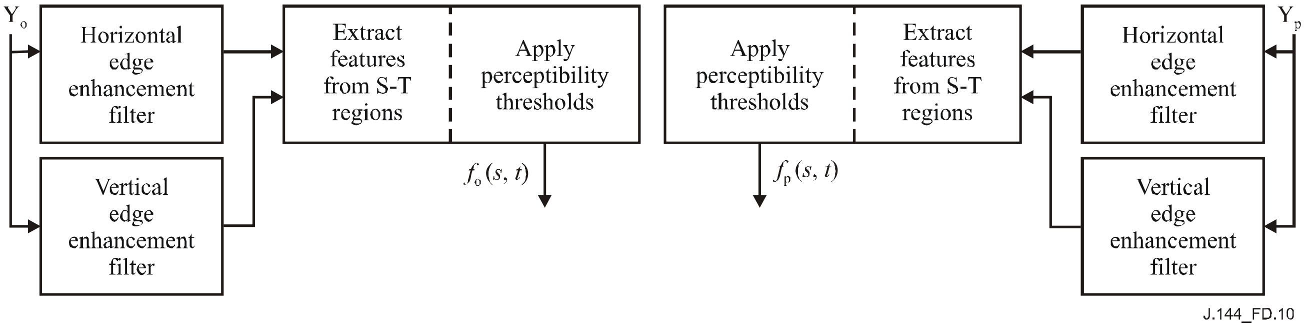

$Y_o$
Horizontal edge enhancement filter
Vertical edge enhancement filter
Extract features from S-T regions
Apply perceptibility thresholds
$f_o(s, t)$
Apply perceptibility thresholds
Extract features from S-T regions
$f_p(s, t)$
Horizontal edge enhancement filter
Vertical edge enhancement filter
$Y_p$
J.144_FD.10

**Figure D.10 – Overview of algorithm used to extract spatial gradient features**

### D.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 Figure D.11 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).

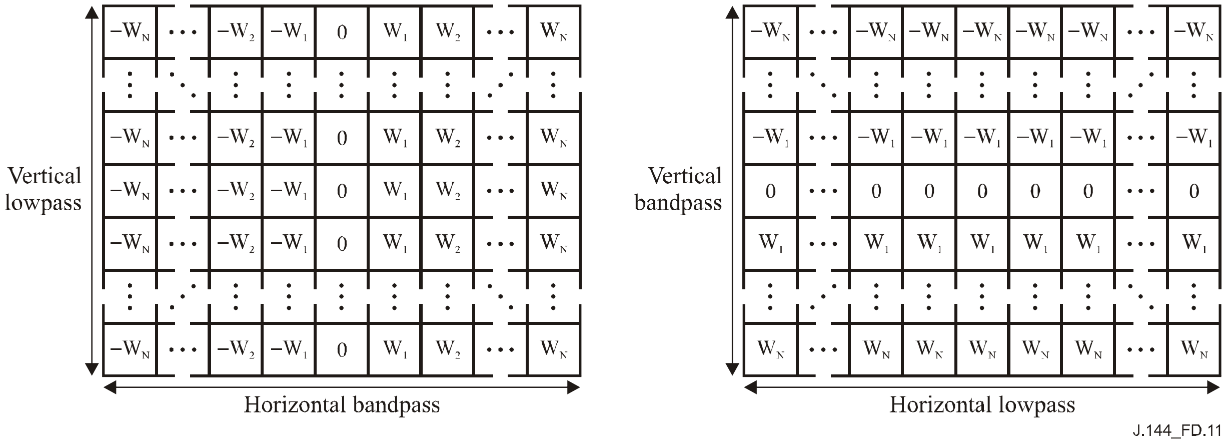

–W_N ··· –W_2 –W_1 0 W_1 W_2 ··· W_N
Vertical lowpass
Horizontal bandpass
–W_N ··· –W_N –W_N –W_N –W_N –W_N ··· –W_N
–W_1 ··· –W_1 –W_1 –W_1 –W_1 –W_1 ··· –W_1
0 ··· 0 0 0 0 0 ··· 0
W_1 ··· W_1 W_1 W_1 W_1 W_1 ··· W_1
W_N ··· W_N W_N W_N W_N W_N ··· W_N
Vertical bandpass
Horizontal lowpass
J.144_FD.11

**Figure D.11 – Edge enhancement filters**

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

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

where $x$ is the pixel displacement from the centre of the filter (0, 1, 2, …, N), $c$ is a constant that sets the width of the bandpass filter, and $k$ is a normalization constant selected such that each filter would produce the same gain as a true Sobel filter [D-6]. 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:

[–.0052625, –.0173446, –.0427401, –.0768961, –.0957739, –.0696751, 0, .0696751, .0957739, .0768961, .0427401, .0173446, .0052625].

Note that the filters in Figure D.11 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 Figure D.11, one merely has to sum the pixels in a column and multiply once by the weight).

### D.7.2.2 Description of Features $f_{SI13}$ and $f_{HV13}$

This clause describes the extraction of two spatial activity features from S-T regions of the edge-enhanced original and processed video streams from D.7.2.1. These features will be used to detect spatial impairments such as blurring and blocking. The filter shown in Figure D.11 (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 Figure D.12 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

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

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

The first feature is a measure of overall spatial information (SI) and hence is denoted as $f_{\text{SI13}}$ since images were preprocessed using the 13 × 13 filter masks shown in Figure D.11. 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*, $f_{\text{SI13}}$ is set equal to *P*), namely

$$f_{SI13}=\left\{std\left[R(i,j,t)\right]\right\}\big|_P : \; i,j,t\in\left\{\text{S}-\text{T Region}\right\}$$

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.

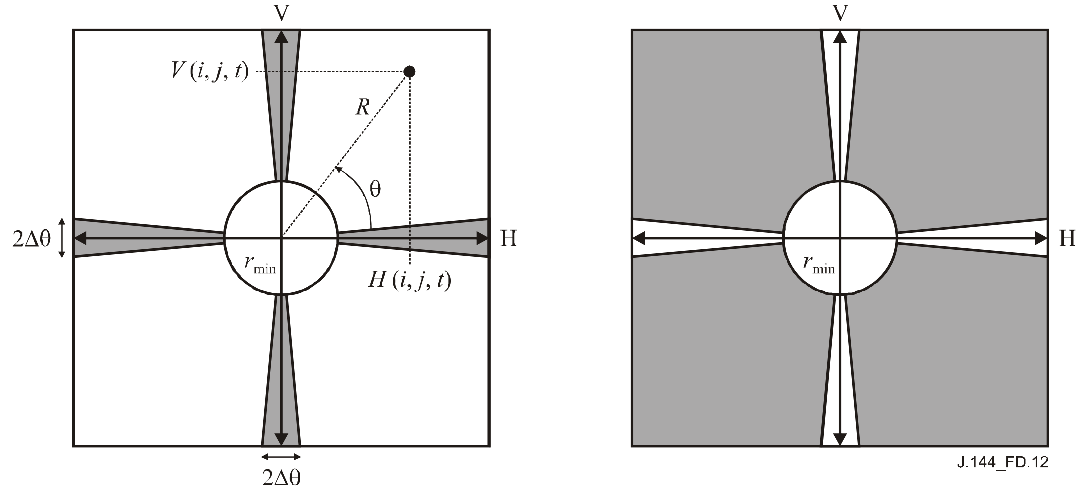


**Figure D.12 – Division of horizontal (H) and vertical (V) spatial activity into *HV* (left) and $\overline{HV}$ (right) distributions**

The second feature, $f_{\text{HV13}}$, 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 Figure D.12. 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

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

and

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

where

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

For the computation of $HV$ and $\overline{HV}$ above, the recommended value for $r_{\min}$ is 20 and the recommended value for $\Delta\theta$ is 0.225 radians. Feature $f_{\text{HV13}}$ 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

$$f_{HV13}=\{mean[HV(i,j,t)]\}|_P \Big/ \{mean[\overline{HV}(i,j,t)]\}|_P$$

The recommended perceptibility threshold $P$ for the mean of $HV$ and $\overline{HV}$ is 3. The $f_{\text{HV13}}$ 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, $f_{\text{HV13}}$ of the processed video will be less than $f_{\text{HV13}}$ 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 $f_{\text{HV13}}$ of the processed video will be greater than $f_{\text{HV13}}$ of the original video. The $f_{\text{HV13}}$ feature thus provides a simple means to include variations in the sensitivity of the human visual system with respect to angular orientation.[13]

### D.7.3 Features based on chrominance information

This clause presents a single feature that can be used to measure distortions in the chrominance signals ($\text{C}_\text{B}$, $\text{C}_\text{R}$). For a given image pixel located at row $i$, column $j$, and time $t$, let $C_\text{B}(i, j, t)$ and $C_\text{R}(i, j, t)$ represent the BT.601-5 $\text{C}_\text{B}$ and $\text{C}_\text{R}$ values.[14] The components of a two-dimensional chrominance feature vector, $\underline{f}_{\text{COHER\_COLOR}}$, are computed as the mean ($mean$) over the S-T region of the $C_\text{B}(i, j, t)$ and $C_\text{R}(i, j, t)$ samples, respectively, giving more perceptual weight to the $\text{C}_\text{R}$ component:

$$\underline{f}_{COHER\_COLOR}=(mean[C_B(i,j,t)], W_R\times mean[C_R(i,j,t)]):\; i,j,t\in\{\text{S}-\text{T Region}\},$$
$$\text{and } W_R=1.5$$

The above equation performs coherent integration (hence the name $\underline{f}_{\text{COHER\_COLOR}}$) since the phase relationship between $\text{C}_\text{B}$ and $\text{C}_\text{R}$ is preserved. If one is familiar with a vectorscope, the value of the chrominance feature when examining color 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 colors could be included in the calculation, and the usefulness of $\underline{f}_{\text{COHER\_COLOR}}$ is reduced. An S-T region size of 8 horizontal pixels × 8 vertical lines × (1 to 3) video

[13] 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).

[14] Gain and offset corrections are not applied to the $\text{C}_\text{B}$ and $\text{C}_\text{R}$ image planes. See D.6.3.3.

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 BT.601-5 sampling).

### D.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

$$f_{CONT} = \{ std[Y(i,j,t)]\}\big|_P : \; i,j,t \in \{\text{S} - \text{T Region}\}$$

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

### D.7.5 Features based on absolute temporal information (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 absolute temporal information 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

$$f_{ATI} = \{ std \; |Y(i,j,t) - Y(i,j,t-1)|\}\big|_P : \; i,j,t \in \{\text{S} - \text{T Region}\}$$

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 D.7.6) or for use in a model (see clause D.9), the requirement of an extra frame complicates the task of placement of S-T regions (see D.7.1.1).

### D.7.6 Features based on the cross product of contrast and absolute temporal information

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 D.7.4 and D.7.5.[15] 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.

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

## D.8 Quality parameters

### D.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, $f_p(s, t)$ (see D.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 (see D.8.3), and error pooling across time will be referred to as temporal collapsing (see D.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 seconds in duration. The final time-collapsed parameter values may be scaled and clipped (see D.8.5) to account for nonlinear 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.

1) Compare original feature values with processed feature values.

2) Perform spatial collapsing.

3) Perform temporal collapsing.

4) [Optional] Perform nonlinear scaling and/or clipping.

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

### D.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 clause presents the masking functions that are used by the various parameters to produce quality parameters as a function of space and time.

#### D.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:

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

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

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

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

where *pp* is the positive part operator (i.e., negative values are replaced with zero), and *np* is the 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.

### D.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, $f_p(s, t)$,

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

Figure D.13 gives an illustration of Euclidean distance for a two-dimensional feature vector extracted from a S-T region (e.g., the $\underline{f}_{\text{COHER\_COLOR}}$ feature vector of D.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 Figure D.13 shows the Euclidean distance. The Euclidean distance measure can be generalized for feature vectors that have an arbitrary number of dimensions.

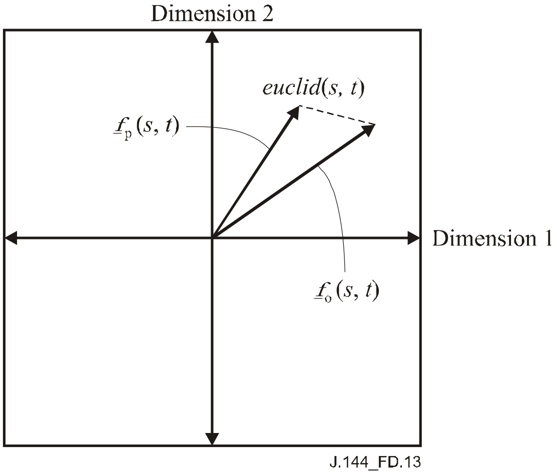


**Figure D.13 – Illustration of the Euclidean distance *euclid*(*s, t*) for a two-dimensional feature vector**

### D.8.3 Spatial collapsing functions

The parameters from the S-T regions (from D.8.2) form three-dimensional matrixes 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 D.8.4. Table D.1 presents a summary of the most commonly used spatial collapsing functions.

**Table D.1 – 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. |

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* [D-10]-[D-13]. 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 D.8.2.1 as the average of the most positive 5% of the values over the spatial index *s*.[16] 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 D.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).

### D.8.4 Temporal collapsing functions

The parameter time history results *p*(*t*) output from the spatial collapsing function (from D.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 seconds in length. Viewers seem to use several temporal collapsing functions when subjectively rating video clips that are approximately 10 seconds in length. The *mean* over time is indicative of the average quality that is observed during the time period. The *90%*

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

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 second disturbance in the processed video). After temporal collapsing, a given parameter *p* is either all negative or all positive, but not both. Table D.2 presents a summary of the most commonly used temporal collapsing functions.

**Table D.2 – 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. |

### D.8.5 Nonlinear scaling and clipping

The all-positive or all-negative temporally collapsed parameter *p* from D.8.4 may be scaled to account for nonlinear relationships between the parameter value and perceived quality. It is preferable to remove any nonlinear relationships before building the video quality models (clause D.9), since a linear least-squares algorithm will be used to determine the optimal parameter weights. The two nonlinear 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

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

### D.8.6 Parameter naming convention

This clause 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 D.3 summarizes the sub-names used to create the technical parameter name, listed in the order that they occur. Clause D.8.6.1 provides examples of technical parameter names and their associated sub-names from Table D.3.

**Table D.3 – Technical naming convention used for video quality parameters**

| Sub-name | Definition | Examples |
|---|---|---|
| Color | The color space image planes used by the parameter. | *Y* for luminance image plane<br>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 "Color" 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 D.7.2.2<br>*hv13_angleX.XXX_rminYY* for the $f_{HV13}$ feature in D.7.2.2, where *X.XXX* is Δθ and *YY* is the $r_{min}$<br>*coher_color* for the $f_{COHER\_COLOR}$ feature in D.7.3<br>*cont* for the $f_{CONT}$ feature in D.7.4<br>*ati* for the $f_{ATI}$ feature in D.7.5<br>*contrast_ati* for the $f_{CONTRAST\_ATI}$ feature in D.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* |
| Block Size | Present when the image is divided into S-T blocks (see D.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, color 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<br>*128×128* for blocks that include 128 frame lines vertically by 128 frame pixels horizontally |

### Table D.3 – Technical naming convention used for video quality parameters

| Sub-name | Definition | Examples |
|---|---|---|
| Block Frames | Indicates the temporal extent of the S-T blocks (see D.7.1.1), referenced to 30 frames per second (fps) video. For example, *6F* 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<br>*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); afterwards, 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<br>*std* is the standard deviation of the pixel values<br>*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<br>*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 D.8.2). Before the Comparison Function, the intermediate results contain time histories of original and processed feature images; afterwards the intermediate results contain a time history of parameter images. | *log_gain* (see D.8.2.1)<br>*ratio_loss* (see D.8.2.1)<br>*euclid* (see D.8.2.2). |
| Spatial Collapsing Function | See D.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 D.1 |
| Temporal Collapsing Function | See D.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 D.2 |

**Table D.3 – Technical naming convention used for video quality parameters**

| Sub-name | Definition | Examples |
|---|---|---|
| Nonlinear Function | See D.8.5. Examination of the parameter's values may indicate that the parameter should be scaled in a nonlinear fashion to linearly track the subjective data. The Nonlinear 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<br>*square* for the square of the temporally collapsed parameter value |
| Clipping Function | See D.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*<br>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.<br>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. |

### D.8.6.1 Example parameter names

This clause includes five example technical names, and a step-by-step description of the sub-naming procedure given in Table D.3.

*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 D.7.2.1 in preparation for extraction of the $f_{SI13}$ feature in D.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 D.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 subsampling 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. *hv13* represents filtering of the Y images with the 13×13 spatial masks in D.7.2.1 in preparation for extraction of the $f_{HV13}$ feature in D.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* represents a Δθ 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 3.0 with 3.0. Next, the $f_{HV13}$ feature in D.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 a S-T block. *below5%* specifies the spatial collapsing function. *mean* specifies the temporal collapsing function. *square* specifies the nonlinear 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 nonlinear 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 D.7.6. The first filter, *contrast*, will consider the luminance planes directly (see D.7.4). The second filter, *ati*, will consider images generated by taking differences between successive luminance planes (see D.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 D.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 15 in D.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.

## D.9 General model

This clause provides a full description of the general model VQM (denoted as $\text{VQM}_G$). 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 D.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 $\text{VQM}_G$ by 100.

The general model was designed based on BT.601-5 video that has been subjectively evaluated at a viewing distance of six picture heights. When analyzing 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.

$\mathrm{VQM}_G$ consists of a linear combination of seven parameters. Four parameters are based on features extracted from spatial gradients of the Y luminance component (see D.7.2.2), two parameters are based on features extracted from the vector formed by the two chrominance components ($C_B$, $C_R$) (see D.7.3), and one parameter is based on contrast and absolute temporal information features, both extracted from the Y luminance component (clauses D.7.4 and D.7.5, respectively). $\mathrm{VQM}_G$ is given by

$$
\begin{aligned}
\mathrm{VQM}_G = \\
&\{-0.2097 \times \mathrm{Y\_si13\_8{\times}8\_6F\_std\_12\_ratio\_loss\_below5\%\_10\%} \\
&+0.5969 \times \mathrm{Y\_hv13\_angle0.225\_rmin20\_8{\times}8\_6F\_mean\_3\_ratio\_loss\_below5\%\_mean\_square\_clip\_0.06} \\
&+0.2483 \times \mathrm{Y\_hv13\_angle0.225\_rmin20\_8{\times}8\_6F\_mean\_3\_log\_gain\_above95\%\_mean} \\
&+0.0192 \times \mathrm{color\_coher\_color\_8{\times}8\_1F\_mean\_euclid\_std\_10\%\_clip\_0.6} \\
&-2.3416 \times [\mathrm{Y\_si13\_8{\times}8\_6F\_std\_8\_log\_gain\_mean\_mean\_clip\_0.004}\,|^{0.14]} \\
&+0.0431 \times \mathrm{Y\_contrast\_ati\_4{\times}4\_6F\_std\_3\_ratio\_gain\_mean\_10\%} \\
&+0.0076 \times \mathrm{color\_coher\_color\_8{\times}8\_1F\_mean\_euclid\_above99\%tail\_std}\}\,|_{0.0}
\end{aligned}
$$

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 color 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 $\mathrm{VQM}_G > 1.0$, then $\mathrm{VQM}_G = (1 + c) \times \mathrm{VQM}_G / (c + \mathrm{VQM}_G)$, where $c = 0.5$.

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

## D.10 Informative references

[D-1] ITU-R Recommendation BT.500-11 (2002), *Methodology for subjective assessment of the quality of television pictures*.

[D-2] ITU-T Recommendation H.261 (1993), *Video codec for audiovisual services at p × 64 kbit/s.*

[D-3] ITU-T Recommendation J.143 (2000), *User requirements for objective perceptual video quality measurements in digital cable television.*

[D-4] ITU-T Recommendation P.910 (1999), *Subjective video quality assessment methods for multimedia applications.*

[D-5] ITU-T Recommendation P.931 (1998), *Multimedia communications delay, synchronization, and frame rate measurement.*

[D-6] Jain A.K., *Fundamentals of Digital Image Processing* (1989), Englewood Cliffs, NJ: Prentice-Hall Inc., pp. 348-357.

[D-7] 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.

[D-8] 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.

[D-9] SMPTE Recommended Practice 187 – 1995, *Center, Aspect Ratio, and Blanking of Video Images*, Society of Motion Picture and Television Engineers, 595 West Hartsdale Avenue, White Plains, NY 10607.

[D-10] Wolf S. and Pinson M., *In-service performance metrics for MPEG-2 video systems*, in Proc. Made to Measure 98 – Measurement Techniques of the Digital Age Technical Seminar (1998), technical conference jointly sponsored by the International Academy of Broadcasting (IAB), the ITU, and the Technical University of Braunschweig (TUB), Montreux, Switzerland, Nov. 12-13.

[D-11] Wolf S. and Pinson M., *Spatial-temporal distortion metrics for in-service quality monitoring of any digital video system* (1999), in Proc. SPIE International Symposium on Voice, Video, and Data Communications, Boston, MA, Sep. 1999.

[D-12] Wolf S. and Pinson M., *The relationship between performance and spatial-temporal region size for reduced-reference, in-service video quality monitoring systems* (2001), in Proc. SCI/ISAS 2001 (Systematics, Cybernetics, and Informatics/Information Systems Analysis and Synthesis), Jul. 2001, pp. 323-328.

[D-13] Wolf S. and Pinson M., *Video Quality Measurement Techniques* (2002), NTIA Report 02-392, Jun. 2002.

[D-14] Pinson M. and Wolf S., *Video Quality Measurement User's Manual* (2002), NTIA Handbook 02-1, Feb. 2002.

[D-15] ITU-T Tutorial (2004), *Objective perceptual assessment of video quality: Full reference television.*

## D.11 Video Quality Metric (VQM) raw objective data

This clause provides a full disclosure of the NTIA VQM raw objective data from the VQEG Phase 2 Full Reference Television (FR-TV) tests.

### Raw data summary

This General Model developed by the National Telecommunications and Information Administration (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 Double Stimulus Continuous Quality Scale (DSCQS). For the raw data presented in this annex, 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 seconds 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 clause D.6). In addition, the spatial region of interest (SROI) used to calculate the VQM value for each clip was chosen as follows:

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

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 processed valid region (abbreviated PVR, calculated automatically as given in D.6.2) 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).

3) The SROI is always centered horizontally such that the left hand sample starts at a BT.601-5 luminance/chrominance co-located sampling point. The SROI is centered 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 centering of the SROI within the video frame may not be possible.

### 525-line Raw Objective Data

| Source # | HRC # | NTIA: Proponent H |
|---|---|---|
| 1 | 1 | 0.660 (Note) |
| 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 |

NOTE – 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 D.6.1.2). For the other scenes of HRC 1, spatial/temporal registration was correctly estimated. Clause D.6.1.5.7 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.

## 625-line Raw Objective Data

| Source # | HRC # | 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 I

# KDDI

# Objective video quality assessment scheme and performance evaluation

## I.1 Scope

Recently, digital television broadcasting and transmission services are beginning to come into practical use. These services use video codecs (video signal encoding devices) based on MPEG-2, an international standard method for compression of digital video signals. Video codecs are comprised of encoders, which perform the compression, and decoders, which reconstruct the compressed video data. These devices work by removing redundant information from the enormous volume of information contained in video signals. This makes it possible to transmit the information efficiently using only a limited amount of bandwidth.

There is always some amount of degradation in the quality of video that has been compressed and transmitted using a video codec. The amount of degradation depends on the contents of the picture. Generally, there is more distortion in fast-moving scenes, like those in a sports broadcast. There are also variations in the quality of the output produced by different codecs. MPEG-2 is an international standard, but the quality of specific types of compressed video still depends, to a certain extent, on the manufacturer's implementation.

For its television transmission, especially in contribution and primary and secondary distribution, it is required to strive to achieve consistently high quality by constantly monitoring the quality of the transmitted pictures.

In conventional analogue FM transmission, there is little degradation in the picture due to the contents or to analogue modulation, so quality is stable. But in the transmission of compressed digital video, the quality of the picture varies as described above according to the nature of the contents and the codec employed, and checking the quality of this kind of video is expected to be a very complex operation.

Hence, it is considered necessary to standardize a scheme to evaluate the picture quality of MPEG-2-based video codecs mainly used in contribution and primary and secondary distribution. In these classes, the following functions are considered to be necessary:

- Generic assessment for various types of video contents (Analogue/Digital/Composite/Component video formats) are supported;
- Real-time assessment;
- Precise temporal and spatial alignment between an original and a codec out signal;
- Sensitive and accurate assessment to subtle and complex distortions.

Considering the above, this appendix describes an effective evaluation scheme and its implementation based on the characteristics of human visual perception, enabling very precise measurements of video quality.

## I.2 Objective video quality assessment scheme

Figure I.1 shows the three-layered picture quality assessment model as seen by the human eye. Generally, the human eye cannot watch a whole frame at a glance, but only a local spot area in a frame, which is around the gaze point of the human eye, and recognizes the texture and also quality of the area depending on the degrees and characteristics of noise mixed in this texture. The whole frame is understood by moving the gaze point among objects, which are picture components of the frame, and picture quality assessment is also conducted for the whole frame at the same time. In this process, picture quality is determined by the noise over a frame. Therefore, to perform objective measurement of subjective picture quality, the macro-to-micro three-layered picture structures (object, texture and noise layers) are used, and a bottom-up noise-weighting scheme is proposed which uses a particular weighting function at each layer taking into account human visual perception (Figure I.2).

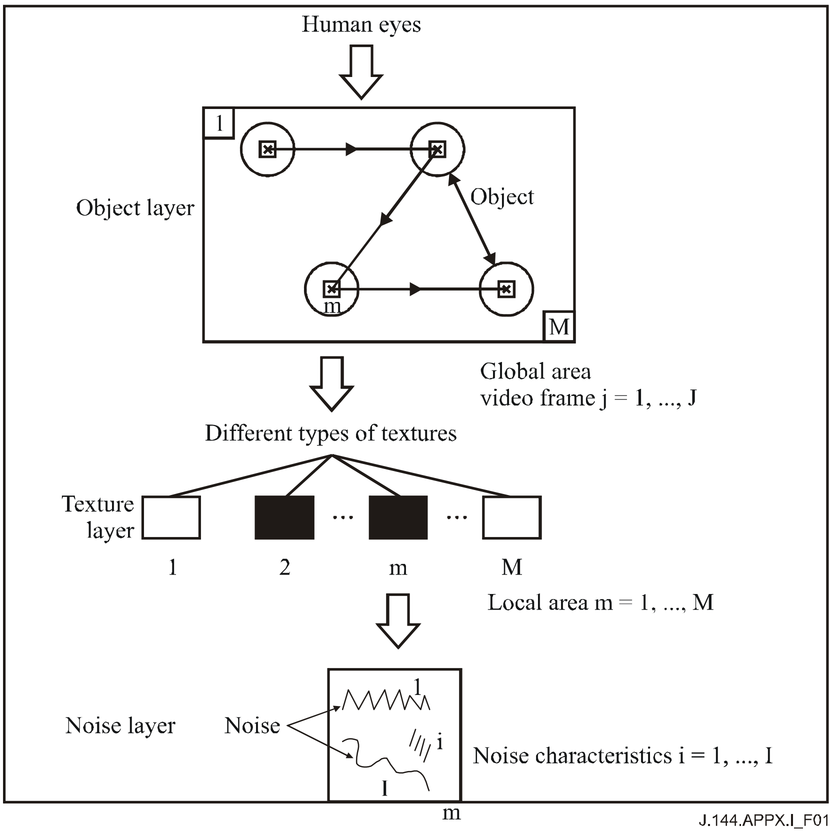


**Figure I.1 – Three-layered model for video signal**

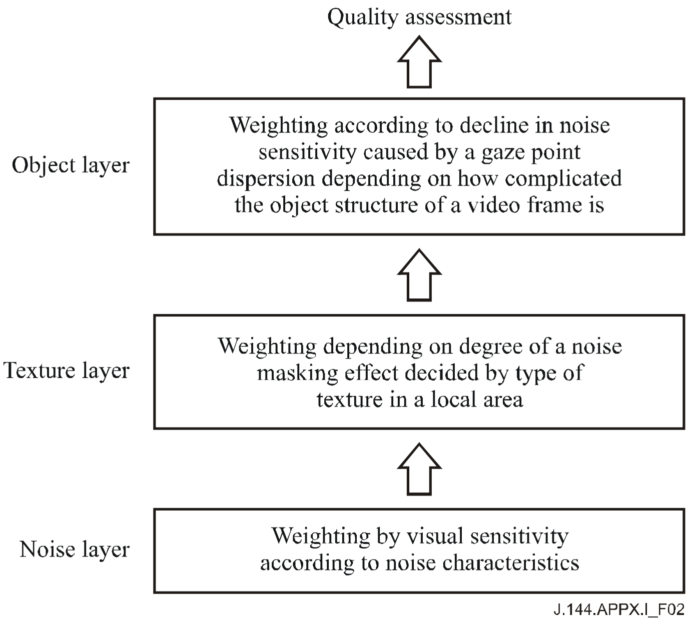

Quality assessment
Object layer
Weighting according to decline in noise sensitivity caused by a gaze point dispersion depending on how complicated the object structure of a video frame is
Texture layer
Weighting depending on degree of a noise masking effect decided by type of texture in a local area
Noise layer
Weighting by visual sensitivity according to noise characteristics
J.144.APPX.I_F02

**Figure I.2 – Three-layered bottom-up noise weighting**

Firstly, at the noise layer, common noise in a video compression process such as high frequency noise, low frequency noise, chroma noise, jerkiness, flicker and so on are weighted depending on their degrees and characteristics. For this weighting, it is useful to perform a frequency conversion to classify these noises. Secondly, at the texture layer, local spot areas are classified into several groups by their texture types. These groups include, for example, "detail texture" such as a forest, trees and a stadium in which noise is strongly masked, and "flat texture" such as a human skin and a sky in which noise is easily recognized. Consequently, noises are weighted more or less according to their texture types. Finally, at the object layer, the dispersion degree of the gaze point is predicted by measuring how complicated the structure is of objects in the video frame. Then, noises in the whole frame are weighted corresponding to a decline in noise sensitivity caused by this dispersion.

To obtain mathematical expressions for these weighting processes, we make the following definitions:

*P(j,m,i)*: Power of a noise *i* in a local area *m* of a frame *j*;

*hi*: Weighting function for a noise *i*;

*C(j,m)*: Texture in a local area (*j*,*m*);

*tc*: Noise weighting function in a texture *C*;

*G(j)*: Parameter indicating how complicated the structure is of objects of a frame *j*;

9 *G*: Noise weighting function depending on dispersion degree of a gaze point.

Following these definitions, noises are summed up in order from the low layer to the high layer.

In the noise layer, by summing up noise which is weighted by *hi* corresponding to noise characteristics in a local area (*j*,*m*), we calculate $WMSE_{NL}$ as:

$$WMSE_{NL}(j,m) = \frac{1}{I}\sum_{i=1}^{I} hi \cdot P(j,m,i) \qquad \text{(I-1)}$$

Next, at the texture layer, by summing up $WMSE_{NL}(j,m)$ over the whole frame (*m*=1, ...,M) being weighted by *tc* corresponding to a texture *C*(*j*,*m*) in a local area (*j*,*m*), we calculate $WMSE_{TL}(j)$ as:

$$WMSE_{TL}(j) = \frac{1}{M}\sum_{m=1}^{M} t_c(j,m)\cdot WMSE_{NL}(j,m) \qquad \text{(I-2)}$$

Finally, at the object layer, by taking an average value of *WMSE<sub>TL</sub>* over frames *j*=1, ...,*J* being weighted by *G*(*j*) corresponding to the dispersion degree of the gaze point, we calculate *WMSE<sub>OL</sub>* as:

$$WMSE_{OL} = \frac{1}{J}\sum_{j=1}^{J} q_G(j)\cdot WMSE_{TL}(j) \qquad \text{(I-3)}$$

We further convert this *WMSE<sub>OL</sub>* to *WSNR* and calculate the *DSCQS* (Double-stimulus continuous quality-scale method) (0-100%) defined in ITU-R Rec. BT.500-11 as:

$$WSNR(dB) = 10\log_{10}\frac{255^2}{WMSE} \qquad \text{(I-4)}$$

$$D(\%) = f(WSNR) \qquad \text{(I-5)}$$

**Power of a local area noise P(*j*,*m*,*i*)**

At first, local area m is defined as $m_w \times m_h$ square block.

Assume that the noise characteristics I=1 is frequency domain weighted noise.

$$P(j,m,i) = \sum_{q=1}^{m_h}\sum_{p=1}^{m_w}\{X(p,q) - Y(p,q)\}^2$$

where, *X*, *Y* are transformed coefficient values of original picture and coded picture respectively.

## I.3 Implementation

The system is made up of two parts: a synchronization module, which enables precise comparison between the reconstructed video and the original video, and a calculation module, which determines video quality with reference to characteristics of human visual perception. Figure I.3 shows the configuration of the system, and Table I.2 describes principal parameters. As Table I.2 shows, both composite (NTSC)/component signals with full samplings are supported.

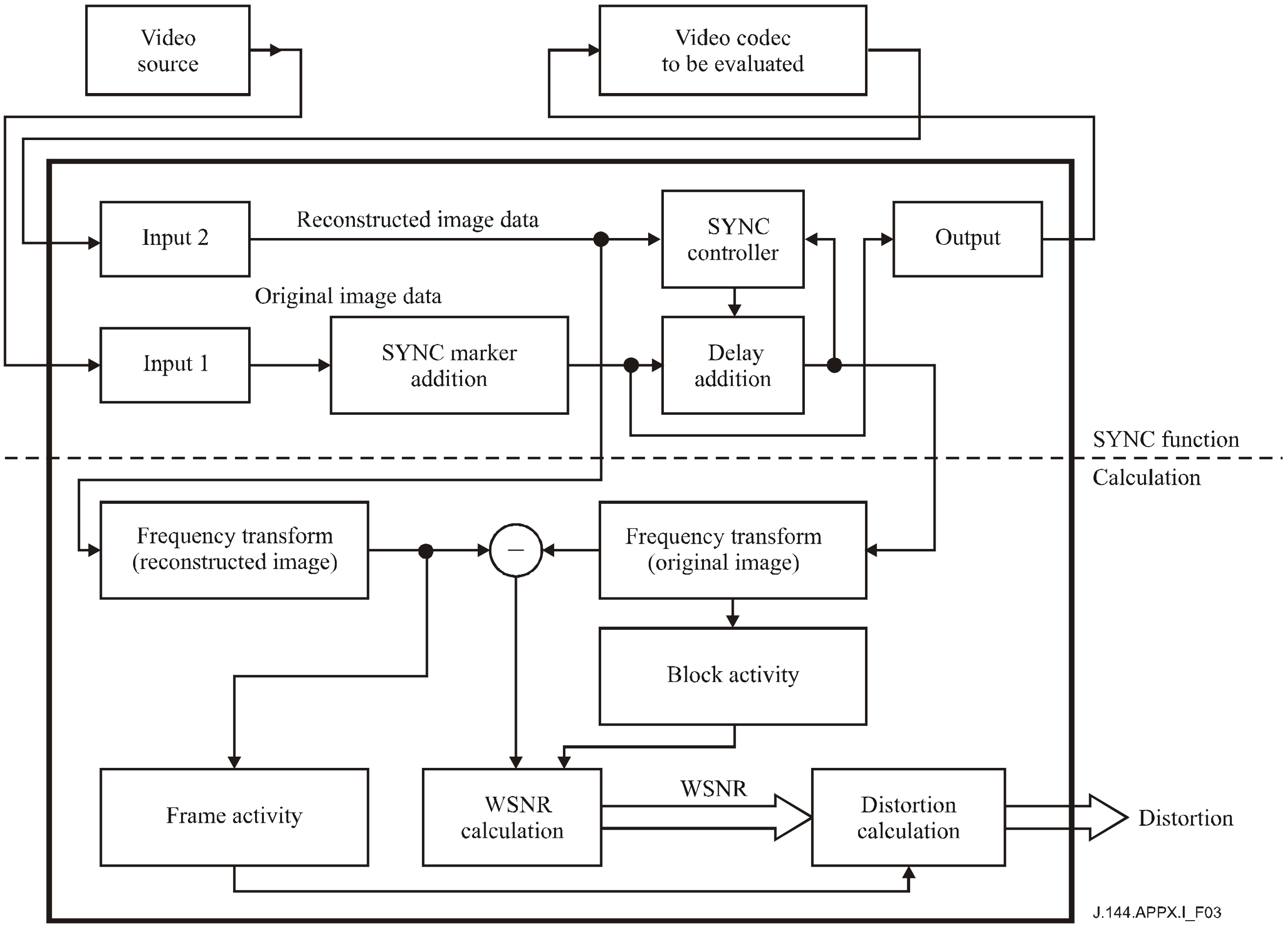


**Figure I.3 – System configuration**

### I.3.1 Synchronization module

A real-time video quality evaluation system requires the synchronization module, which is independent from the calculation module. Note that it is not necessary for the off-line calculation, such as video file comparison by software quality evaluation. The following describes one of the synchronization methods as an example.

Television signals from the original video source are read into the system through input module 1 and marked with a synchronization marker that varies with each frame. The marker, for example, is like a sine wave form whose frequency is modulated by the frame number. Then the frames with markers are sent to the delay module, where they are stored in memory. At the same time, the frames are sent via the output module to the video codec that is to be evaluated. The video codec compresses the frames, which are read into the system again through input module 2 and compared with the marked frames stored in the delay of the video codec being evaluated. Finally, the synchronization module performs temporal (frame delay) and spatial (line and pixel shift) alignment precisely so that the amount of quality degradation described below will be as close as possible to subjective assessment by human viewers.

These operations provide the synchronization needed for the evaluation and the markers used in these operations are designed so as to work well even through the severely signal-distorted process such as high compression, Y/C separation and filtering in a video codec.

### I.3.2 Calculation module

Unlike human vision, calculation of the quality of the picture takes a bottom-up approach, building up the whole from the various parts. Firstly, in order to evaluate the effect of variations in sensitivity due to the spatial frequencies of noise, a difference value (noise) is obtained for the frequency components of the original picture and the reconstructed image. This value is input into the WSNR (Weighted Signal-to-Noise Ratio) module, which assigns different sensitivity weights for each frequency region. At the same time, it obtains a value (the block activity) that indicates whether each block in the picture is flat or busy. The noise masking effect is also applied to obtain an overall WSNR.

Finally, a value to indicate the size of the objects making up the picture is obtained (the frame activity). This enables the system to estimate the degree to which sensitivity to noise decreases due to dispersion of the amount of degradation in quality and is obtained by applying the decrease in sensitivity to noise to the WSNR.

**Table I.1 – Principal parameters**

| | |
|---|---|
| Applicable video signal format | NTSC composite signal<br>525/60 component signal<br>D1 serial digital |
| Sampling frequency (Analogue input) | 14.318 MHz (NTSC)<br>13.5 MHz (Component Y)<br>6.75 MHz (Component C) |
| Applicable codec | MPEG-1,2-based codec<br>Composite codec, etc. |
| Effective evaluation area | 768 pixels~480 lines (NTSC)<br>720 pixels~480 lines (Component Y)<br>360 pixels~480 lines (Component C) |
| Signal analysis | Hadamard transform (NTSC)<br>Discrete cosine transform (Component)<br>Alternative: Fourier transform |
| Noise Weighting | Spatial frequency visual sensitivity<br>Noise masking effect<br>Gaze point scattering |
| Evaluation result | Picture quality assessment (Distortion, %)<br>WSNR (dB)<br>SNR (dB) |
| Control signal interface | RS-232C |

### I.4 Verification results

We compared the evaluation results of the proposed scheme with the subjective assessment test results that have already been graded following ITU-R Rec. BT.500-11. Assessment targets are MPEG-2 SP@ML with 5 Mbit/s, 7 Mbit/s and 10 Mbit/s applied for ITU-R Rec. BT.601-5, 4:2:2 component TV test signals. These are 17 data including Mobile, Flower garden, and Cheerleaders etc. Therefore, we have in total 17 data × 3 bit rate = 51 samples (Table I.2).

For these samples, we conducted the subjective assessment test on two different days (23rd and 24th March 1995) with the same conditions and viewers. The "triangle" of the objective assessment and two days subjective assessment results is shown in Figure I.4.

**Table I.2 – Test data list**

| | |
|---|---|
| 1 | Susie |
| 2 | Popple |
| 3 | Table tennis |
| 4 | Mobile & Calendar |
| 5 | Autumn leaves |
| 6 | Football |
| 7 | Tempest |
| 8 | Cheer leaders |
| 9 | Credits |
| 10 | Cruising |
| 11 | Bicycle |
| 12 | Horse riding |
| 13 | Summer flowers |
| 14 | Ferris wheel |
| 15 | Flower garden |
| 16 | Kiel Harbor 4 |
| 17 | Balls of wool |

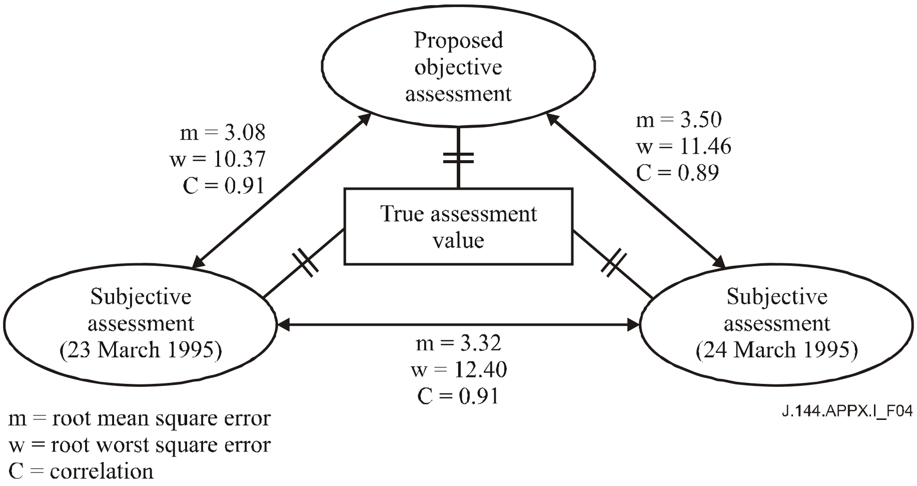

Proposed objective assessment
m = 3.08
w = 10.37
C = 0.91
m = 3.50
w = 11.46
C = 0.89
True assessment value
Subjective assessment (23 March 1995)
Subjective assessment (24 March 1995)
m = 3.32
w = 12.40
C = 0.91
m = root mean square error
w = root worst square error
C = correlation
J.144.APPX.I_F04

**Figure I.4 – Comparisons with subjective assessment tests**

Figure I.4 proves that assessment accuracy expressed by RMSE, RWSE and correlation of three assessment results are nearly equal from the triangle centre, which is the true assessment value. In addition, Figure I.5 shows distributions of 51 samples among one objective and two subjective assessments. Samples in three graphs are randomly distributed but the subtle differences can be seen in each distribution. In the distribution of the 23rd and 24th subjective comparisons it is uniformly random, but inequality in distributions can be seen in subjective and objective assessment comparisons depending on score range. That is, both graphs of 23rd and 24th vs. objective scheme give sample plots with higher correlation at 20%-40% but less correlation at 10%-20%. Further study will be needed to eliminate this.

By this fact, it is concluded that it would be feasible to use the proposed scheme in addition to ITU-R Rec. BT.500-11.

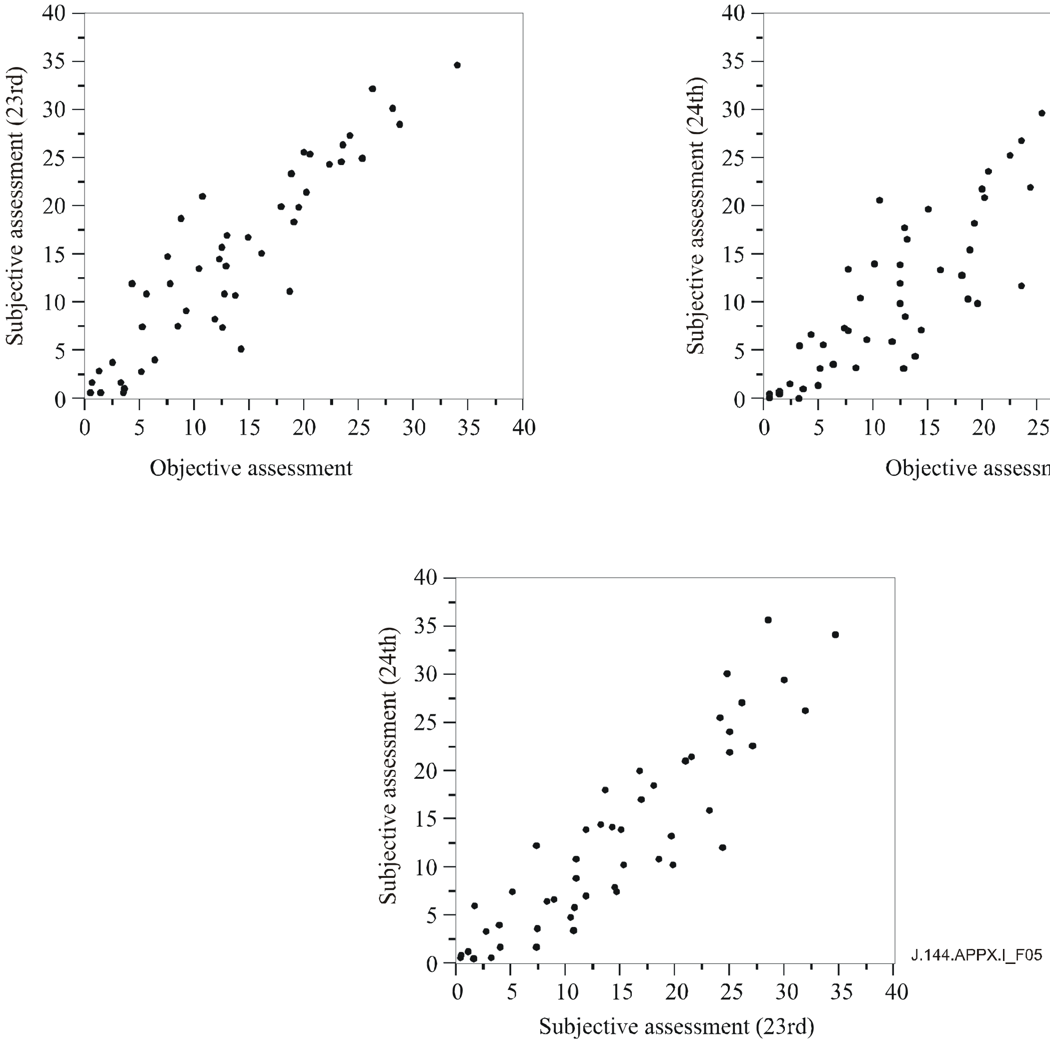


**Figure I.5 – Comparisons between an objective and two subjective assessments**

# Appendix II

## Tektronix Inc. and Sarnoff Corporation

## Objective perceptual video quality measurement using a JND-based full reference technique

### II.1 Scope, purpose, and application

### II.1.1 Scope

This appendix specifies an objective JND-based video quality measurement method utilizing availability of the full reference video signal. It is a double-ended measurement identified as the Picture Quality Rating (PQR) method as shown in Figure II.1.

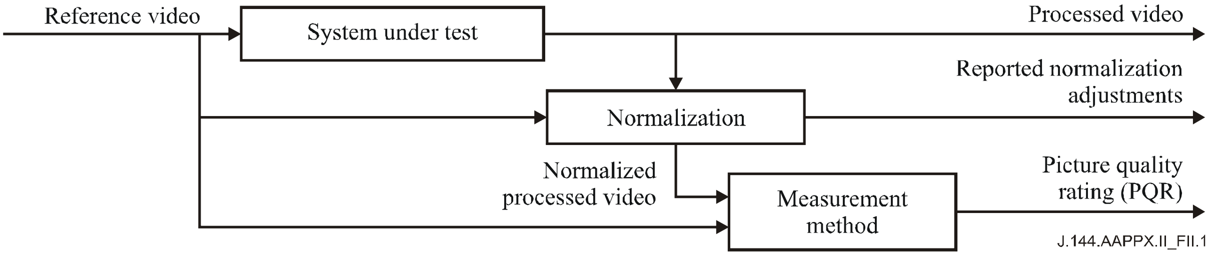


**Figure II.1 – System block diagram**

The PQR method specified in this appendix is based on processing 8-bit digital component video as defined by ITU-R Rec. BT.601-5 in a manner representative of the response of the human visual system. Due to the perceptual nature of the measurement, various compression methods can be accommodated (MPEG, NTSC, PAL, etc.). In addition, the transmission system may include a concatenation of compression methods or be a simple pass-through for evaluation of a codec (encoder/decoder combination). Results of the PQR method are stated in picture quality rating (PQR) values.

Normalization of the processed video is required for application of the PQR method. This appendix specifies only the PQR method algorithm and the normalization accuracy. See II.3.1 for normalization requirements.

### II.1.2 Purpose

This appendix provides the technical description of an objective perceptual video quality measurement method that is currently in use. While improved methods may be developed in the future, this appendix provides a video measurement method necessary to support the interconnection and interoperability of telecommunication networks at interfaces with end-user systems, carriers, information and enhanced-service providers, and customer premises equipment.

### II.1.3 Application and limitations

#### II.1.3.1 PQR method applications

Application of any full reference method provides some operational restrictions. However the PQR method specified in this appendix is not limited to laboratory evaluations. Some specific applications appropriate for use with the PQR method are:

- Codec evaluation, specification, acceptance testing;
- Real-time, in-service transmission monitoring at the source;

- Remote transmission evaluation with a copy of the reference available.

In using the PQR method, the accuracy limitations specified in II.1.3.4 must be carefully considered.

### II.1.3.2 Limitations

Based on validation described the VQEG Phase I final report (see ITU-T Tutorial) the PQR method specified in this appendix is appropriate for short video sequences (2 to 10 seconds in duration) at a viewing distance of 5H (5 picture heights at 480 lines per picture height). PQR values will be useful for shorter and longer viewing distances where it is recognized that human perception of picture degradation will lessen as the viewing distance is increased, whereas the PQR values will remain constant. Although the algorithm can be modified to reflect human perception at other viewing distances, such modifications are not part of this appendix.

A detailed list of test factors, coding technologies and applications relating to PQR method accuracy is shown in Appendix II – Attachment 2 based on the VQEG data selected.

While the normalization method specified in II.3.1 may be used to detect changes in picture size (such as produced by a special effects unit), it has not been shown to provide the information necessary to determine the amount of size change. The PQR method is not suitable for evaluation of pictures that are not the original size of the picture input to the system under test nor that have vertical shifts other than an integer number of lines.

Video classes are defined in Annex B/P.911. This appendix is intended to provide measurements for classes TV1, TV2 and TV3 as quoted below. These classes are differentiated from the multimedia classes in that encoders always provide constant frame rate and constant latency operation. While the compression system may reduce the number of pixels (usually only in the horizontal direction) as part of the encoding process, the resulting output of the decoder will be full resolution component video as per ITU-R Rec. BT.601-5.

- TV 0 – Loss-less: ITU-R Rec. BT.601-5, 8 bits per sample, video used for applications without compression.
- TV 1 – Used for complete post-production, many edits and processing layers, intra-plant transmission. Also used for remote site to plant transmission. Perceptually transparent when compared to TV 0.
- TV 2 – Used for simple modifications, few edits, character/logo overlays, program insertion, and inter-facility transmission. A broadcast example would be network-to-affiliate transmission. Other examples are a cable system regional downlink to a local head-end and a high quality video conferencing system. Nearly perceptually transparent when compared to TV 0.
- TV 3 – Used for delivery to home/consumer (no changes). Other examples are a cable system from the local head-end to a home and medium-to-high quality video conferencing. Low artifacts are present when compared to TV 2.

All of these classes have constant, but not necessarily low, one-way latency and constant delay variation. The PQR method specified in this appendix is not appropriate for video conferencing applications that repeat fields, or do not meet the latency and delay requirements of the video classes. In addition, the PQR method is only applicable to typical broadcast transmission systems with very low error rates such as those included in the VQEG tests.

### II.1.3.3 Comparison with subjective assessment

While objective measurements with good correlation to subjective quality assessment are desirable in order to attain optimal quality of service, it must be realized that objective measurements are not a direct replacement for subjective quality assessment. Subjective quality assessments are carefully designed procedures intended to determine the average opinion of human viewers to a specific set of video sequences for a given application. Results of such tests are valuable in basic system design and benchmark evaluations. Subjective quality assessments for a different application with different test conditions will still provide meaningful results. However, opinion scores for the same set of video sequences are likely to have different values. Objective measurements are intended for use in a broad set of applications producing the same results with a given set of video sequences. The choice of video sequences to use, and the interpretation of the resulting objective measurements, are some of the factors varied for a specific application. Therefore, objective measurements and subjective quality assessment are complementary rather than interchangeable. Where subjective assessment is appropriate for research related purposes, objective measurements are required for equipment specifications and day-to-day system performance measurement and monitoring.

### II.1.3.4 Accuracy and cross-calibration

For PQR values as given by equation II-36, the VQM accuracy and cross-calibration methods detailed in ITU-T Rec. J.149 were applied to the VQEG 525-line subjective data with HRCs 15 and 16 removed (see reference ITU-T COM 9-80, June 2000 – in Appendix II – Attachment 1). These data were selected so as to be representative of American broadcast television. In evaluating a VQM, the objective data is curve fitted to a normalized scale (0-1) to provide the best correlation between subjective and objective data. For the accuracy calculation, this transformation is also important as it provides the best consistency of statistical confidence of resolving power (accuracy) over the range of objective values.

The curve fitting function named Logistic II in 4.2/J.149 is used and is constrained to intersect or asymptotically approach zero for increasingly perfect pictures ($a = -e^{-cd}$).

$$VQM = a + \frac{b - a}{1 + e^{-c(PQR - d)}}$$

The logistic function is also constrained to have a maximum value of 1 (b=1) in the normalized 0-1 scale or 100 in the native DSCQS scale (see ITU-R Rec. BT.500-11 used for subjective testing of full reference VQMs. Once transformed to the Common Scale, the PQR method can be cross-calibrated to any other VQM. With these constraints, the equation for the logistic function for the PQR method is shown below.

$$VQM = \frac{1 - e^{-c \times PQR}}{1 + e^{c(d - PQR)}}$$

Where:

c = 0.5031

d = 9.634

Figure II.2 shows the plot of selected VQEG data and constrained logistic function for the native PQR values. Figure II.3 shows the conversion from native PQR values to the common scale using the constrained logistic function. Although the constrained logistic function curve can be calculated to extend beyond the VQEG data points to limits appropriate for the PQR value calculations and subjective data DSCQS scores, it is only considered to be valid for accuracy analysis in the range of native PQR values from 3.5 to 10.5.

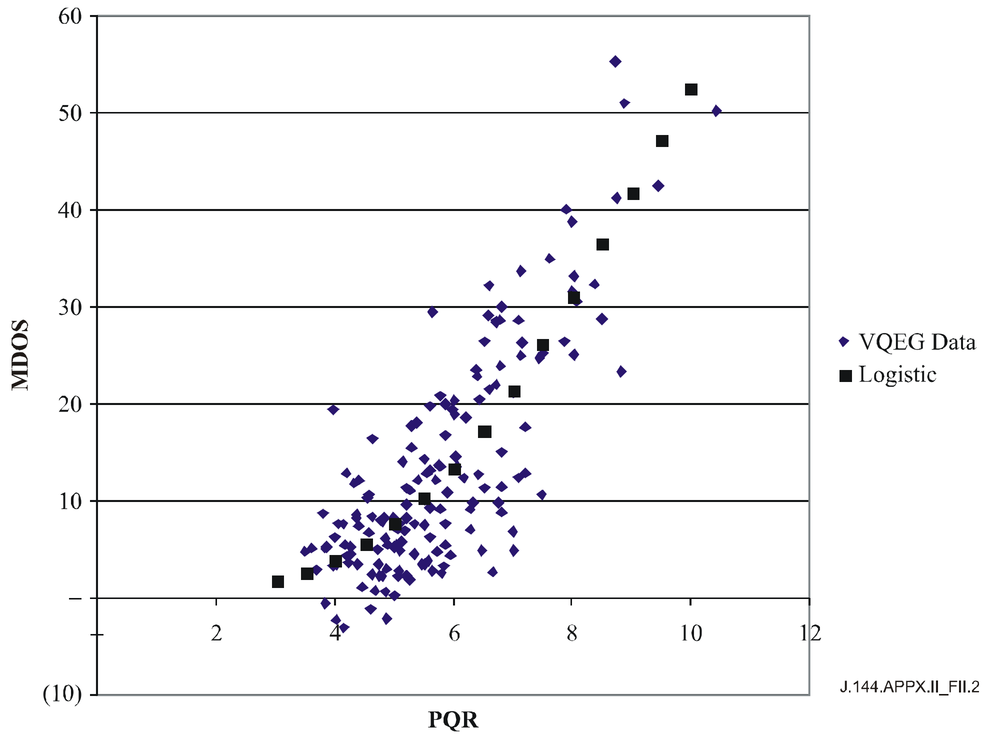


**Figure II.2 – VQEG data and constrained logistic function**

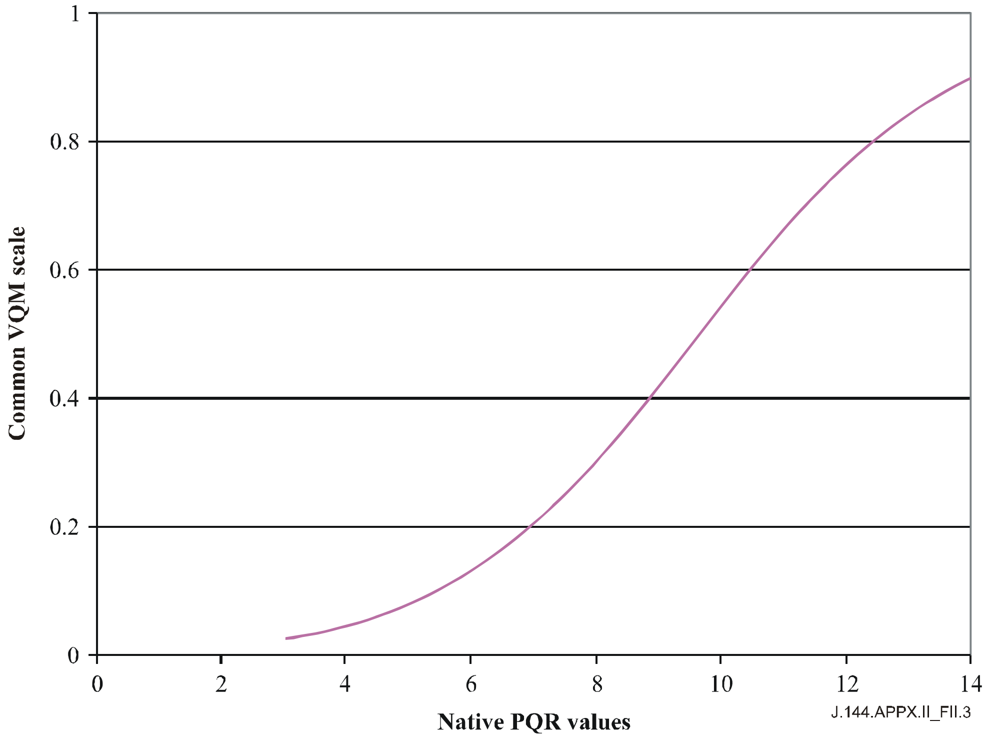


**Figure II.3 – PQR value conversion from native to common scale**

Resolving power is the difference in VQM values for two measurements such that there is a known confidence level that the measurement with the better VQM value also has the better subjective score. Two methods are provided in ITU-T Rec. J.149 to calculate the resolving power of a VQM. A sophisticated statistical method called the z-test takes into account that resolving power is not a simple function of required confidence but varies based on the subjective/objective data points available. Figure II.4 shows the results of that calculation for the PQR method. The delta-VQM values are stated in the normalized domain where the resolving power is nominally a constant for all VQM values.

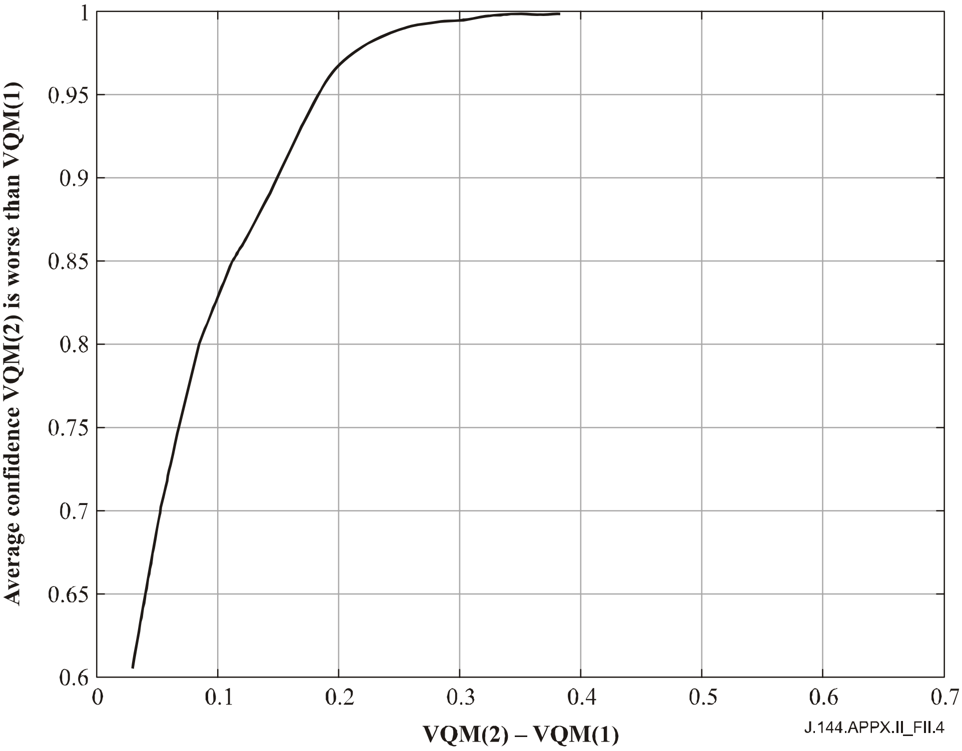


**Figure II.4 – Resolving power versus normalized VQM scale**

The graph shows that for a difference in normalized PQR values of 0.1 there is a confidence level of 0.81and that the sequence with the lowest PQR value (higher quality) will have the lowest DSCQS subjective score (higher quality). Choice of what resolving power to use for various applications is aided by a classification of error analysis as shown in Annex C.

A first order calculation of resolving power can be made by simply calculating the root mean squared error (RMSE) of the subjective scores versus the objective values in the normalized domain. Differences in VQM values equal to the RMSE provide a 68% confidence level and 1.96 times the RMSE provides a 95% confidence level. While this method does not give the same result as the more complex approach, it is easily understood and may be quite useful considering the accuracy levels in operational environments.

VQM_RMSE = 0.06723

This corresponds approximately to the more accurate curve of Figure II.4 as shown below.

| Confidence level | Figure II.4 | RMSE |
|---|---|---|
| 68% | 0.053 | 0.066 |
| 95% | 0.187 | 0.132 |

Figure II.4 can be converted to the PQR scale if so desired by scaling the x-axis (i.e., VQM (2) – VQM (1)) using the derivative of the curve fitting function Logistic II as shown in 4.2/J.149.

$$[PQR(2)-PQR(1)]=[VQM(2)-VQM(1)]\frac{\left(1+e^{-c(PQR-d)}\right)^2}{c(b-a)e^{-c(PQR-d)}}$$

Applying the constraints a = $-e^{-cd}$ and b = 0 the equation becomes:

$$[PQR(2)-PQR(1)]=[VQM(2)-VQM(1)]\frac{\left(1+e^{c(d-PQR)}\right)^2}{c\left(e^{c(d-PQR)}+e^{-c(PQR)}\right)}$$

Where:

c = 0.5031

d = 9.634

This produces a family of curves since the x-axis scaling factor depends on the PQR value.

## II.2 References

The following standards contain provisions, which through reference in this text constitute provisions of this appendix. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this Appendix are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

- ITU-R Recommendation BT.601-5 (1995), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*.
- ITU-T Recommendation J.149 (2004), *Method for specifying accuracy and cross-calibration of Video Quality Metrics (VQM).*
- ITU-T Recommendation P.911 (1998), *Subjective audiovisual quality assessment methods for multimedia applications*.

## II.3 Introduction

### II.3.1 Normalization

Normalization means that time-invariant systematic changes in the video from reference input to processed video output are removed prior to performing the human vision model based measurement (see Figure II.1). The PQR method is based on human vision model filters that compare reference and processed pictures on what is effectively a pixel-by-pixel basis. The normalization method specified in T1.TR.73-2001*, Annex B (see Appendix II – Attachment 1) is appropriate for use with the PQR method.

Parameters to be adjusted by the normalization process are horizontal and vertical picture shifts; luminance and color gain changes; luminance and color DC level changes; and component or luminance to color channel-to-channel delay offset. Because these changes could produce changes in perceived picture quality, they shall be reported as part of test results. It is necessary to separate these changes from the PQR calculation for two reasons. The main reason is to provide the most accurate PQR value. Secondly, such normalization corresponds closely with typical system operation for the gain and DC level parameters where appropriate adjustments are generally available and routinely made. Small values of picture shift, horizontally or vertically, are generally

* T1 standards are maintained since November 2003 by ATIS.

not considered to change perceived picture quality. However, their presence is a picture error and will produce significant problems in multi-generation applications. Temporal alignment must be perfect so each processed field/frame is compared with the equivalent reference.

Processed video is normalized on a field-by-field basis by measurement of calibrated test signals embedded in the reference sequence. Only time-invariant static changes in the video are removed, dynamic changes, due to the compression and decompression processes, are measured as part of the PQR calculation. Normalization of the processed video prior to PQR calculations shall meet the tolerances shown in Table II.1. PQR values based on normalization not meeting the tolerances of Table II.1 will have less accuracy than those specified in II.1.3.4.

**Table II.1 – Normalization parameters and tolerance**

| Parameter | Normalization tolerance |
|---|---|
| Luminance level | < 0.2 dB of peak white |
| Color-difference level | < 0.2 dB of max allowed excursion |
| Luminance DC level | < 0.5% of peak white |
| Color-difference DC level | < 0.5% of max allowed excursion |
| Channel-to-channel delay offset | < 2 ns |
| Horizontal pixel shift | < 0.1 pixel |
| Vertical line shift | 0 lines (limited to integer line shifts) |
| Temporal shift | 0 fields |

### II.3.2 PQR measurement method overview

The PQR method predicts the perceptual ratings that human subjects will assign to a degraded color-image sequence relative to its non-degraded counterpart. The model takes in two image sequences and produces several different estimates, including a single metric of perceptual differences between the sequences. These differences are quantified in units of the modelled human just-noticeable difference (JND).

An input video sequence passes through two different channels on the way to a human observer (not shown in the figure). One channel is uncorrupted (the reference channel), and the other distorts the image in some way (the channel under test). The distortion, a side effect of some measure taken for economy (such as compression), can occur at an encoder prior to transmission, in the transmission channel itself, or in the decoding process. In Figure II.8, the box called "system under test" refers schematically to any of these alternatives. The PQR method is implemented by replacing the display and observer by the Human Vision Model, which compares the test and reference sequences to produce a sequence of JND maps instead of the subjective assessment.

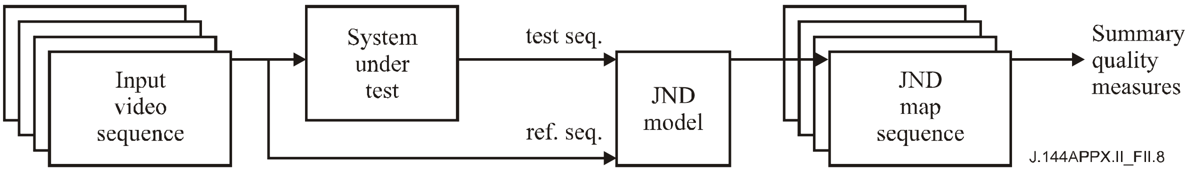


**Figure II.8 – Human vision model in system evaluation**[17]

[17] Figures II.5 through II.7 are intentionally not used.

Figure II.9 shows an overview of the algorithm. The inputs are two image sequences of arbitrary length. For each field of each input sequence, there are three data sets, labelled Y', $C_b$', and $C_r$' at the top of Figure II.9 derived, e.g., from a D1 tape. Y, $C_b$, $C_r$ data are then transformed to R', G', and B' electron-gun voltages that give rise to the displayed pixel values. In the model, R', G', B' voltages undergo further processing to transform them to a luminance and two chromatic images that are passed to subsequent stages.

The purpose of the front-end processing is to transform video input signals to light outputs, and then to transform these light outputs to psychophysically defined quantities that separately characterize luma and chroma.

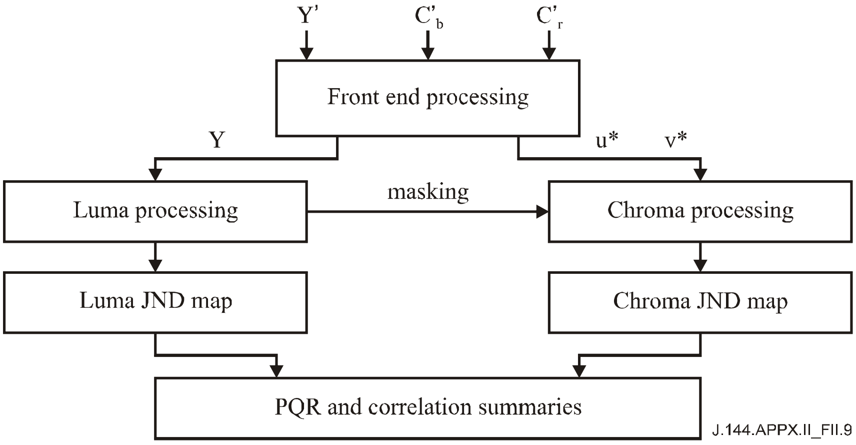


**Figure II.9 – Human vision model flow chart**

A luma-processing stage accepts two images (test and reference) of luminance Y, expressed as fractions of the maximum luminance of the display. From these inputs, the luma-processing stage generates a luma JND map. This map is an image whose gray levels are proportional to the number of JNDs between the test and reference image at the corresponding pixel location.

Similar processing, based on the CIE L*u*v* uniform-color space, occurs for each of the chroma images u* and v*. Outputs of u* and v* processing are combined to produce the chroma JND map. Both chroma and luma processing are influenced by inputs from the luma channel called masking, which render perceived differences more or less visible depending on the structure of the luma images.

Luma, chroma and combined luma-chroma JND maps are each available as output, together with a small number of summary measures derived from these maps. Single PQR value summaries model an observer's overall rating of distortions in a test sequence. JND maps give a more detailed view of the location and severity of artifacts.

## II.4 Algorithm overview

### II.4.1 Front-end processing

The stack of four fields labelled Y', $C_b$', $C_r$' at the top of Figure II.10 indicates a set of four consecutive fields from either a test or reference image sequence. The first stage of processing transforms Y', $C_b$', $C_r$' data, to R', G', B' gun voltages (see II.5.1.1).

The second stage of processing, applied to each R', G', B' image, is a point-non-linearity. This stage models the transfer from R', G', B' gun voltages to model-intensities (R, G, B) of the display (fractions of maximum luminance). The non-linearity also performs clipping at low luminance in each plane by the display.

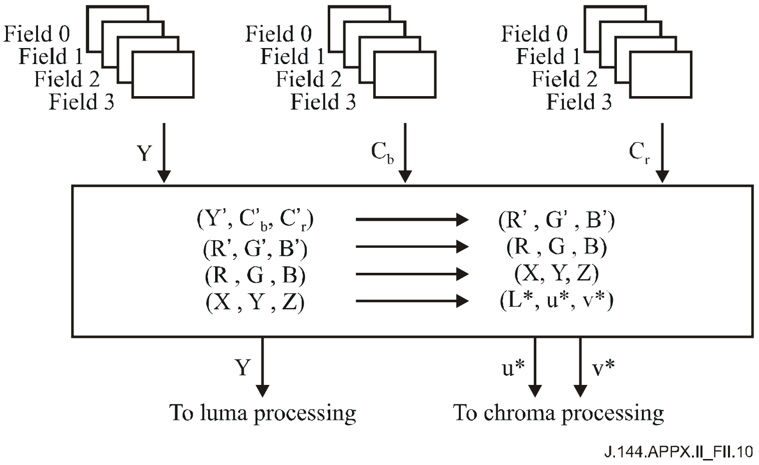


**Figure II.10 – First stage processing overview**

Following the non-linearity, one of two processing options is available: half-height and full-height. For interlaced scans, half-height images[18] are processed as given, without blank inter-lines. Full-height modeling is available for progressive scans (in which a field contains one frame, i.e., a single image rather than two interlaced fields).

Then, the vector (R,G,B) at each pixel in the field is subjected to a linear transformation (which depends on the display phosphors) to CIE 1931 tri-stimulus coordinates (X, Y, Z). The luminance component Y of this vector is passed to luma processing.

To ensure (at each pixel) approximate perceptual uniformity of the color space to isoluminant color differences, the individual pixels are mapped into CIELUV, an international-standard uniform-color space. The chroma components $u^*$, $v^*$ of this space are passed to the chroma processing steps in the model.[19]

### II.4.2 Luma processing

As shown in Figure II.11, each luma value is first subjected to a compressive non-linearity. Then, each luma field is filtered and down-sampled in a four-level Gaussian pyramid, in order to model the psychophysically and physiologically observed decomposition of incoming visual signals into different spatial-frequency bands. After this decomposition, the bulk of subsequent processing by the model consists of similar operations (e.g., oriented filtering) performed at each pyramid level.

After this pyramid-making process, the lowest-resolution pyramid image is subjected to temporal filtering and contrast computation, and the other three levels are subjected to spatial filtering and contrast computation. In each case, the contrast is a local difference of pixel values divided by a local sum, appropriately scaled. Initially, this establishes the definition of 1 JND, which is passed on to subsequent stages of the model.[20] (Calibration iteratively revises the 1-JND interpretations at intermediate model stages.) The absolute value of the contrast response is passed to the following stage, and the algebraic sign is preserved for reattachment just prior to image comparison (JND map computation).

[18] Rows in a half-height image correspond to one field, i.e., to either the even or odd lines of a frame.

[19] The luminance channel $L^*$ from CIELUV is not used in luma processing, but instead is replaced by a visual nonlinearity for which the vision model has been calibrated over a range of luminance values. L* is used in chroma processing, however, to create a chroma metric that is approximately uniform and familiar to display engineers.

[20] The association of a constant contrast with 1 JND is an implementation of what is known as Weber's law for vision.

The next stage (contrast masking) is a gain-setting operation in which each contrast response is divided by a function of all the contrast responses. This combined attenuation of each response by other local responses is included to model visual "masking" effects such as the decrease in sensitivity to distortions in "busy" image areas. At this stage in the model, temporal structure (flicker) is made to mask spatial differences, and spatial structure is also made to mask temporal differences. Luma masking is also applied on the chroma side, as discussed below.

The masked contrast responses (together with the contrast signs) are used to produce the Luma JND map. This is done by:

- separating each image into positive and negative components (half-wave rectification);
- performing local pooling (averaging and down-sampling, to model the local spatial summation observed in psychophysical experiments);
- evaluating the absolute image differences channel-by-channel;
- up-sampling to the same resolution (which will be half the resolution of the original image due to the pooling stage);
- evaluating the Minkowski Q-norm over all channels.

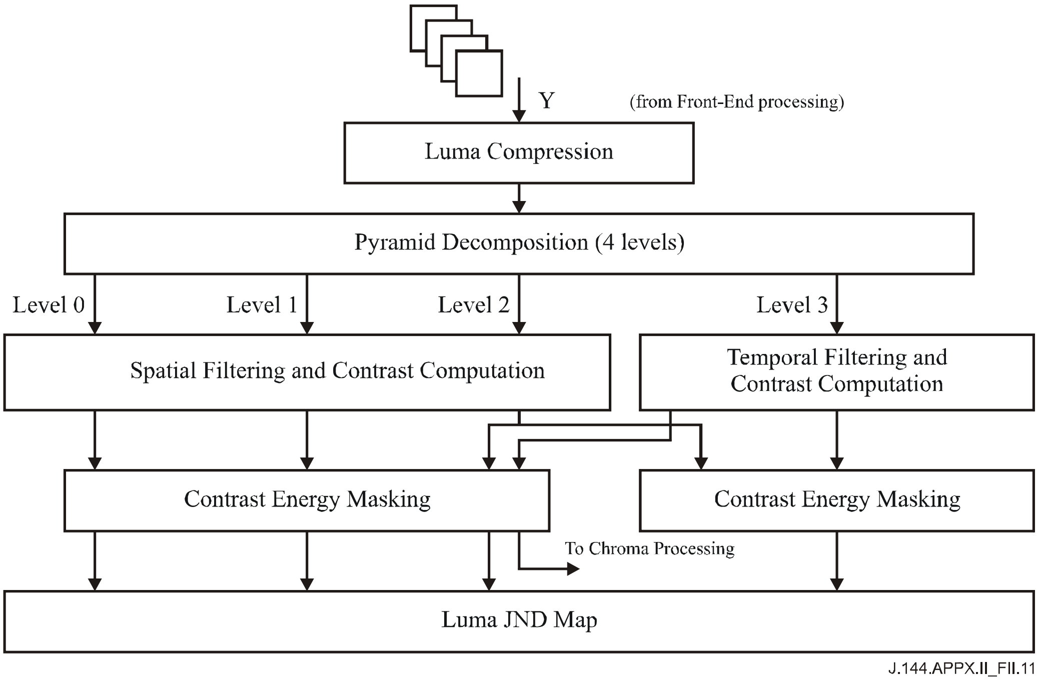

Y
(from Front-End processing)
Luma Compression
Pyramid Decomposition (4 levels)
Level 0
Level 1
Level 2
Level 3
Spatial Filtering and Contrast Computation
Temporal Filtering and Contrast Computation
Contrast Energy Masking
Contrast Energy Masking
To Chroma Processing
Luma JND Map
J.144.APPX.II_FII.11

**Figure II.11 – Luma processing overview**

### II.4.3 Chroma processing

Chroma processing parallels luma processing in several ways. Intra-image differences of chroma (u* and v*) of the CIELUV space are used to define the detection thresholds for the chroma model; in analogy to the way contrast (and Weber's law) is used to define the detection threshold in the luminance model. Also, in analogy with the luminance model, the chromatic "contrasts" defined by $u^*$ and $v^*$ differences are subjected to a masking step. A transducer non-linearity makes the discrimination of a contrast increment between one image and another depend on the contrast response that is common to both images.

Figure II.12 shows that, as in luma processing, each chroma component u*, v* is subjected to pyramid decomposition. However, whereas luma processing needs only four pyramid levels, chroma processing is given seven levels. This captures the empirical fact that chromatic channels are sensitive to far lower spatial frequencies than luma channels. Also, it takes into account the intuitive fact that color differences can be observed in large, uniform regions.

To reflect the inherent insensitivity of the chroma channels to flicker, temporal processing is accomplished by averaging over four image fields.

Then, spatial filtering by a Laplacian kernel is performed in u* and v*. This operation produces a color difference in u*, v*, which (by definition of the uniform color space) is metrically connected to just-noticeable color differences. A value of 1 at this stage is taken to mean a single JND has been achieved, in analogy to the role of Weber's-law-based contrast in the luma channel. (As in the case of luma, the 1-JND chroma unit must undergo reinterpretation during calibration.)

This color difference value is weighted, absolute-valued, and passed (with the contrast algebraic sign) to the contrast-masking stage. The masking stage performs the same function as it did in the luma model. It is somewhat simpler, since it receives input only from the luma channels and from the chroma channel whose difference is being evaluated. Finally, the masked contrast responses are processed exactly as in the luma model (see last paragraph of II.4.2).

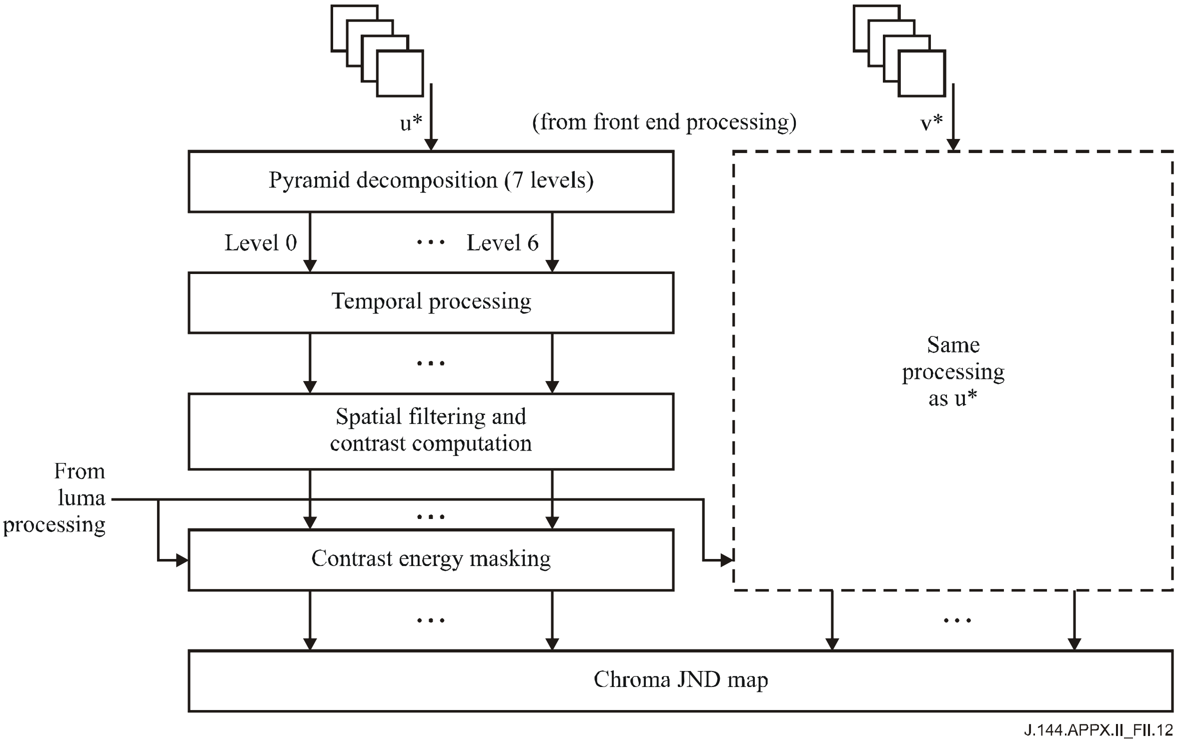


**Figure II.12 – Chroma processing overview**

### II.4.4 Output summaries

For each field in the video-sequence comparison, the luma and chroma JND maps are first combined to give a total-JND map. This total-JND map is computed as the square root of the sum of the squares of the luma and chroma map values, pixel-by-pixel.

Then, each of the three JND maps (luma, chroma, and combined luma-chroma) is reduced to a single-number summary, called a Picture Quality Rating (PQR) value. Single number summaries are computed by the Minkowski Q-norm. With this approach, each JND-map pixel value is raised to the Qth power. The PQR is then computed as the Qth root of a normalized sum of all Qth power pixel values.

Next, three single performance measures for many fields of a video sequence (one for luma, one for chroma, and one for combined luma-chroma) are computed. PQR values for each field in a sequence are reduced to one PQR for the entire sequence, again by a Minkowski Q-norm.

Although the PQR method is valid for a range of sequence lengths, for this appendix, PQR values shall be calculated for 60 frames (2 seconds) of video. It is also important to note that PQR values for sequences of one-half second or shorter may not compare well with subjective assessment. This is due to the fact that subjective assessment data is unreliable for such short sequences.

## II.5 Algorithm details

Application of the PQR measurement model described in this appendix requires the use the parametric values that calibrate the algorithm to approximate the response of the human visual system. Table II.2 shows the parametric values that shall be used in the implementation of this appendix.

**Table II.2 – PQR model parameter values**

| Parameter type | Symbol | Value | Clause |
|---|---|---|---|
| Luma compression | <br>m<br>$L_d$ | <br>0.65<br>7.5 cd/m$^2$ | II.5.2.1 |
| Temporal filtering 60 Hz | <br>$t_e$<br>$t_l$ | <br>33/64<br>31/64 | II.5.2.3.2 |
| Temporal filtering 50 Hz | <br>$t_e$<br>$t_l$ | <br>11/16<br>5/16 | II.5.2.3.2 |
| Luma contrast threshold<br>(by pyramid level) | <br>$w_0$<br>$w_1$<br>$w_2$<br>$w_3$ | <br>1/150<br>1/900<br>1/1280<br>1/500 | II.5.2.4 |
| Luma masking contrasts | <br>β<br>a<br>c<br>$m_f$<br>$m_t$<br>$m_{ft}$ | <br>1.4<br>3/32<br>5/32<br>10/1024<br>50<br>3/64 | II.5.2.5 |
| Chroma contrast threshold<br>(by pyramid level) | <br>$q_0$<br>$q_1$<br>$q_2$<br>$q_3$<br>$q_4$<br>$q_5$<br>$q_6$ | <br>384<br>60<br>24<br>6<br>4<br>3<br>3 | II.5.3.4 |
| Luma masking constants | <br>$\beta_c$<br>$a_c$<br>$c_c$<br>$m_c$<br>k | <br>1.4<br>0.5<br>0.5<br>10/1024<br>0.7 | II.5.3.5 |

### II.5.1 Front-end processing

See Figure II.10. Front-end processing transforms Y′, C′$_b$, C′$_r$ video input signals first to electron gun voltages, then to luminance values of three phosphors, and finally into psychophysical variables that separate into luma and chroma components. The tristimulus value Y, computed in II.5.1.3, replaces the "model intensity value" used before chroma processing was added to the JND model. In addition, chroma components $u^*$ and $v^*$ are generated, pixel-by-pixel, according to CIE uniform-color specifications.

#### II.5.1.1 (Y′, C′$_b$, C′$_r$) to (R′, G′, B′)

The steps outlined below describe the transformation from Y′, C′$_b$, C′$_r$ image frames to R′, G′, B′ voltage signals that drive the display. Here, the apostrophe indicates that the input signals have been gamma-pre-corrected at the encoder. These signals, after further transformation, drive a CRT display device[21] whose voltage-current transfer function can be closely approximated by a gamma non-linearity.

It is assumed here that the input digital images are in 4:2:2 format: full resolution on the luminance correlate Y′, and half-resolution horizontally for the chrominance correlates C′$_b$ and C′$_r$. Y′, C′$_b$, C′$_r$ data are assumed to be stored in the order specified in ITU-R Rec. BT.601-5, namely,

$$C'_{b0}, Y'_0, C'_{r0}, Y'_1, C'_{b1}, Y'_2, C'_{r1}, Y'_3, ..., C'_{bn/2-1}, Y'_{n-1}, C'_{rn/2-1}, Y'_{n-2}, ... .$$

*Step 1.* Input the Y′ C′$_b$ C′$_r$ arrays from a single frame. Then expand the C′$_b$ and C′$_r$ arrays to the full resolution of the Y′ image. The C′$_b$ and C′$_r$ arrays are initially at half-resolution horizontally, and must be up-sampled to create the full-resolution fields. To begin, the alternate C′$_b$, C′$_r$ pixels on a row are assigned to the even-numbered Y′$_i$ they bracket in the data stream. Then, the C′$_b$, C′$_r$ pair to associate with the odd-numbered Y′$_i$ are computed by averaging with its two nearest horizontal neighbors.

*Step 2.* Parcel the full-resolution Y′, C′$_b$, C′$_r$ arrays into two fields. In the case of Y′, the first field contains the odd lines of the Y′ array, and the second field contains the even lines of the Y′ array. Identical processing is performed on C′$_b$ and C′$_r$ arrays to produce the first and second C′$_b$ and C′$_r$ fields.

*Step 3.* For each pixel in each of the two fields, convert the corresponding Y′, C′$_b$, C′$_r$ values to the gun input values R′, G′, B′. For the purposes of this appendix, the Y′, C′$_b$, C′$_r$ values are taken to be related to the R′G′B′ values by the following equation:

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.371 \\ 1 & -0.336 & -0.698 \\ 1 & 1.732 & 0 \end{bmatrix} \left[ \begin{Bmatrix} Y' \\ C'_b \\ C'_r \end{Bmatrix} - \begin{Bmatrix} 0 \\ 128 \\ 128 \end{Bmatrix} \right] \qquad \text{(II-1)}$$

The R′, G′and B′ arrays are now ready for the next step in the front-end processing algorithm.

#### II.5.1.2 (R′, G′, B′) to (R, G, B)

##### II.5.1.2.1 Pixel-value transformation

Compute for each pixel the fraction of maximum luminance R corresponding to input R′. Similarly, compute the fractional luminances G and B from inputs G′, B′. The maximum luminance from each gun is assumed to correspond to the input value 255. The following equations describe the transformation from (R′, G′, B′) to (R, G, B):

[21] See II.5.1.2.1 for a description of the CRT display model.

$$R = \left[\frac{\max(R', t_d)}{255}\right]^{\gamma}$$

$$G = \left[\frac{\max(G', t_d)}{255}\right]^{\gamma} \qquad \text{(II-2)}$$

$$B = \left[\frac{\max(B', t_d)}{255}\right]^{\gamma}$$

Here, the default threshold value $t_d$ is taken to be 16 to correspond with the black level of the display, and $\gamma$ defaults to 2.5. The value of 16 for $t_d$ is chosen to give the display a dynamic range of about 1000:1 (i.e., $(255/16)^{2.5}$).

#### II.5.1.2.2 Full- and half-height image processing options

The PQR model provides two options for specifying the vertical representation of (R, G, B) images, for each frame (in progressive images) and for odd and even fields (in interlaced images).

1) *Frame*

   Images are full-height and contain one progressively scanned image.

2) *Half-height Interlace*

   Half-height images are processed directly.

The first six subclauses in II.5.2 and II.5.3 describe full-height Luma and Chroma processing. Clauses II.5.2.7 and II.5.3.7 describe half-height processing.

#### II.5.1.3 (R, G, B) to (X, Y, Z)

Compute the CIE 1931 tristimulus values X, Y, and Z for each pixel, given the fractional luminance values R, G, B. This involves the following inputs that depend on the display device: the chromaticity coordinates $(x_r, y_r)$, $(x_g, y_g)$, $(x_b, y_b)$ of the three phosphors, and the chromaticity of the monitor white point $(x_w, y_w)$.

White point chromaticities $(x_w, y_w) = (0.3127, 0.3290)$ correspond to Illuminant D65. Table II.3 shows the display phosphor coordinate options.

**Table II.3 – Display phosphor coordinate options**

| Source | $(x_r, y_r)$ | $(x_g, y_g)$ | $(x_b, y_b)$ |
|---|---|---|---|
| ITU-R Rec. BT.709-5 (SMPTE 274M) | (0.640,0.330) | (0.300,0.600) | (0.150,0.060) |
| SMPTE 240M | (0.630,0.340) | (0.310,0.595) | (0.155.0.070) |
| EBU | (0.640,0.330) | (0.290,0.600) | (0.150,0.060) |

Given the above parameter values, the values X, Y, Z of the pixel are given by the following equations:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{x_r}{y_r} Y_{or} & \frac{x_g}{y_g} Y_{og} & \frac{x_b}{y_b} Y_{ob} \\ Y_{or} & Y_{og} & Y_{ob} \\ \frac{z_r}{y_r} Y_{or} & \frac{z_g}{y_g} Y_{og} & \frac{z_b}{y_b} Y_{ob} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \text{(II-3)}$$

Here, $z_r = (1–x_r–y_r)$, $z_g = (1–x_g–y_g)$, $z_b = (1–x_b–y_b)$, and the values $Y_{0r}$, $Y_{0g}$, $Y_{0b}$ are given by

$$\begin{bmatrix} Y_{0r} \\ Y_{0g} \\ Y_{0b} \end{bmatrix} = \begin{bmatrix} \frac{x_r}{y_r} & \frac{x_g}{y_g} & \frac{x_b}{y_b} \\ 1 & 1 & 1 \\ \frac{z_r}{y_r} & \frac{z_g}{y_g} & \frac{z_b}{y_b} \end{bmatrix}^{-1} \begin{bmatrix} \frac{x_w}{y_w} \\ 1 \\ \frac{z_w}{Y_w} \end{bmatrix} \tag{II-4}$$

where $z_w = (1–x_w–y_w)$.

The tristimulus values $X_n$, $Y_n$, $Z_n$ of the white point of the device will also be needed. They correspond to the chromaticity $(x_w, y_w)$ and are such that, at full phosphor activation (R′ = G′ = B′ = 255), Y = 1. The tristimulus values for the white point are $(X_n, Y_n, Z_n) = (x_w/y_w, 1, z_w/y_w)$.

As a final stage in deriving the values X, Y, Z, an adjustment is made to accommodate an assumed ambient light due to veiling reflection from the display screen. This adjustment takes the form

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \leftarrow \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \left( \frac{L_a}{L_{\max}} \right) \begin{bmatrix} X_n \\ Y_n \\ Z_n \end{bmatrix} \tag{II-5}$$

Here, two user-specifiable parameters, $L_{max}$ and $L_a$, are introduced and assigned default values. $L_{max}$, the maximum luminance of the display, is set to 100 cd/m$^2$ to correspond to commercial displays. The veiling luminance, $L_a$, is set to 5 cd/m$^2$, consistent with measured screen values under BT.500-11 conditions.

The chromaticity of the ambient light is assumed to be the same as that of the display white point. It should be noted that in the luma-only model option, which does not compute the neutral point $(X_n, Y_n, Z_n)$, the adjustment

$$Y \leftarrow Y + \frac{L_a}{L_{\max}} \tag{II-6}$$

is made instead of equation II-5. This is equivalent to the Y component of equation II-5 because $Y_n$ is always 1. Note also that the quantity $L_{max}$ *Y is the luminance of the display in cd/m$^2$.

### II.5.1.4 (X, Y, Z) to (L$^*$, u$^*$, v$^*$)

Transform the X, Y, Z values, pixel-by-pixel, to the 1976 CIELUV uniform-color system:

$$L^* = 116 \left( \frac{Y}{Y_n} \right)^{1/3} - 16 \qquad \mathit{for}\ \frac{Y}{Y_n} > 0.008856$$

$$L^* = 903.3 \left( \frac{Y}{Y_n} \right) \qquad \mathit{for}\ \frac{Y}{Y_n} \le 0.008856 \tag{II-7}$$

$$u^* = 13L^*(u' - u'_n) \tag{II-8}$$

$$v^* = 13L^*(v' - v'_n) \tag{II-9}$$

Here,

$$u' = \frac{4X}{(X + 15Y + 3Z)} \qquad \text{(II-10)}$$

$$v' = \frac{9Y}{(X + 15Y + 3Z)} \qquad \text{(II-11)}$$

$$u'_n = \frac{4X_n}{(X_n + 15Y_n + 3Z_n)} \qquad \text{(II-12)}$$

$$v'_n = \frac{9Y_n}{(X_n + 15Y_n + 3Z_n)} \qquad \text{(II-13)}$$

Note that the coordinate L* does not enter the luminance computation. L* is used only in computing the chroma coordinates u* and v*.[22] Consequently, out of the above quantities, only u* and v* images are saved for further processing.

### II.5.2 Luma processing

See Figure II.13. In this clause, input test and reference field images are denoted by $I_k$ and $I^{ref}_k$ (k = 0, 1, 2, 3). Pixel values in $I_k$ and $I^{ref}_k$ are denoted by $I_k(i,j)$ and $I^{ref}_k(i,j)$, respectively. They start out as Y tristimulus values computed in front-end processing. Only the fields $I_k$ are discussed in the following. $I^{ref}_k$ processing is identical. k=3 denotes the most recent field in a 4-field sequence.

Clauses II.5.2.1 to II.5.2.6 describe full-height processing. Clause II.5.2.7 discusses the modifications required for half-height processing.

#### II.5.2.1 Luma compression

The first step of the luma model is a non-linearity comprising a decelerating power function offset by a constant. Let the relative-luminance array from the latest field be $Y_3(i,j)$, where 3 denotes the latest field. Then

$$I_3(i,j) = [L_{\max} Y_3(i,j)]^n + L_d^m \qquad \text{(II-14)}$$

$L_{max}$, the maximum luminance of the display, is set to 100 cd/m$^2$. The values of $L_d$ and m were chosen so as to match contrast detection data at luminance levels from 0.01 to 100 ft-L.

#### II.5.2.2 Luma pyramid decomposition

Spatial decomposition at four resolution levels is done through a computationally efficient method called pyramid processing, which smears and down-samples the image by a factor of 2 at each successively coarser level of resolution.

[22] The luminance channel L$^*$ from CIELUV is not used in luma processing, but instead is replaced by a visual non-linearity for which the vision model has been calibrated over a range of luminance values. L* is used in chroma processing, however, to create a chroma metric that is approximately uniform and familiar to display engineers.

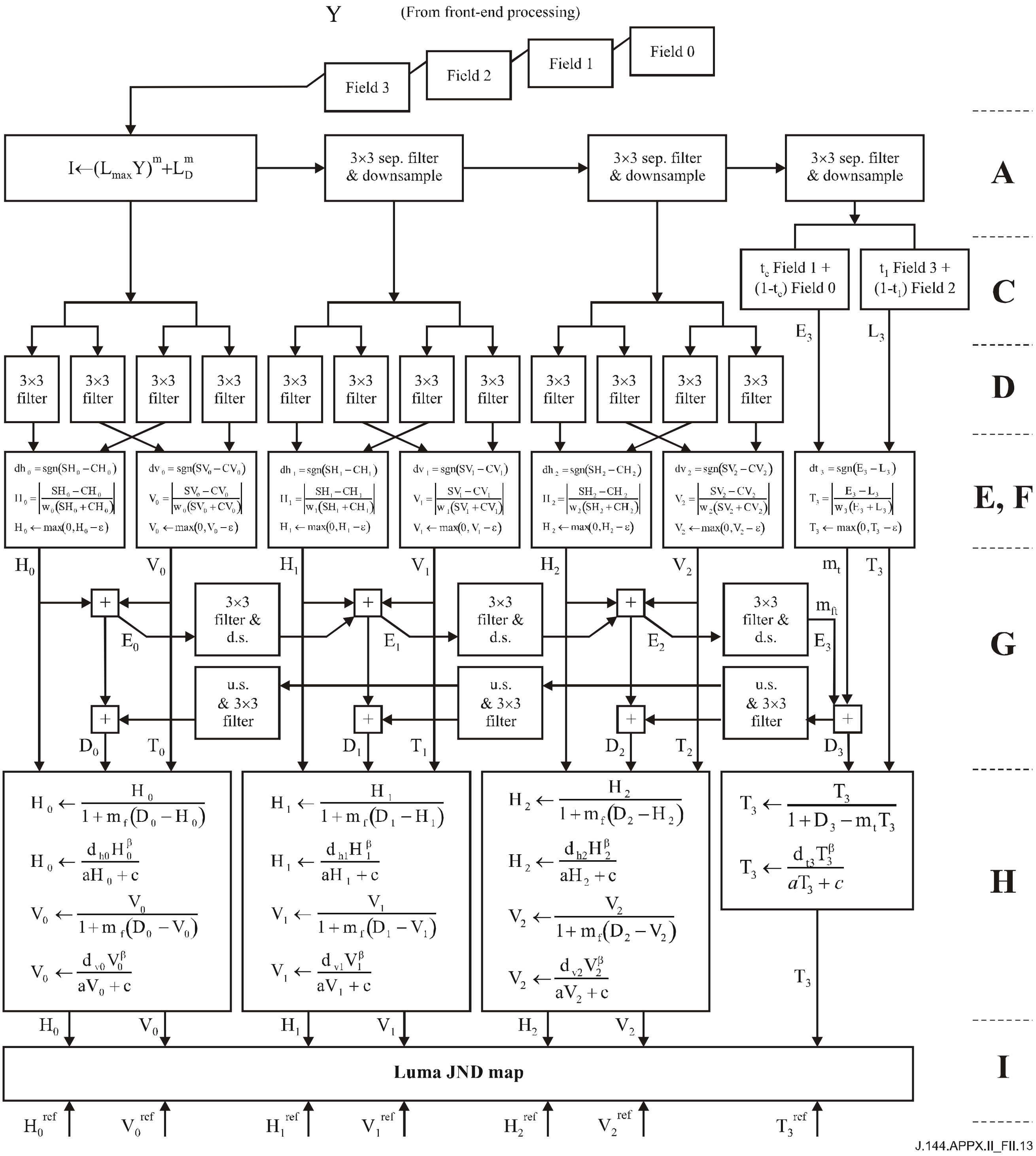

Y
(From front-end processing)
Field 3
Field 2
Field 1
Field 0
$I \leftarrow (L_{max}Y)^m + L_D^m$
3×3 sep. filter & downsample
3×3 sep. filter & downsample
3×3 sep. filter & downsample
$t_e$ Field 1 + (1-$t_e$) Field 0
$t_l$ Field 3 + (1-$t_l$) Field 2
$E_3$
$L_3$
3×3 filter
$dh_0 = sgn(SH_0 - CH_0)$
$H_0 = \left|\frac{SH_0 - CH_0}{w_0(SH_0 + CH_0)}\right|$
$H_0 \leftarrow max(0, H_0 - \varepsilon)$
$dv_0 = sgn(SV_0 - CV_0)$
$V_0 = \left|\frac{SV_0 - CV_0}{w_0(SV_0 + CV_0)}\right|$
$V_0 \leftarrow max(0, V_0 - \varepsilon)$
$dh_1 = sgn(SH_1 - CH_1)$
$H_1 = \left|\frac{SH_1 - CH_1}{w_1(SH_1 + CH_1)}\right|$
$H_1 \leftarrow max(0, H_1 - \varepsilon)$
$dv_1 = sgn(SV_1 - CV_1)$
$V_1 = \left|\frac{SV_1 - CV_1}{w_1(SV_1 + CV_1)}\right|$
$V_1 \leftarrow max(0, V_1 - \varepsilon)$
$dh_2 = sgn(SH_2 - CH_2)$
$H_2 = \left|\frac{SH_2 - CH_2}{w_2(SH_2 + CH_2)}\right|$
$H_2 \leftarrow max(0, H_2 - \varepsilon)$
$dv_2 = sgn(SV_2 - CV_2)$
$V_2 = \left|\frac{SV_2 - CV_2}{w_2(SV_2 + CV_2)}\right|$
$V_2 \leftarrow max(0, V_2 - \varepsilon)$
$dt_3 = sgn(E_3 - L_3)$
$T_3 = \left|\frac{E_3 - L_3}{w_3(E_3 + L_3)}\right|$
$T_3 \leftarrow max(0, T_3 - \varepsilon)$
$H_0$ $V_0$ $H_1$ $V_1$ $H_2$ $V_2$ $m_t$ $T_3$
+
$E_0$ $E_1$ $E_2$ $E_3$
$m_{ft}$
3×3 filter & d.s.
u.s. & 3×3 filter
$D_0$ $T_0$ $D_1$ $T_1$ $D_2$ $T_2$ $D_3$
$H_0 \leftarrow \frac{H_0}{1 + m_f(D_0 - H_0)}$
$H_0 \leftarrow \frac{d_{h0}H_0^\beta}{aH_0 + c}$
$V_0 \leftarrow \frac{V_0}{1 + m_f(D_0 - V_0)}$
$V_0 \leftarrow \frac{d_{v0}V_0^\beta}{aV_0 + c}$
$H_1 \leftarrow \frac{H_1}{1 + m_f(D_1 - H_1)}$
$H_1 \leftarrow \frac{d_{h1}H_1^\beta}{aH_1 + c}$
$V_1 \leftarrow \frac{V_1}{1 + m_f(D_1 - V_1)}$
$V_1 \leftarrow \frac{d_{v1}V_1^\beta}{aV_1 + c}$
$H_2 \leftarrow \frac{H_2}{1 + m_f(D_2 - H_2)}$
$H_2 \leftarrow \frac{d_{h2}H_2^\beta}{aH_2 + c}$
$V_2 \leftarrow \frac{V_2}{1 + m_f(D_2 - V_2)}$
$V_2 \leftarrow \frac{d_{v2}V_2^\beta}{aV_2 + c}$
$T_3 \leftarrow \frac{T_3}{1 + D_3 - m_t T_3}$
$T_3 \leftarrow \frac{d_{t3}T_3^\beta}{aT_3 + c}$
$T_3$
Luma JND map
$H_0^{ref}$ $V_0^{ref}$ $H_1^{ref}$ $V_1^{ref}$ $H_2^{ref}$ $V_2^{ref}$ $T_3^{ref}$
A
C
D
E, F
G
H
I
J.144.APPX.II_FII.13

**Figure II.13 – Luma processing detail**

The original, full-resolution image is called the zeroth level of the pyramid, $G_0 = I_3(i,j)$. Subsequent levels, at lower resolutions, are obtained by an operation called REDUCE, which works as follows. A three-tap low-pass filter with weights (1,2,1)/4 is applied to $G_0$ sequentially in each direction of the image to generate a blurred image. The resulting image is then subsampled by a factor of 2 (every other pixel is removed as shown by the shaded pixels in Figure II.14 below) to create the next level, $G_1$.

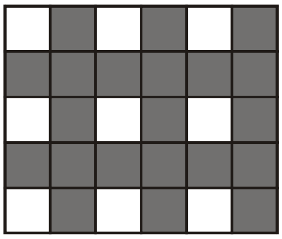


**Figure II.14 – Image subsampling with gray pixels removed**

Denoting fds1() as the operation of filtering and down-sampling by one pyramid level, the REDUCE process can be represented as

$$G_{i+1} = fds1(G_i), \ for \ i = 1, 2, 3. \qquad \text{(II-15)}$$

The REDUCE process is applied recursively to each new level (as described by Burt and Adelson, 1983).

Conversely, an operation EXPAND is defined that up-samples and filters by the same 3×3 kernel. This operation is denoted by usf1(), and appears in the context of II.5.2.5 and II.5.2.6.

The fds1 and usf1 filter kernels in each direction (horizontal and vertical) are $k_d$ [1,2,1] and $k_u$ [1,2,1], respectively, where constants $k_d$ and $k_u$ are chosen so that uniform-field values are conserved. For fds1, the constant is $k_d = 0.25$, and for ufs1, the constant is $k_u = 0.5$ (because of the zeros in the up-sampled image). To implement usf1 as an in-place operation, the kernel is replaced by the equivalent linear interpolation to replace the zero values. However, for conceptual simplicity, we still refer to the operation as "up-sample-filter".

### II.5.2.3 Luma spatial and temporal filtering

Oriented spatial filters (center and surround) are applied to the level 0, 1, and 2 images for field 3. At the lowest resolution level (level 3), the first and last pairs of fields are combined linearly into Early and Late images, respectively.

#### II.5.2.3.1 Spatial filtering

The centre and surround filters are separable 3×3 filters and yield all combinations of orientation: Center Vertical (CV), Center Horizontal (CH), Surround Vertical (SV), and Surround Horizontal (SH). The filter kernels are as follows:

$$\text{CH} = \begin{bmatrix} 000 \\ 242 \\ 000 \end{bmatrix}; \ \text{SH} = \begin{bmatrix} 121 \\ 000 \\ 121 \end{bmatrix}; \ \text{CV} = \begin{bmatrix} 020 \\ 040 \\ 020 \end{bmatrix}; \ \text{SV} = \begin{bmatrix} 101 \\ 202 \\ 101 \end{bmatrix} \qquad \text{(II-16)}$$

#### II.5.2.3.2 Temporal filtering

The level 3 Early and Late images are, respectively,

$$E_3 = t_e I_{3,1}(i,j) + (1 - t_e) I_{3,0}(i,j) \qquad \text{(II-17)}$$

$$L_3 = t_l I_{3,3}(i,j) + (1 - t_l) I_{3,2}(i,j) \qquad \text{(II-18)}$$

The constants $t_e$ and $t_l$ for 60 Hz are different from the values at 50 Hz.

### II.5.2.4 Luma contrast computation

Inputs are the centre and surround images $CV_i$, $CH_i$, $SV_i$, and $SH_i$ (i=0,1,2 for pyramid levels 0, 1, and 2), and the Early and Late images $E_3$ and $L_3$ (level 3) computed in II.5.2.3. The formula used to compute the contrast ratio is analogous to the Michelson contrast, $(L_{max} - L_{min})/(L_{max} + L_{min})$, which has proven successful for vision modeling. For the horizontal and vertical orientations, the respective contrasts, pixel-by-pixel, are

$$\frac{(SH_i - CH_i)}{w_i(CH_i + SH_i)} \; and \; \frac{(SV_i - CV_i)}{w_i(CV_i + SV_i)} \tag{II-19}$$

Similarly, the contrast ratio for the temporal component is

$$\frac{(E_3 - L_3)}{w_3(E_3 + L_3)} \tag{II-20}$$

Values of $w_i^{-1}$ for i = 0,1,2,3 were found by calibration.

Contrast-response images are computed as clipped versions of the absolute values of the quantities defined by the two preceding equations. These quantities are computed as

$$H_i = \max\left(0, \left|\frac{(SH_i - CH_i)}{w_i(CH_i + SH_i)}\right| - \varepsilon\right), V_i = \max\left(0, \left|\frac{(SV_i - CV_i)}{w_i(CV_i + SV_i)}\right| - \varepsilon\right) \tag{II-21}$$

i = 0,1,2, and

$$T_3 = \max\left(0, \left|\frac{(E_3 - L_3}{w_3(E_3 + L_3)}\right| - \varepsilon\right), \; where \; \varepsilon = 0.75. \tag{II-22}$$

The algebraic sign of each contrast ratio pixel value prior to the absolute-value operation (Steps E, F in Figure II.13) must be retained for use later in Step H.

### II.5.2.5 Luma contrast masking

Contrast masking is a non-linear function applied to each of the contrast responses computed in II.5.2.4. It models the effect of spatiotemporal structure in the reference image sequence on the discrimination of distortion in the test image sequence.

Suppose, for example, a test and a reference image differ by a low-amplitude spatial sine wave. It is known that this difference is more visible when both images have in common a mid-contrast sine wave of the same spatial frequency, than if both images contain a uniform field. However, if the contrast of the common sine wave is too great, the image difference becomes less visible. It is also the case that sine waves of other spatial frequencies can have an effect on the visibility of the contrast difference. This behavior can be modelled by a non-linearity that is sigmoid at low contrast energies, and an increasing power function for high contrast energies. Furthermore, the following rules can be observed approximately in human vision. Each channel masks itself, high spatial frequencies mask low ones (but not the reverse), and temporal flicker masks spatial contrast sensitivity (and also the reverse).

In response to these properties of vision, in the present model, the following form for the non-linearity (applied pixel-by-pixel) is used:

$$T(y, D_i) = \frac{d_y Z_i^{\beta}}{a z_i + c} \tag{II-23}$$

where $z_i = \dfrac{y}{[1 + m_f(D_i - y)]}$ for i = 0,1,2, and $z_3 = \dfrac{y}{(1 + D_3 - m_t y)}$

Here, y is the contrast to be masked: spatial, $H_i$ or $V_i$ (equation II.21) or temporal ($T_3$) (equation II.22). The quantity $D_i$ refers (pixel by pixel) to an image that depends on the pyramid level i to which y belongs. Quantities B, a, c, $m_f$, and $m_t$ were found by calibration. $d_y$ is the algebraic sign of contrast y that is saved before taking the absolute value.

Computation of $D_i$ requires pyramid construction (filtering followed by down-sampling) and pyramid reconstruction (up-sampling followed by filtering). This can be seen from Figure II.13 and by the equations below. In these equations, fds1() denotes 3×3 filtering followed by down-sampling by one pyramid level, and usf1() denotes up-sampling by one pyramid level followed by 3×3 filtering (see end of II.5.2.2). First, array $E_0$ is computed as

$$E_0 = H_0 + V_0 \tag{II-24}$$

Then, for i = 1, 2, the arrays $E_i$ are computed recursively:

$$E_i = H_i + V_i + fds1(E_{i-1}), \text{ for i = 1,2} \tag{II-25}$$

$$E_3 = fds1(E_2) \tag{II-26}$$

The arrays $E_i$ are then combined with the temporal contrast image $T_3$ and images $T_i$ to give the contrast denominator arrays $D_i$, as follows:

$$D_3 = m_t T_3 + m_{ft}\, fds1(E_2), \tag{II-27}$$

$$T_2 = usf1(D_3),\ \ T_i = usf1(T_{i+1}), \text{ for i = 1,0, and}$$

$$D_i = E_i + T_i, \text{ for i = 0,1,2} \tag{II-28}$$

Here, parameter $m_{ft}$ modulates the strength with which the temporal (flicker) luma-channel is masked by all the spatial-luma channels together; and parameter $m_t$ modulates the strength with which each of the spatial-luma channels is masked by the temporal (flicker) luma-channel.

It can be seen from the above processing that the higher spatial frequencies mask the lower ones (since $D_i$ are influenced by pyramid levels less than or equal to i), and the temporal channel masks, and is masked by, all the spatial channels. This is roughly in accord with psychophysical observation. As will be seen, the quantities $D_i$, i = 0,1,2, also mask chroma contrasts (but not the reverse).

### II.5.2.6 Luma JND map construction

The construction described below applies to all the masked-contrast images generated by step H above (see Figure II.13).

- the images in pyramids H and V (i.e., images $H_0$, $V_0$, $H_1$, $V_1$, $H_2$, and $V_2$);
- the image $T_3$ (having resolution at level 3);
- the corresponding images derived from the reference sequence (denoted with superscript$^{ref}$ in Figure II.13).

The first four steps in the following process apply to the above images separately. In discussing them, we denote by X any of these images derived from the test sequence, and by $X^{ref}$ the corresponding image derived from the reference sequence. Given this notation, here are the steps:

- Separate image X into two half-wave-rectified images, one for positive contrasts and the other for negative contrasts. In the positive-contrast image (called $X_+$), the signs from the X contrast (separately stored at stage E) are used to assign zeros to all pixels in $X_+$ that have negative contrasts. The opposite happens in the negative-contrast image $X_–$.
- For each image $X_+$ and $X_–$, perform the local pooling operation suggested by psychophysics by convolving the image with the kernel 0.25(1,2,1) both horizontally and vertically.

- Down-sample the resulting images by a factor of 2 in each direction, to remove redundancy resulting from pooling in the previous step. Presuming that the same processing as was done for X has been done for the corresponding reference image $X^{ref}$, compute pixel-by-pixel the absolute-difference images $| X_+ – X_+^{ref}|$ and $| X_– – X_–^{ref}|$. The resulting images are JND maps.

After this process has been completed for all pairs X, $X^{ref}$, repeatedly up-sample, filter, and add all the images to the level required to compute summary measures. This is done as follows:

- Initialize a running-sum image to contain the sum of the Q'th power of the level-3 images derived from $T_3$, $T_3^{ref}$: $| T_{3+} – T_{3+}^{\ ref} |^Q$ and $| T_{3–} – T_{3–}^{ref}|^Q$. Here, Q has the value 2.
- Up-sample/filter the running-sum image to comprise a level-2 image.
- Update the running-sum image by adding to it the Q'th power of the level-2 images derived from $H_2$, $H_2^{ref}$, $V_2$ and $V_2^{ref}$.
- Up-sample/filter the running-sum image to comprise a level-1 image.
- Update the running-sum image by adding to it the Q'th power of the level-1 images derived from $H_1$, $H_1^{ref}$, $V_1$ and $V_1^{ref}$.
- Upsample/filter the running-sum image to comprise a level-0 image.
- Update the running-sum image by adding to it the Q'th power of the level-0 images derived from $H_0$, $H_0^{ref}$, $V_0$ and $V_0^{ref}$. Send this image directly to summary processing (see Figure II.9 and II.5.4).

Note that after this process, the resulting image is half the resolution of the original. In a similar vein, note that each pyramid-level index in this clause refers to the pyramid level from which it was originally derived, which is twice the resolution of that associated with that level after filtering/down-sampling.

### II.5.2.7 Half-height luma processing

If the half-height images are to be passed through directly without zero-filling to the true image height, then the above luma processing must be modified to reflect that the inherent vertical resolution is only half the inherent horizontal resolution. Figure II.15 summarizes luma for the half-height algorithm.

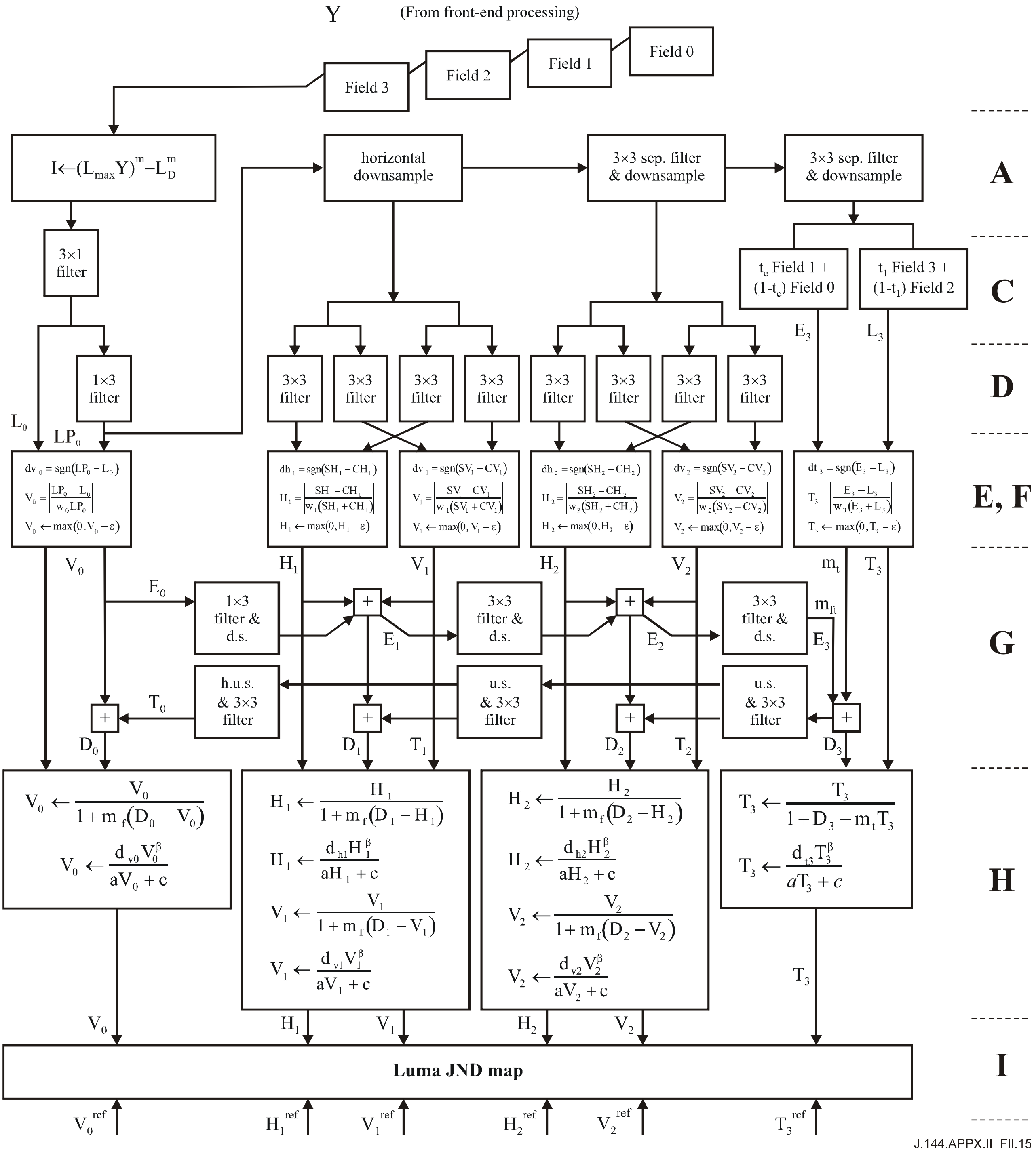

Y
(From front-end processing)
Field 3
Field 2
Field 1
Field 0
I←(L_max Y)^m+L_D^m
horizontal downsample
3×3 sep. filter & downsample
3×3 sep. filter & downsample
A
3×1 filter
t_c Field 1 + (1-t_c) Field 0
t_l Field 3 + (1-t_l) Field 2
C
E_3
L_3
1×3 filter
3×3 filter
D
L_0
LP_0
dv_0 = sgn(LP_0 − L_0)
V_0 = |(LP_0 − L_0)/(w_0 LP_0)|
V_0 ← max(0,V_0 − ε)
dh_1 = sgn(SH_1 − CH_1)
H_1 = |(SH_1 − CH_1)/(w_1(SH_1 + CH_1))|
H_1 ← max(0,H_1 − ε)
dv_1 = sgn(SV_1 − CV_1)
V_1 = |(SV_1 − CV_1)/(w_1(SV_1 + CV_1))|
V_1 ← max(0,V_1 − ε)
dh_2 = sgn(SH_2 − CH_2)
H_2 = |(SH_2 − CH_2)/(w_2(SH_2 + CH_2))|
H_2 ← max(0,H_2 − ε)
dv_2 = sgn(SV_2 − CV_2)
V_2 = |(SV_2 − CV_2)/(w_2(SV_2 + CV_2))|
V_2 ← max(0,V_2 − ε)
dt_3 = sgn(E_3 − L_3)
T_3 = |(E_3 − L_3)/(w_3(E_3 + L_3))|
T_3 ← max(0,T_3 − ε)
E, F
V_0
H_1
V_1
H_2
V_2
m_t
T_3
E_0
1×3 filter & d.s.
3×3 filter & d.s.
3×3 filter & d.s.
m_fl
E_1
E_2
E_3
G
h.u.s. & 3×3 filter
u.s. & 3×3 filter
u.s. & 3×3 filter
T_0
D_0
D_1
T_1
D_2
T_2
D_3
V_0 ← V_0/(1+m_f(D_0 − V_0))
V_0 ← d_v0 V_0^β/(aV_0 + c)
H_1 ← H_1/(1+m_f(D_1 − H_1))
H_1 ← d_h1 H_1^β/(aH_1 + c)
V_1 ← V_1/(1+m_f(D_1 − V_1))
V_1 ← d_v1 V_1^β/(aV_1 + c)
H_2 ← H_2/(1+m_f(D_2 − H_2))
H_2 ← d_h2 H_2^β/(aH_2 + c)
V_2 ← V_2/(1+m_f(D_2 − V_2))
V_2 ← d_v2 V_2^β/(aV_2 + c)
T_3 ← T_3/(1+D_3 − m_t T_3)
T_3 ← d_t3 T_3^β/(aT_3 + c)
H
T_3
V_0
H_1
V_1
H_2
V_2
Luma JND map
I
V_0^ref
H_1^ref
V_1^ref
H_2^ref
V_2^ref
T_3^ref
J.144.APPX.II_FII.15

**Figure II.15 – Luma processing detail (half-height)**

Comparison between this diagram and the corresponding diagram for full-height (Figure II.13) reveals the following basic changes:

1) The highest-resolution horizontal channel, $H_0$, is eliminated.

2) After step A, the highest resolution image is lowpass-filtered vertically (i.e., along columns) with a 3×1 "Kell" filter with weights (1/8, 3/4, 1/8). This operation corresponds to the joint filtering of the assumed de-interlace filter, together with the filtering performed by the vertical components of the 3×3 filters in step D of the full-height algorithm. The resulting vertically filtered image, $L_0$, is then horizontally filtered with a 1×3 filter (kernel 0.25[1,2,1]). The resulting image, $LP_0$, is a horizontally low-passed version of $L_0$.

3) $L_0$ and $LP_0$ are combined in step E,F to produce a bandpass ($LP_0 - L_0$) divided by lowpass ($LP_0$) oriented response analogous to the (S–C)/(S+C) responses of the other oriented channels.

4) Image $LP_0$ (a half-height image of 720 × 240 pixels) is horizontally down-sampled in stage A to a full height half-resolution image (360×240). Processing on this image, and throughout the remaining three pyramid levels, continues as in the full-height options.

5) In step G, down and up-sampling between the half-height images from Level 0 and the full height images of Level 1 is done with 1×3 filtering/horizontal down-sampling (labelled 1×3 filter and d.s.) and horizontal up-sampling (h.u.s.)/1×3 filtering respectively. Horizontal downsampling means decimation by a factor of two in the horizontal dimension; i.e., throwing out every other column of the image. Horizontal up-sampling means putting in a column of zeros between each two columns of the existing image. The filter kernel after up-sampling is defined by 0.5 [1,2,1], for the reason noted at the end of II.5.2.2.

In addition, in JND map construction, 3×3 filter and downsampling from $V_0$ is replaced with 1×3 filtering and horizontal downsampling.

### II.5.3 Chroma processing

Clauses II.5.3.1 to II.5.3.6 describe full-height processing. Clause II.5.3.7 discusses the modifications required for half-height processing.

#### II.5.3.1 Chroma pyramid decomposition

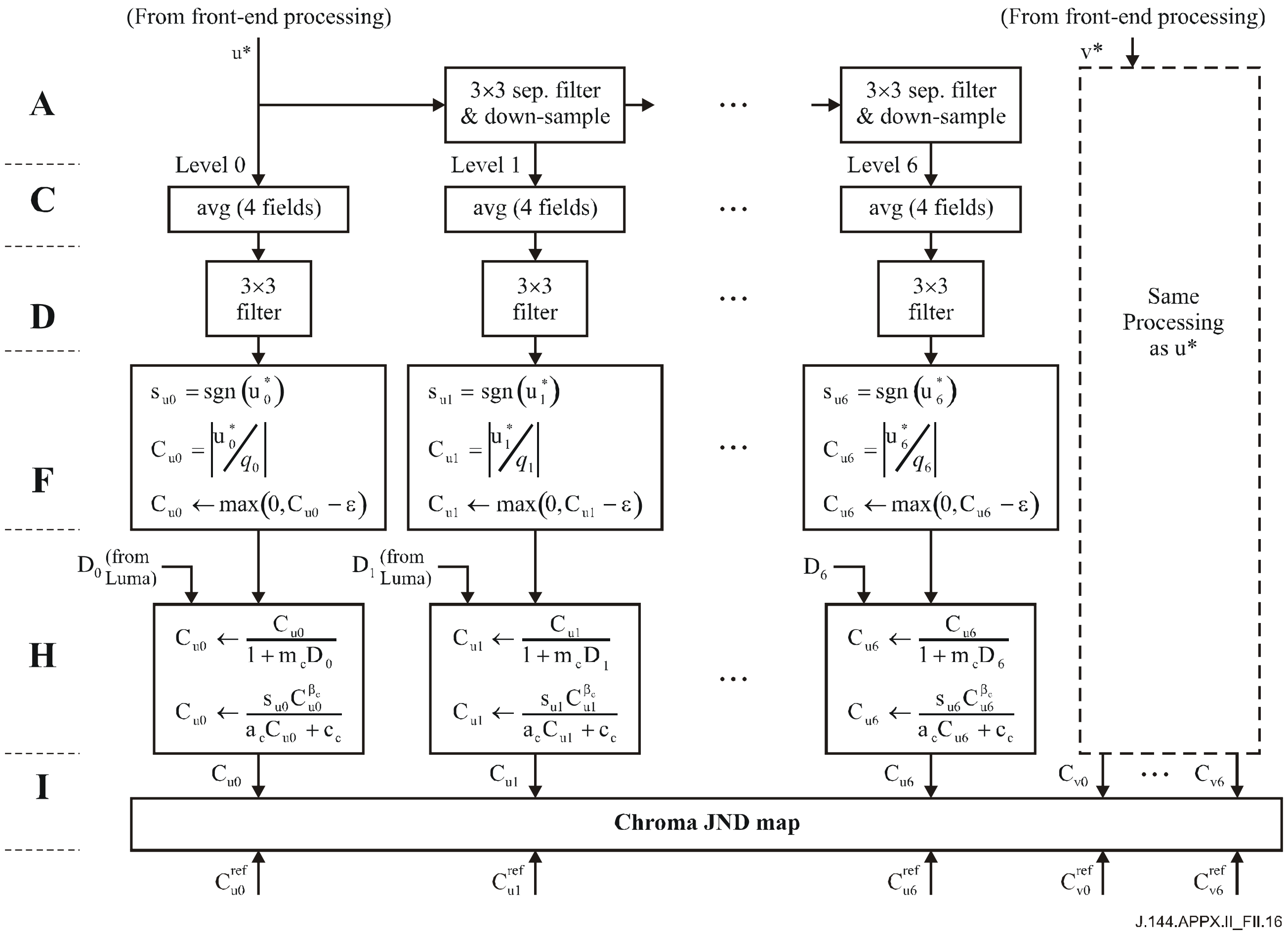
In addition to the pyramid with levels 0, 1, 2 (as computed for Y in Luma Processing), compute pyramids with levels 3, 4, 5, 6 for both u* and v*. Let

$$u_0 = u^*,\ v_0 = v^*,\ u_i = fds1(u_{i-1}),\ v_i = fds1(v_{i-1}),\ i = 1,...,6, \qquad \text{(II-29)}$$

where fds1() denotes the operation of filtering and downsampling described in II.5.2.2. See Figure II.16.

The spatial resolution of the highest resolution chroma channel (Level 0) is chosen equal to that of Level 0 luminance channel because the resolution is driven by the inter-pixel spacing, and not by the inter-receptor spacing. The inter-receptor spacing is 0.007 degrees of visual angle, and the inter-pixel spacing is 0.03 degrees, derived from a screen with 480 pixels in its height, viewed at four times its height. Also, the resolution of the blue-yellow chromatic channel is limited by the fact that the visual system is tritanopic (blue blind) for lights subtending less than about 2' (or .033 deg.) of visual angle. The pixel resolution of 0.03 degrees of visual angle is so close to the largest of these values that it is safe to equate the pixel resolutions of luminance and chroma channels.

The chroma pyramid extends to level 6 instead of 2. This supports evidence that observers notice differences between large, spatially uniform fields of color.


(From front-end processing)
u*
(From front-end processing)
v*
A
C
D
F
H
I
3×3 sep. filter & down-sample
Level 0
Level 1
Level 6
avg (4 fields)
3×3 filter
$s_{u0} = \mathrm{sgn}(u_0^*)$
$C_{u0} = |u_0^*/q_0|$
$C_{u0} \leftarrow \max(0, C_{u0} - \varepsilon)$
$s_{u1} = \mathrm{sgn}(u_1^*)$
$C_{u1} = |u_1^*/q_1|$
$C_{u1} \leftarrow \max(0, C_{u1} - \varepsilon)$
$s_{u6} = \mathrm{sgn}(u_6^*)$
$C_{u6} = |u_6^*/q_6|$
$C_{u6} \leftarrow \max(0, C_{u6} - \varepsilon)$
Same Processing as u*
$D_0$ (from Luma)
$D_1$ (from Luma)
$D_6$
$C_{u0} \leftarrow \frac{C_{u0}}{1+m_c D_0}$
$C_{u0} \leftarrow \frac{s_{u0} C_{u0}^{\beta_c}}{a_c C_{u0} + c_c}$
$C_{u1} \leftarrow \frac{C_{u1}}{1+m_c D_1}$
$C_{u1} \leftarrow \frac{s_{u1} C_{u1}^{\beta_c}}{a_c C_{u1} + c_c}$
$C_{u6} \leftarrow \frac{C_{u6}}{1+m_c D_6}$
$C_{u6} \leftarrow \frac{s_{u6} C_{u6}^{\beta_c}}{a_c C_{u6} + c_c}$
$C_{u0}$
$C_{u1}$
$C_{u6}$
$C_{v0}$
$C_{v6}$
Chroma JND map
$C_{u0}^{ref}$
$C_{u1}^{ref}$
$C_{u6}^{ref}$
$C_{v0}^{ref}$
$C_{v6}^{ref}$
J.144.APPX.II_FII.16

NOTE – $D_3$, ... , $D_6$ are computed by successively filtering and downsampling $D_2$ (from Luma)

**Figure II.16 – Chroma processing detail[23]**

### II.5.3.2 Chroma temporal processing

For each resolution level i, perform a four-field average of the $u_i$ images, and also of the $v_i$ images, with tap weights (0.25, 0.25, 0.25, 0.25), i.e., let

$$u_i \leftarrow \frac{1}{4}\sum_{j=0}^{3} u_i^j \qquad v_i \leftarrow \frac{1}{4}\sum_{j=0}^{3} v_i^j \tag{II-30}$$

where j is the field index.

This step reflects the inherent low-pass temporal filtering of the color channels, and replaces the early-late processing of the temporal luminance channel.

### II.5.3.3 Chroma spatial filtering

Apply a non-oriented Laplacian spatial filter to each of the $u_i$ and $v_i$ images. The filter used in each case is the following 3×3 kernel:

$$1/4\begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{II-31}$$

[23] Step labels are as shown to maintain continuity with the labelling of steps in luma processing.

chosen to have zero total weight and to respond with a maximum strength of 1 to any straight edge between two uniform areas with unit value difference between them. (The maximum response is attained by a horizontal or vertical edge.) This renders the $u_i$ and $v_i$ images into maps of chroma difference, evaluated in uniform-color-space (JND) units.

### II.5.3.4 Chroma contrast computation

Adopt directly the $u_i$ and $v_i$ images from step D as the chroma contrast pyramids, to be interpreted analogously with the Michelson contrasts computed by step E of the luminance model. In an analogy with luminance contrasts, chroma contrasts are computed via intra-image comparisons affected by Laplacian pyramids. Just as the Laplacian difference divided by a spatial average represents the Michelson contrast, which via Weber's law assumes a constant value at the 1-JND level (detection threshold), the Laplacian pyramid operating on $u_i$ and $v_i$ has a 1-JND interpretation. As was the case in the luma model, this interpretation must be modified in the course of calibration. The modification reflects the interaction of all parts of the model, and the fact that stimuli eliciting the 1-JND response are not simple in terms of the model.

Next, level-by-level, divide the contrast pyramid images by seven constants $q_i$ (i= 0,...,6) whose values are determined by calibration. These constants are analogous to the quantities $w_i$ (i = 0,1,2,3) in the luma model.

Compute the clipped absolute values of all the $u_i$ and v* contrasts [where clip(x) = max(0, x – ε)], where ε = 0.75. Preserve the algebraic signs until step H and then re-attach these signs. This prevents the possibility of recording 0 JNDs between two different images because of the ambiguity of the sign loss in the absolute-value operation. The results are two chroma contrast pyramids $C_u$, $C_v$.

### II.5.3.5 Chroma contrast masking

Adopt the denominator pyramid levels $D_m$ (m = 0, 1, 2) directly from step G of the luminance model, without further alteration. For levels 3, ..., 6, perform sequential filtering and down-sampling of $D_2$ using the same method as in the luma processing, but without adding new terms. These $D_m$ values are used in step H in the spirit of perturbation theory. Because luminance effects are expected to predominate over chroma effects in most cases, the chroma model can be viewed as a first-order perturbation on the luminance model. Therefore, the effects of luminance (the $D_m$) are modelled as masking chroma, but not the reverse.

Use the luminance-channel denominator pyramid $D_m$ and the same functional form that is used for the luminance transducer to mask the chroma contrast pyramids, for all pyramid levels m = 0, ..., 6:

$$C_{um} \leftarrow \frac{s_{um} z_{um}^{\beta c}}{a_c C_{um} + c_c} \tag{II-32}$$

where $z_{um} = \dfrac{C_{um}}{(1 + m_c D_i)}$

and $D_i$ is a filtered and down-sampled version of $D_2$ when i > 2. Similarly,

$$C_{vm} \leftarrow \frac{s_{vm} z_{vm}^{\beta c}}{a_c C_{vm} + c_c} \tag{II-33}$$

where $z_{vm} = \dfrac{C_{vm}}{(1 + m_c D_i)}$

Note that the algebraic sign removed in step F has been reattached through the factors $s_{um}$ and $s_{vm}$. This produces masked contrast pyramids for $u_i$ and $v_i$. Calibration determines the values $a_c$, $c_c$, $\beta_c$, $m_c$ and $m_f$.

### II.5.3.6 Chroma JND map construction

The construction of the chroma JND map proceeds completely analogously with the construction of the luma JND map (see II.5.2.6). In this case, the procedure applies to all the masked-contrast chroma images generated by step H above (see Figure II.16).

- images $C_{u0}$, $C_{v0}$, ..., $C_{u6}$, $C_{v6}$
- corresponding images derived from the reference sequence (denoted with superscript $^{ref}$ in Figure II.12).

The first three steps in the following process apply to the above images separately. In discussing them, we denote by X any of these images derived from the test sequence, and by $X^{ref}$ the corresponding image derived from the reference sequence. Given this notation, here are the steps:

- Separate image X into two half-wave-rectified images, one for positive contrasts and the other for negative contrasts. In the positive-contrast image (called $X_+$), the signs from the X contrast (separately stored at stage E) are used to assign zeros to all pixels in $X_+$ that have negative contrasts. The opposite happens in the negative-contrast image $X_–$.
- For each image $X_+$ and $X_–$, perform the local pooling operation suggested by psychophysics by convolving the image with the kernel 0.5(1,2,1) both horizontally and vertically. Then, down-sample the resulting images by a factor of 2 in each direction, to remove redundancy resulting from pooling.
- Presuming that the same processing as was done for X has been done for the corresponding reference image $X^{ref}$, compute pixel-by-pixel the absolute-difference images $|X_+ – X_+^{ref}|$ and $|X_– – X_–^{ref}|$. The resulting images are JND maps.

After this process has been completed for all pairs X, $X^{ref}$, repeatedly up-sample, filter, and add or max all the images to the level required to compute summary measures. This is done as follows:

- Initialize a running-sum image to contain the sum of the Q'th powers of the level-6 images derived from $C_{u6}$, $C_{u6}^{ref}$, $C_{v6}$, and $C_{v6}^{ref}$ . Here, Q = 2.

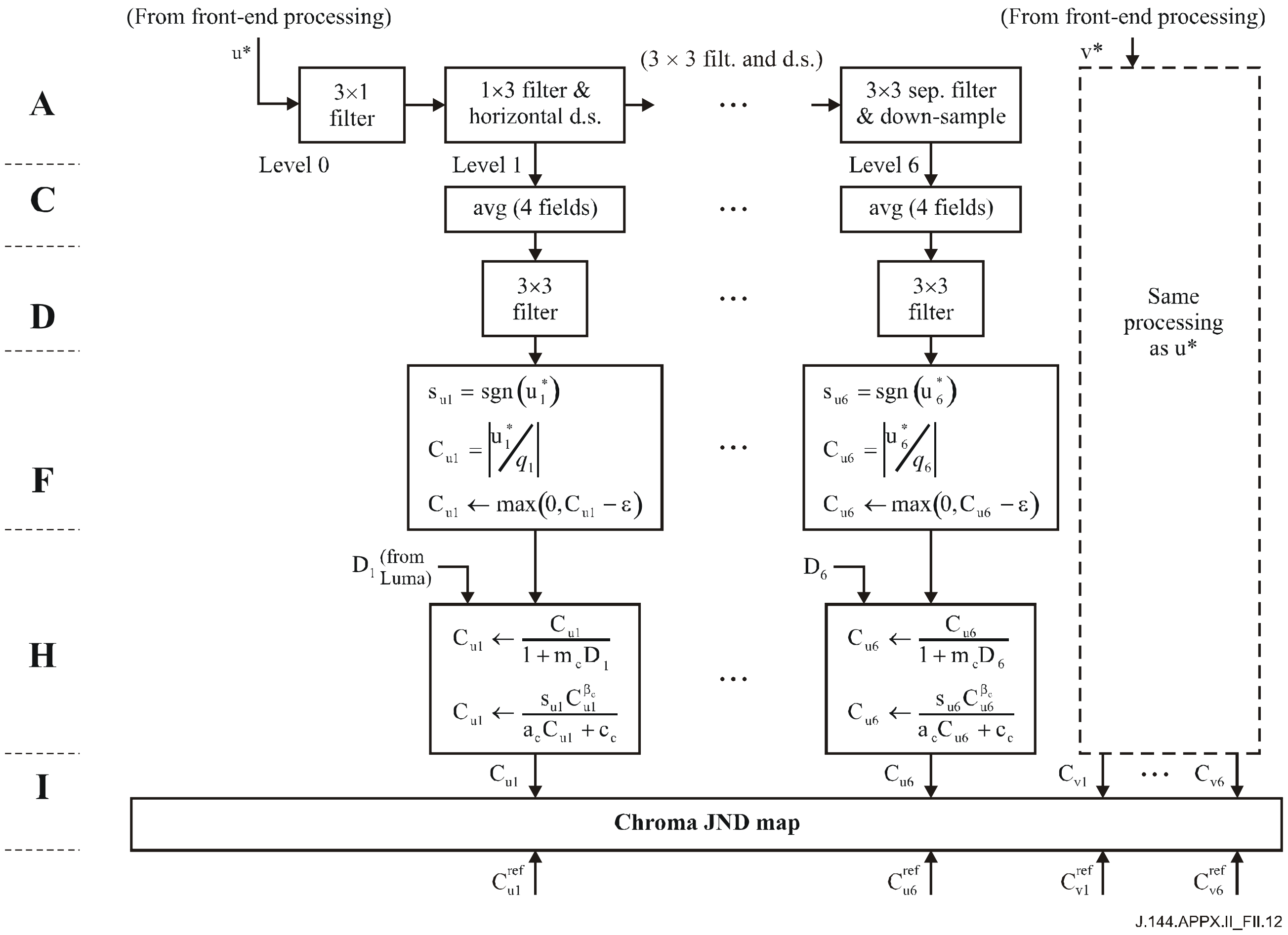
Then perform the following two steps for pyramid-level m starting at 5 and reducing to 0 in steps of 1:

- Up-sample/filter the running-sum image to comprise a level-m image;
- Update the running-sum image by adding to it the Q'th powers of the level-m images derived from $C_{um}$, $C_{um}^{ref}$, $C_{vm}$, and $C_{vm}^{ref}$.

As in luma processing, after these operations, the resulting image is half the resolution of the original. Note that each pyramid-level index in this clause refers to the pyramid level from which it was originally derived, which is twice the resolution of that associated with that level after filtering/down-sampling. The level-0 image is sent directly to summary processing (see Figure II.9 and II.5.4).

### II.5.3.7 Half-height chroma processing

If the half-height images are to be passed through directly without zero filling to the true image height, then the above chroma processing must be modified to reflect that the inherent vertical resolution is only half the inherent horizontal resolution. Figure II.17 summarizes chroma processing for the half-height algorithm.


(From front-end processing)
u*
(From front-end processing)
v*
A
C
D
F
H
I
3×1 filter
1×3 filter & horizontal d.s.
(3 × 3 filt. and d.s.)
3×3 sep. filter & down-sample
Level 0
Level 1
Level 6
avg (4 fields)
avg (4 fields)
3×3 filter
3×3 filter
Same processing as u*
$s_{u1} = \mathrm{sgn}(u_1^*)$
$C_{u1} = \left| u_1^* / q_1 \right|$
$C_{u1} \leftarrow \max(0, C_{u1} - \varepsilon)$
$s_{u6} = \mathrm{sgn}(u_6^*)$
$C_{u6} = \left| u_6^* / q_6 \right|$
$C_{u6} \leftarrow \max(0, C_{u6} - \varepsilon)$
$D_1$ (from Luma)
$D_6$
$C_{u1} \leftarrow \frac{C_{u1}}{1 + m_c D_1}$
$C_{u1} \leftarrow \frac{s_{u1} C_{u1}^{\beta_c}}{a_c C_{u1} + c_c}$
$C_{u6} \leftarrow \frac{C_{u6}}{1 + m_c D_6}$
$C_{u6} \leftarrow \frac{s_{u6} C_{u6}^{\beta_c}}{a_c C_{u6} + c_c}$
$C_{u1}$
$C_{u6}$
$C_{v1}$
$C_{v6}$
Chroma JND map
$C_{u1}^{ref}$
$C_{u6}^{ref}$
$C_{v1}^{ref}$
$C_{v6}^{ref}$
J.144.APPX.II_FII.12

NOTE – $D_3$, ... , $D_6$ are computed by successively filtering and downsampling $D_2$ (from Luma)

**Figure II.17 – Chroma processing detail (half-height)**

Comparison between this diagram and the corresponding diagram for full-height interlace (Figure II.11) reveals the following basic changes:

1) The highest-resolution chroma channels, $u_0$* and $v_0$*, are eliminated. Because chroma sensitivity is low at high spatial frequencies, nothing significant is lost in this step.

2) In step A, to produce the next-highest resolution chroma images $u_1$* and $v_1$*, a low-pass "Kell" filter kernel with weights (1/8, 3/4, 1/8) is applied vertically (i.e., along columns). This operation corresponds to the joint filtering of the assumed de-interlace filter, together with the filtering performed by the vertical components of the 3×3 filters in step D of the full-height algorithm. The resulting vertically filtered images are then horizontally filtered with a 1×3 filter kernel 0.25(1,2,1). This filtering of u* and v* images makes the half-height images isotropic in resolution. The resolution is that of full-height pyramid-level 1.

3) Because the Q-norm stream is fully accumulated at pyramid level 1 in the chroma model, the chroma JND map for summary measures is only half the size (both horizontally and vertically) as the fully accumulated luma map. Prior to combining the chroma and luma maps to produce the total-JND map, the chroma map must first be brought to the same resolution as the luma map. To achieve this goal, an up-sample followed by 3×3 filter is performed to produce the chroma JND map for summary measures.

### II.5.4 Output summaries

As discussed in the previous clauses, the luma and chroma JND maps passed to the output summary step are JND images, and are represented at half the resolution of the original image. This exploits the redundancy inherent in having performed pooling at each masked-contrast stage.

Next, the luma and chroma JND maps $JND_L$ and $JND_C$ are combined into a total-field JND map, $JND_T$. The combination rule is a Minkowski Q-norm ($Q = 2$), in analogy with the combination of channels to produce the maps $JND_L$ and $JND_C$:

$$JND_T\,(i,j) = [JND_L\,(i,j)^Q + JND_C\,(i,j)^Q]^{1/Q} \qquad \text{(II-34)}$$

Then, each of the three JND maps (luma, chroma, and combined luma-chroma) is reduced to a single-number summary, called a Picture Quality Rating (PQR) value. Single number summaries are computed by Minkowski Q-norm, as summarized below.

Each of the half-resolution JND images (three for each field: luma, chroma, and total-field) is reduced to a single performance measure called a PQR by using the formulas:

$$PQR_{luma} = \left[\left(\frac{1}{N_p}\right)\sum_{i,j} JND_L(i,j)^Q\right]^{1/Q}$$

$$PQR_{chroma} = \left[\left(\frac{1}{N_p}\right)\sum_{i,j} JND_C(i,j)^Q\right]^{1/Q} \qquad \text{(II-35)}$$

$$PQR_{total} = \left[\left(\frac{1}{N_p}\right)\sum_{i,j} JND_T(i,j)^Q\right]^{1/Q}$$

where the summation is over all pixels in the JND map, $Q = 4$, and $N_p$ is the number of pixels in the map. In this way, three summary measures corresponding respectively to $JND_L$, $JND_C$, and $JND_T$ are computed for each field k in a video sequence.

From N single-field $PQR_{field}$ values in a video sequence[24], a single performance measure $PQR_N$ is computed through the following Minkowski Q-norm:

$$PQR_N = \left[\left(\frac{1}{N}\right)\sum_{k} PQR_{field}(k)^Q\right]^{1/Q} \qquad \text{(II-36)}$$

NOTE – Subjective rating data are noisy and unreliable for short video sequences (less than 1/2 second, or 15 frames). PQR estimates will correlate poorly with subjective ratings for short sequences.

### II.5.5 Image border processing

To minimize cropping and, hence, avoid border artifacts, the PQR method replaces the screen border by a gray bezel of infinite extent, but does so without enhancing the real image size by more than six pixels on a side. Use of this "virtual-bezel" eliminates the need to crop the JND map to avoid border artifacts. The infinite gray bezel models viewing conditions and hence can be considered non-artifactual. With this interpretation, the whole JND map is uncontaminated by artifacts.

This clause describes the border algorithm. In the following discussion, an image that has been padded with 6 pixels on all sides is referred to as a *padded image*, and an unpadded image or its locus within a padded image as the *image proper*.

[24] Here, *field* denotes a PQR value in any one of the three sequences luma, chroma, or total.

### II.5.5.1 Color of the bezel

Since image operations are local, the virtually infinite bezel can be implemented efficiently. Sufficiently far outside the *image proper*, an infinite bezel results in a set of identical, constant values at any given model stage. The effect of image operations, e.g., filtering, performed in this constant region can be computed a priori. Thus, a narrow border (6 pixels in the current implementation) can provide the proper transition from the *image proper* to the infinite bezel.

At the input, the bezel is given the values Y' = 90, U' = V' = 0. (The value of Y' = 90 corresponds to half the BT.500-11 background value of 15% of the maximum screen luminance.) However, the bezel is not needed until after front-end processing, because spatial interactions that extend beyond the image borders do not occur until after this stage. In the luma channel, no borders (and, hence, no bezel values) are appended to images until after luma compression. In the chroma channel, borders are appended after front-end processing.

In the luma channel, the first bezel value after luma compression is

$$first\_luma\_bezel = \left[ L_{\max} \left( \frac{90}{255} \right)^{\gamma} \right]^{m} + L_{d}^{m} \qquad \text{(II-37)}$$

In the u* and v* channels, the first bezel values are both 0.

These values are propagated through subsequent stages of the model in three ways:

1) Pixel-by-pixel functions operate on old bezel values to produce new bezel values. For example, the bezel value resulting from the power function (equation II-23) is

$$bezel\_out = (bezel\_in)^{\mathrm{R}} \qquad \text{(II-38)}$$

2) 3×3 spatial filters whose rows and columns sum to P, set the output bezel value to the input bezel times P.

3) Contrast function numerators and four-field time filters (which have tap sums of zero), set the output bezel value to 0.

At the contrast stage, and subsequently, the bezel is given the value 0 in luma and chroma channels: the logical consequence of operating with a zero-sum linear kernel on a spatially constant array.

The 3 categories above introduce some, but by no means all, of the complexities required to understand and implement the border algorithm. In the following clause, the next level of detail is introduced.

### II.5.5.2 Integrating image and bezel

Starting with the pyramid stages of the model, borders need to be supplied. The first border operation on an N-by-M input image is to pad the image with 6 pixels (on all sides) with the appropriate bezel value (first_luma_bezel for the compressed luma image, and 0 for u* and v* images). The padded image has dimensions (N + 12) × (M + 12). For the k$^{\text{th}}$ pyramid level (where k can range from 0 to 7)[25], the padded image has dimensions ([N/2$^{k}$] + 12) × ([M/2$^{k}$] + 12), where "[*x*]" denotes the greatest integer in *x*.

Images at all pyramid levels are registered to each other at the upper left hand corner of the *image proper*. Indices of the *image proper* run from 0 = y = height, 0 = x = width. The upper left hand corner of the *image proper* always has indices (0,0). Indices of bezel pixels take on height and width values less than 0. For example, the upper left hand bezel pixel is (–6, –6). Looking along the x-dimension starting at the left hand edge for an image of width w (image plus bezel width w+12),

[25] Level 7 is required only for chroma map construction.

the bezel pixels are indexed by $x$ = (–6,–5,...,–1) the real image is indexed (0,1,...,w–1) and the right hand bezel indices span (w,w+1,...,w+5).

Given a padded image, there are four things that can happen depending on the subsequent stage of processing. In describing these operations below, single image lines are used to summarize spatial processing (with the understanding that the analogous events take place in the vertical direction).

a) *For pixel-by-pixel operations.* When the next operation is to operate pixel-by-pixel (e.g., with a non-linearity), the padded image is simply passed through the operation, and the output-image dimensions are the same as the input-image dimensions. The same occurs when the operation is between corresponding pixels in different fields or different color-bands.

b) *For 3×3 spatial filters.* Suppose (in one dimension) the unpadded input image has dimension $N_k$. Then the padded input image has dimension $N_k + 12$, and the padded output image has dimension $N_k + 12$ as well. The output bezel value is first computed (e.g., as in equation II-37) and written into at least those bezel pixels not otherwise filled by the subsequent image operation. Then, starting 1 pixel away from the left edge of the padded input image, the 3×3 kernel starts operating on the input image and over-writing the bezel values of the output image, stopping 1 pixel away from the right (or bottom) edge of the image (where the original bezel value survives). The pre-written bezel value makes it unnecessary for the kernel operation ever to go outside the original (padded) image to compute these values.

c) *For filtering and down-sampling in REDUCE.* Given an input padded image with dimension $N_k + 12$, an output array is allocated with dimension $[N_k/2] + 12$. The bezel value (computed, e.g., as in equation II-37) is written into at least those bezel pixels not otherwise filled by the subsequent filter and downsample operation. Then, the input image is filtered according to b above, but the filter is applied at pixels –4, –2, 0, 2, 4, until the input image is exhausted, and the output values are written into consecutive pixels –2, –1, 0, 1, 2, ..., until there is no further place for them in the output image. Note that the position of pixel 0 in the new image is 7 pixels from the left end of the new image. The last-pixel application of the filter takes input pixel $N_k + 3$ to output pixel $[N_k/2] + 2$ if $N_k$ is odd, and it takes input pixel $N_k + 4$ to output pixel $[N_k/2] + 2$ if $N_k$ is even. (Here, the filter's input pixel is defined as the pixel corresponding to the centre of the 3-pixel kernel.)

The following are four simplified examples of border processing in REDUCE. In each case, pixels are labelled consecutively, in brackets and bold for the *image proper*, and underlined for the pre-written bezel.

EXAMPLE 1: $N_k$ = 3. (Odd size in, odd size out.)

In: –6 –5 –4 –3 –2 –1 **[0 1 2]** 3 4 5 6 7 8

Out: $\underline{-6\quad -5\quad -4\quad -3}$ –2 –1 **[0]** 1 2 3 $\underline{4\quad 5\quad 6}$

EXAMPLE 2: $N_k$ = 4. (Even size in, even size out.)

In: –6 –5 –4 –3 –2 –1 **[0 1 2 3]** 4 5 6 7 8 9

Out: $\underline{-6\quad -5\quad -4\quad -3}$ –2 –1 **[0 1]** 2 3 4 $\underline{5\quad 6\quad 7}$

EXAMPLE 3: $N_k$ = 5. (Odd size in, even size out.)

In: –6 –5 –4 –3 –2 –1 **[0 1 2 3 4]** 5 6 7 8 9 10

Out: $\underline{-6\quad -5\quad -4\quad -3}$ –2 –1 **[0 1]** 2 3 4 $\underline{5\quad 6\quad 7}$

EXAMPLE 4: $N_k$ = 6. (Even size in, odd size out.)

In: –6 –5 –4 –3 –2 –1 **[0 1 2 3 4 5]** 6 7 8 9 10 11

Out: $\underline{-6\quad -5\quad -4\quad -3}$ –2 –1 **[0 1 2]** 3 4 5 $\underline{6\quad 7\quad 8}$

d) *For up-sampling and filtering in EXPAND.* Given an input padded image at level k+1 with dimension $N_{k+1} + 12$, an output array at level k with dimension $N_k + 12$ is allocated, and initialized to 0. (Note that $N_{k+1}$ was predefined as $[N_k/2]$.) Next, the input pixels –2, –1, 0, 1, ... are inserted into output pixels –4, –2, 0, 2, 4, ... . Then, the filtering operation in b above is performed on the resulting image. Finally, the bezel value at level k is computed, e.g., by use of equation II-37, and written into all the outermost 3 pixels on all sides of the output image. Note that the position of pixel 0 in the new image is 7 pixels from the left end of the new image. The last-pixel application of the filter takes input pixel $[N_k/2] + 2$ to output pixel $N_k + 3$ if $N_k$ is odd, and it takes input pixel $[N_k/2] + 2$ to output pixel $N_k + 4$ if $N_k$ is even. (Here again, the filter's input pixel is defined as the pixel corresponding to the centre of the 3-pixel kernel.)

The following are four simplified examples of border processing in EXPAND. In each case, pixels are labelled consecutively, in brackets and bold for the *image proper*, and underlined for the post-written bezel.

EXAMPLE 1: $N_k = 3$. (Odd size in, odd size out.)

In: –6 –5 –4 –3 –2 –1 **[0]** 1 2 3 4 5 6

Out: –6 –5 –4 –3 –2 –1 **[0 1 2]** 3 4 5 6 7 8

EXAMPLE 2: $N_k = 4$. (Even size in, even size out.)

Out: –6 –5 –4 –3 –2 –1 **[0 1]** 2 3 4 5 6 7

In: –6 –5 –4 –3 –2 –1 **[0 1 2 3]** 4 5 6 7 8 9

EXAMPLE 3: $N_k = 5$. (Even size in, odd size out.)

Out: –6 –5 –4 –3 –2 –1 **[0 1]** 2 3 4 5 6 7

In: –6 –5 –4 –3 –2 –1 **[0 1 2 3 4]** 5 6 7 8 9 10

EXAMPLE 4: $N_k = 6$. (Odd size in, even size out.)

Out: –6 –5 –4 –3 –2 –1 **[0 1 2]** 3 4 5 6 7 8

In: –6 –5 –4 –3 –2 –1 **[0 1 2 3 4 5]** 6 7 8 9 10 11

From these illustrative examples, it can be seen that over-writing the bezel has no effect when the EXPAND process is repeated for successive levels.

# Appendix II – Attachment 1

## Bibliography

- ITU-R Recommendation BT.500-11 (2002), *Methodology for the subjective assessment of the quality of television pictures*.
- USA Standards Committee T1[*], Technical Report T1.TR.73-2001: Video Normalization Methods Applicable to Objective Video Quality Metrics Utilizing a Full Reference Technique.
- Burt PJ, and Adelson EH (1983), *The laplacian pyramid as a compact image code*. IEEE Trans Comm 31-532-540. <http://www.citeseer.ifi.unizh.ch/burt83laplacian.html>

[*] T1 standards are maintained since November 2003 by ATIS.

# Appendix II – Attachment 2

## Test factors, coding technologies and applications

See the VQEG final report (ITU-T Tutorial) for further details regarding the data in these tables. All data is for the 525-line system.

**Table II – Att. 2.1 – Test factors, coding technologies and applications for which the PQR method has shown the accuracy specified in II.1.3.4**

| Bit rate | Res | Method | Comments |
|---|---|---|---|
| 2 Mbit/s | ¾ resolution | mp@ml | This is horizontal resolution reduction only |
| 2 Mbit/s | ¾ resolution | sp@ml | |
| 4.5 Mbit/s | | mp@ml | With errors |
| 3 Mbit/s | | mp@ml | With errors |
| 4.5 Mbit/s | | mp@ml | |
| 3 Mbit/s | | mp@ml | |
| 4.5 Mbit/s | | mp@ml | Composite NTSC and/or PAL |
| 6 Mbit/s | | mp@ml | |
| 8 Mbit/s | | mp@ml | Composite NTSC and/or PAL |
| 8 & 4.5 Mbit/s | | mp@ml | Two codecs concatenated |
| 19 Mbit/s – NTSC-<br>19 Mbit/s – NTSC-<br>12 Mbit/s | | 422p@ml | NTSC 3 generations |
| 50-50-… -50 Mbit/s | | 422p@ml | 7th generation with shift / I frame |
| 19-19-12 Mbit/s | | 422p@ml | 3rd generation |
| n/a | | n/a | Multi-generation Betacam with drop-out compensation (4 or 5, composite/component) |

**Table II – Att. 2.2 – Test factors, coding technologies and applications for which the PQR method has not shown the accuracy specified in II.1.3.4**

| Bit rate | Res | Method | Comments |
|---|---|---|---|
| 1.5 Mbit/s | CIF | H.263 | Full screen |
| 768 kbit/s | CIF | H.263 | Full screen |
| Other | | | The PQR method specified in Appendix I is not appropriate for video conferencing applications that repeat fields or do not meet the latency and delay requirements of the video classes. In addition, the PQR method is only applicable to typical broadcast transmission systems with very low-error rates such as those included in the VQEG tests. |

### Table II – Att. 2.3 – Test sequences used to determine test factors, coding technologies and applications for which the PQR method has shown the accuracy specified in II.1.3.4

| Sequence | Characteristics |
|---|---|
| Baloon-pops | film, saturated color, movement |
| NewYork 2 | masking effect, movement |
| Mobile&Calendar | available in both formats, color, movement |
| Betes_pas_betes | color, synthetic, movement, scene cut |
| Le_point | color, transparency, movement in all the directions |
| Autumn_leaves | color, landscape, zooming, water fall movement |
| Football | color, movement |
| Sailboat | almost still |
| Susie | skin color |
| Tempete | color, movement |

# Appendix II – Attachment 3

## Classification of errors

Classification errors are one way to evaluate the effectiveness of a Video Quality Metric (VQM). This appendix discusses the meaning of the classification errors, in terms of the plots of subjective z score versus delta-VQM described in the main body of the Recommendation. For the following description, we are using the common [0, 1] scale for both the subjective and objective scores. For this common [0, 1] scale, "0" represents no impairment and "1" represents maximum impairment.

For any subjective test one can set a threshold Δz, that defines when two data points (A, B) are statistically equivalent and when they are statistically distinguishable[26]. Once this has been done, the subjective test results allow one to place each pair of data points (A, B) into one of three categories:

| | | |
|---|---|---|
| $\Delta z_{AB} < -\Delta z$ | → A is better than B | → Bs |
| $-\Delta z \leq \Delta z_{AB} \leq \Delta z$ | → A is same as B | → Es |
| $\Delta z < \Delta z_{AB}$ | → A is worse than B | → Ws |

The abbreviations for the three categories (Bs, Es, and Ws) denote subjectively better, subjectively equivalent, and subjectively worse, respectively.

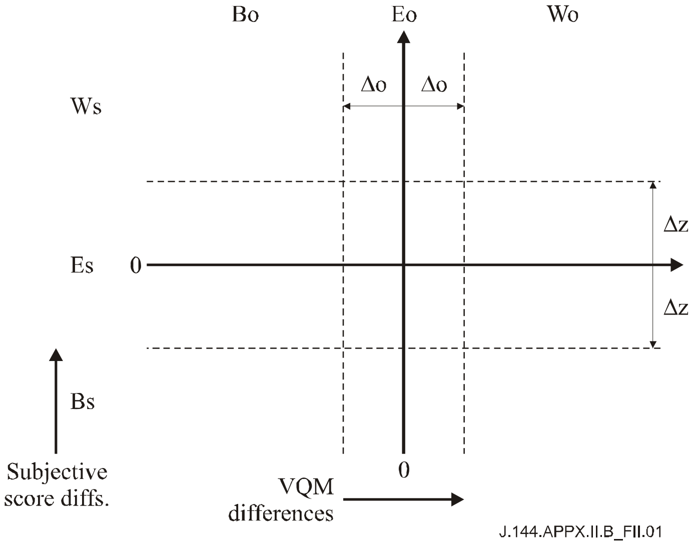
Now consider a similar threshold for VQM values, Δo:

| | | |
|---|---|---|
| VQM(A)– VQM (B) < – Δo | → A is better than B | → Bo |
| –Δo ≤ VQM (A) – VQM (B) ≤ Δo | → A is same as B | → Eo |
| Δo < VQM (A) – VQM (B) | → A is worse than B | → Wo |

The abbreviations for the three categories (Bo, Eo, and Wo) denote objectively better, objectively equivalent, and objectively worse, respectively.

[26] The data points A and B actually represent sets of observations of two SRC/HRC combinations. As discussed in the main body of the Recommendation, the quantity $\Delta z_{AB}$ is the difference in the means of A and B $(\hat{S}_{A\bullet} - \hat{S}_{B\bullet})$, divided by the inferred standard deviation $\sqrt{(V_A / N_A + V_B / N_B)}$, where $V_A$ is the variance of scores from situation A, and $N_A$ is the number of observations from situation A, etc.

Since each pair of data points undergoes a three-way classification by the subjective test and a separate three-way classification by the VQM, there are nine possible outcomes. These nine outcome spaces are illustrated graphically below by the broken lines in the two-dimensional space of subjective-score difference versus VQM difference:

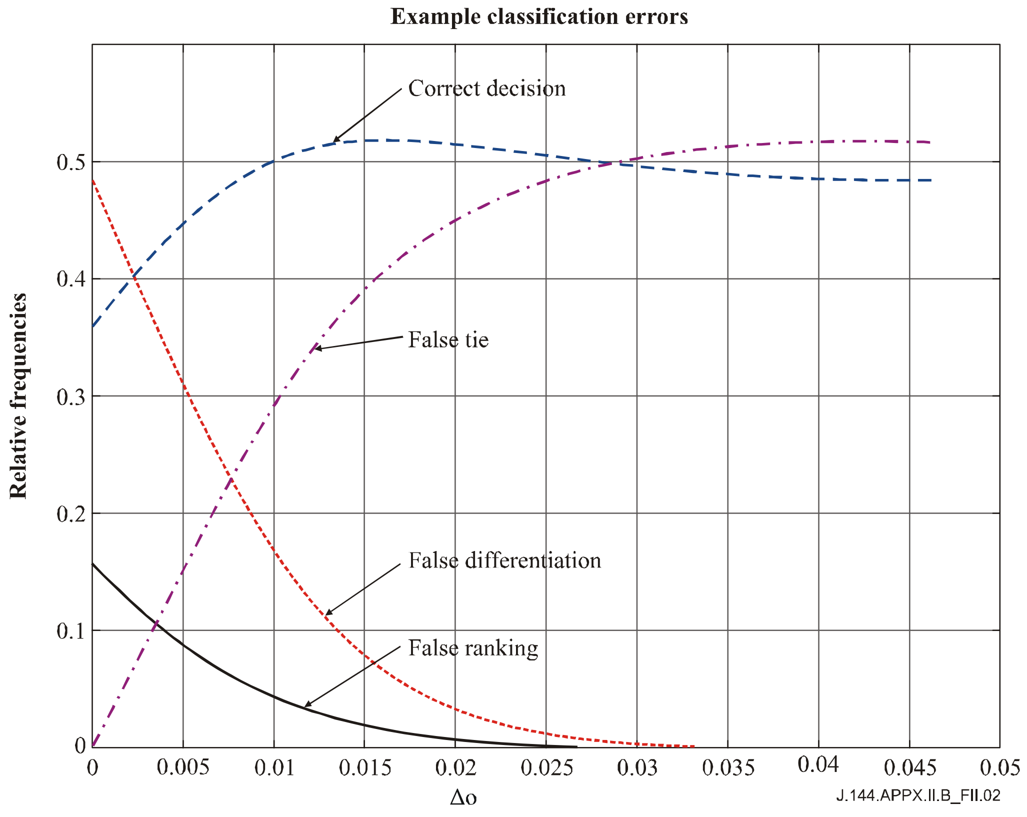


In the table below, we label each of these nine outcomes with an eye towards answering the question "How does the VQM-based 3-way classification compare with the subjective test-based 3-way classification?"

| | **Bs** | **Es** | **Ws** |
|---|---|---|---|
| **Wo** | False Ranking | False Differentiation | Correct Decision |
| **Eo** | False Tie | Correct Decision | False Tie |
| **Bo** | Correct Decision | False Differentiation | False Ranking |

Note that for three of the outcomes, the VQM classification agrees with the subjective test classification. These three outcomes are labelled "Correct Decision." The six remaining outcomes correspond to three different types of errors that can arise when using a VQM. The false tie is probably the least offensive error. This occurs when the subjective test says two data points are different, but the VQM says they are the same. A false differentiation is usually more offensive. This occurs when the subjective test says two data points are the same, but the VQM says they are different. The false ranking would generally be the most offensive error. In false ranking, the subjective test says A is better than B, but the VQM says B is better than A.

For any subjective test and any VQM, we can form all possible distinct pairs of data points and count the number of pairs that fall into each of the four distinct outcome categories: correct decision, false tie, false differentiation, and false ranking. We can then normalize by the total number of distinct pairs and report relative frequencies for these four outcome categories. In general, these results will be functions of both Δs and Δo. Example results for a fictitious VQM are given in the graph below. Δz was selected to give an estimated 95% confidence in the subjective classifications and Δo is the free parameter on the x-axis of the graph.

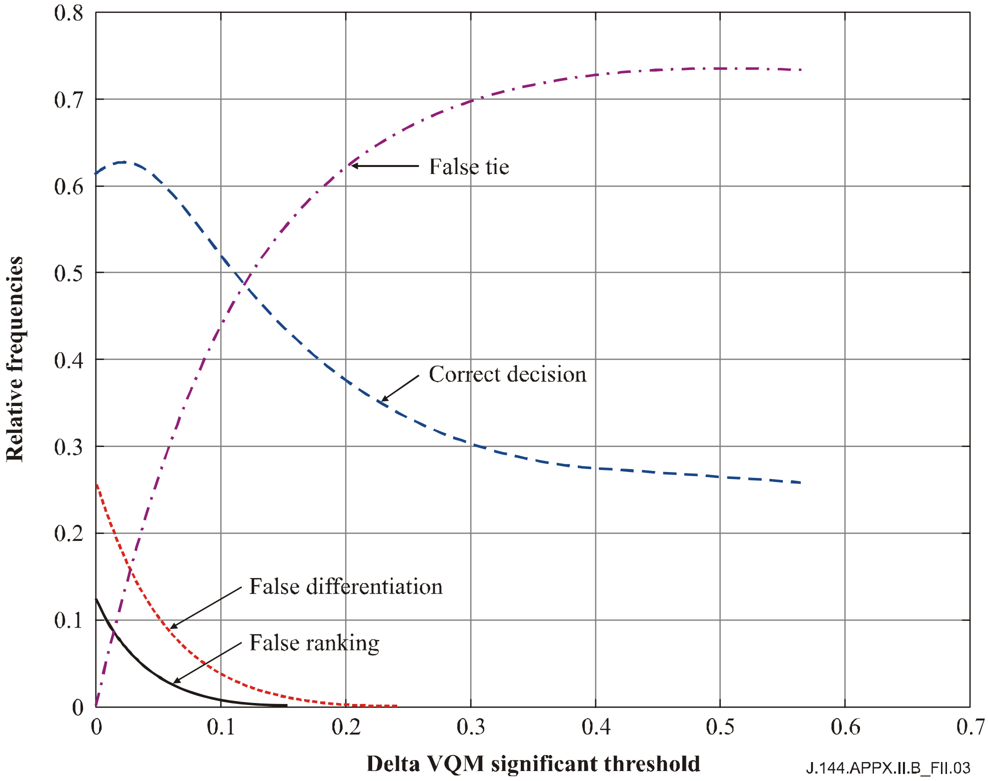


Note that as Δo is increased, the VQM will declare more and more pairs of data points as equivalent. This reduces the occurrences of false differentiations and false rankings, but increases the occurrence of false ties. As Δo goes to 0.05, the false-tie rate tends towards 0.52. At this point, the VQM is declaring all pairs to be equivalent and, in doing so, the VQM is wrong 52% of the time, and correct 48% of the time. This is consistent with the fact that in this test, 48% of the pairs of data points were declared equivalent by the subjective test. One might use a graph like this to select an appropriate value of Δo. For example, one might select Δo to maximize the probability of making correct decisions, or one might select Δo to minimize some weighted sum of the error relative frequencies.

In the code that generated the above figure (part of the MATLAB code in Annex B), the threshold used for the subjective test is subj_th. The threshold used for the ΔVQM is vqm_th and this is left as a free parameter. The code plots the frequency of occurrence for the three different kinds of errors and for no error vs. vqm_th. An optimal value of vqm_th might be one that maximizes the frequency of occurrence of no error, or one that minimizes a cost-weighted sum of the errors. Note that, in general, it is likely that false ties will be the least offensive error, false differentiations will be more offensive, and false rankings will be the worst sort of error.

NOTE – The nine outcomes and the three-by-three grid in (ΔVQM, subjective Z score) space is the most natural way to describe this analysis. This assumes bipolar values for ΔVQM. But the code has already taken the absolute value of ΔVQM (and replaced Z with –Z for all points with negative values of ΔVQM). This does not change the mathematics, but the more natural description of the situation is now 6 outcomes and a 2 by 3 grid. Two correct outcomes (A better than B and A worse than B) have been folded on top of each other. There are still two false tie outcomes, but only one false differentiation outcome and one false ranking outcome.

For the PQR method specified in this appendix the following classification of errors applies.



# SERIES OF ITU-T RECOMMENDATIONS

| | |
|---|---|
| Series A | Organization of the work of ITU-T |
| Series D | General tariff principles |
| Series E | Overall network operation, telephone service, service operation and human factors |
| Series F | Non-telephone telecommunication services |
| Series G | Transmission systems and media, digital systems and networks |
| Series H | Audiovisual and multimedia systems |
| Series I | Integrated services digital network |
| **Series J** | **Cable networks and transmission of television, sound programme and other multimedia signals** |
| Series K | Protection against interference |
| Series L | Construction, installation and protection of cables and other elements of outside plant |
| Series M | Telecommunication management, including TMN and network maintenance |
| Series N | Maintenance: international sound programme and television transmission circuits |
| Series O | Specifications of measuring equipment |
| Series P | Terminals and subjective and objective assessment methods |
| Series Q | Switching and signalling |
| Series R | Telegraph transmission |
| Series S | Telegraph services terminal equipment |
| Series T | Terminals for telematic services |
| Series U | Telegraph switching |
| Series V | Data communication over the telephone network |
| Series X | Data networks, open system communications and security |
| Series Y | Global information infrastructure, Internet protocol aspects and next-generation networks |
| Series Z | Languages and general software aspects for telecommunication systems |

Printed in Switzerland
Geneva, 2009