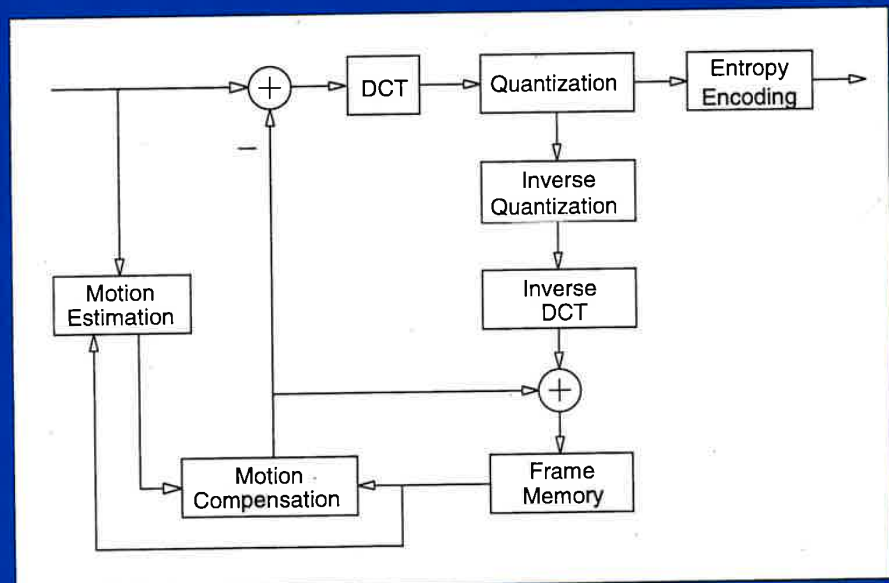


IMAGE AND VIDEO COMPRESSION STANDARDS

Algorithms and Architectures

Vasudev Bhaskaran
Konstantinos Konstantinides



K L U W E R A C A D E M I C P U B L I S H E R S

**IMAGE AND VIDEO
COMPRESSION STANDARDS**
Algorithms and Architectures

**THE KLUWER INTERNATIONAL SERIES
IN ENGINEERING AND COMPUTER SCIENCE**

**IMAGE AND VIDEO
COMPRESSION STANDARDS**
Algorithms and Architectures

by

**Vasudev Bhaskaran
Konstantinos Konstantinides**
Hewlett-Packard Laboratories



KLUWER ACADEMIC PUBLISHERS
Boston / Dordrecht / London

Distributors for North America:

Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061 USA

Distributors for all other countries:

Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS

Library of Congress Cataloging-in-Publication Data

A C.I.P. Catalogue record for this book is available
from the Library of Congress.

Copyright © 1995 by Kluwer Academic Publishers. Third Printing 1996.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, mechanical, photo-copying, recording, or otherwise, without the prior written permission of the publisher, Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, Massachusetts 02061

Printed on acid-free paper.

Printed in the United States of America

CONTENTS

PREFACE	xiii
1 COMPRESSION FUNDAMENTALS	1
1.1 Introduction	1
1.2 Background	2
1.3 Compression Taxonomy	5
1.4 Issues in Compression Method Selection	12
1.5 To Probe Further	14
2 METHODS AND STANDARDS FOR LOSSLESS COMPRESSION	15
2.1 Introduction	15
2.2 Preliminaries	15
2.3 Huffman Encoding	18
2.4 Huffman Decoding	22
2.5 Huffman Codes with Constrained Length	24
2.6 Constrained-Length Huffman Codes - Ad-Hoc Design	31
2.7 Constrained-Length Huffman Codes - The Voorhis Method	33
2.8 Arithmetic Coding	37
2.9 Implementation Issues	42
2.10 Standards for Lossless Compression	43
2.11 To Probe Further	50
3 FUNDAMENTALS OF LOSSY IMAGE COMPRESSION	53
3.1 Introduction	53

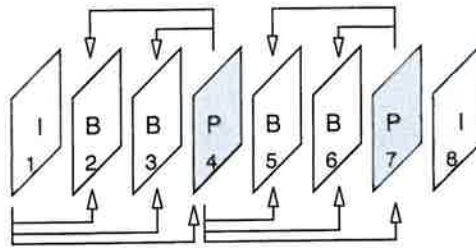


Figure 6.3 Example of inter-dependence among I-, P-, and B-pictures in a video sequence.

6.2.3 Video Encoder

The MPEG standards *do not* define an encoding process. They only specify the syntax of the coded bit stream and the decoding process. Based on these requirements, Figure 6.5 shows the functions that need to be executed by a typical MPEG encoder.

Preprocessing. The encoding process usually begins with some preprocessing. This may include color conversion to YCbCr, format translation (interlaced to progressive), prefiltering, and subsampling. None of these operations is specified in the standards.

Motion estimation and compensation. After preprocessing, the encoder selects the coding type for the input picture. I-pictures require no motion estimation or compensation. Each macroblock is DCT coded, and the DCT coefficients are quantized, coded using a variable-length coder (VLC), and stored into the output buffer. Within each macroblock, processing is performed on 8×8 blocks. In applications requiring constant bit rate, a buffer regulator may adjust the quantization matrix so that the rate of the compressed bit stream is relatively constant. For variable bit-rate applications, the regulator function may not be needed.

The quantized blocks are also inverse quantized (Q^{-1}) and transformed into the spatial domain by an inverse DCT. This operation duplicates the behavior of the decoder and yields a copy of the encoded picture as it will be seen by the decoder. That copy is then stored into local memory and will be used

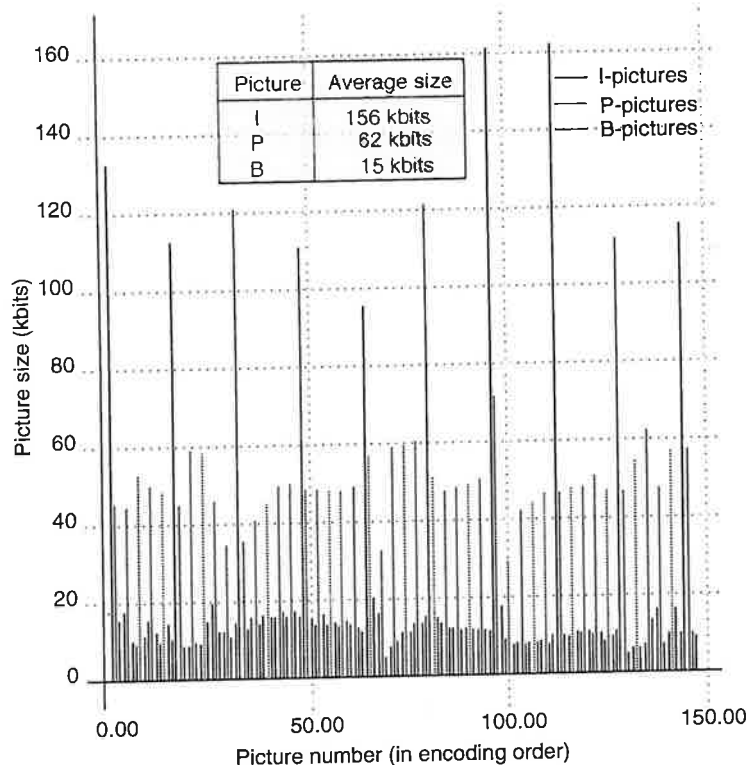


Figure 6.4 Example of bit distribution among I-, P-, and B-pictures in an MPEG-1 coded bit stream. Pictures are coded in IPBBPBBPBBPBBPBB sequence.

for future predictive coding. In effect, this operation allows the encoder to monitor the quality of the transmitted image so that it does not diverge from the original signal. Since the VLC operation is lossless, there is no need to include the VLC unit in the feedback path.

If the input is coded as a P-picture or a B-picture, then the encoder does not code the picture macroblocks directly. Instead it codes the prediction errors. In MPEG terminology, this is referred to as *interframe predictive coding*. For P-pictures, this process is explained in Figure 6.6. For each macroblock in the current picture, the motion estimator yields the coordinates (motion vector)

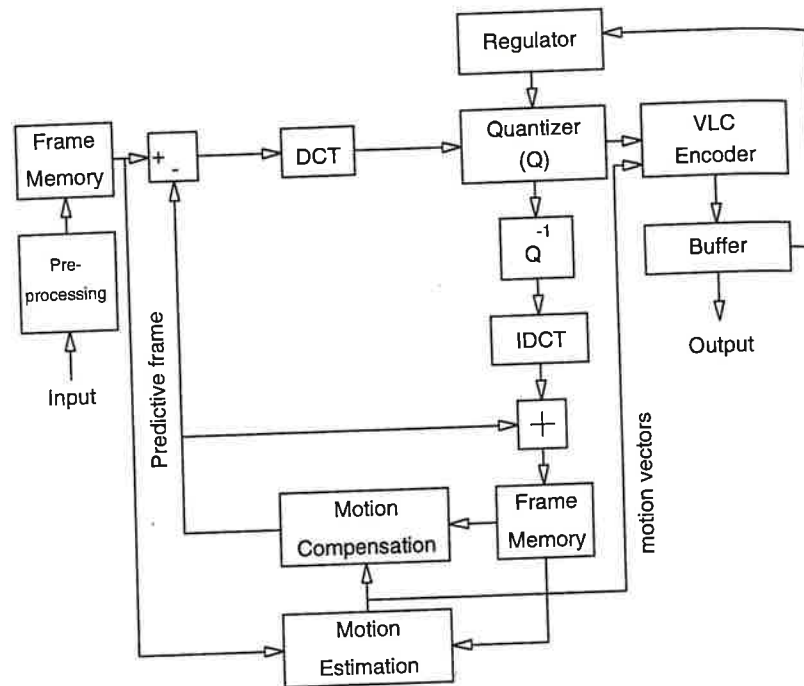


Figure 6.5 Block diagram of an MPEG encoder.

of the macroblock that best matches its characteristics in a search area in the past picture. The two macroblocks are subtracted, and their difference is then DCT coded. Any of the block-based motion estimation methods described in Chapter 4 can be used.

For B-pictures, the motion estimation process is performed twice, once for a past picture and once for a future picture, so as to yield two motion vectors. The encoder can form a prediction error macroblock from either of the two candidate macroblocks or from their average. This process is shown in detail in Figure 6.7. In MPEG terminology, this is referred to as *interframe interpolative coding*. The prediction error is then coded using the block-based DCT. The quantized DCT coefficients of the prediction errors, together with the motion vectors are then multiplexed and coded using the variable-length

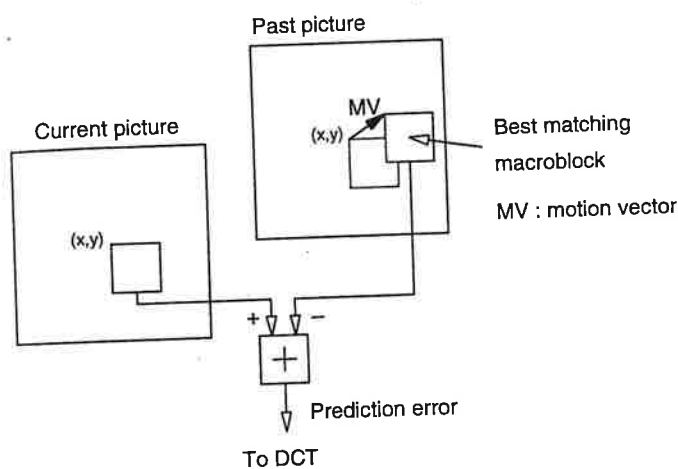


Figure 6.6 Forward motion compensation.

coder. Unlike JPEG, MPEG uses only a Huffman coder for variable-length coding.

Because of the bidirectional prediction, some reordering of the input pictures is needed at the encoder so that they are delivered in the correct order to the decoder. For example, consider the sequence of pictures in Figure 6.3. Since picture p_2 is a B-picture that depends on pictures p_1 and p_4 , it can be encoded only after the input of picture p_4 . After reordering of the input pictures, the correct encoding sequence is $p_1, p_4, p_2, p_3, p_7, p_5, p_6$, and p_8 . This bit stream provides the decoder with all the information needed to decode each received picture; however, the decoder will need to reorder these pictures so that they can be displayed in the correct order.

6.2.4 Video Decoder

Figure 6.8 shows a block diagram of an MPEG video decoder. As expected, the circuitry of the video decoder is very similar to the circuitry in the feedback loop of the video encoder. Input pictures are first Huffman decoded and their type determined from the header information. For each macroblock, the

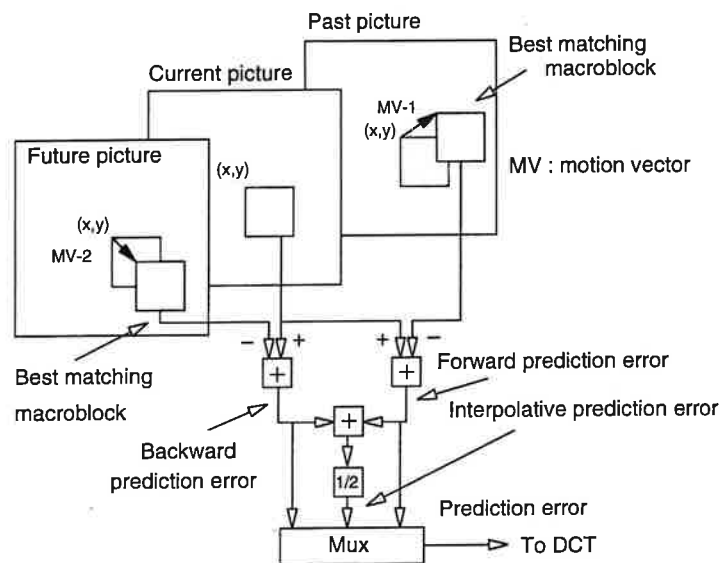


Figure 6.7 Bidirectional motion compensation.

coded coefficients are dequantized and translated into the spatial domain by an IDCT. Consider the encoded sequence $p_1, p_4, p_2, p_3, p_7, p_5, p_6$, and p_8 . For the purposes of this discussion, we will assume that all macroblocks in a P-picture are coded as P-macroblocks and all macroblocks in B-picture are coded as B-macroblocks. MPEG-1 offers more flexibility in macroblock coding and we will discuss this in a later section in this chapter. The decoding process is as follows:

1. Input p_1 (I-picture). No motion compensation is performed (the input to the multiplexor in Figure 6.8 is set to zero). Compute the IDCT and save the output into a display buffer and in the previous picture store.
2. Input p_4 (P-picture). For each macroblock, compute the IDCT and perform motion compensation; that is, add the area pointed to by the motion vector in the the previous picture store to the output of the IDCT. The reconstructed picture is saved in the future picture store.

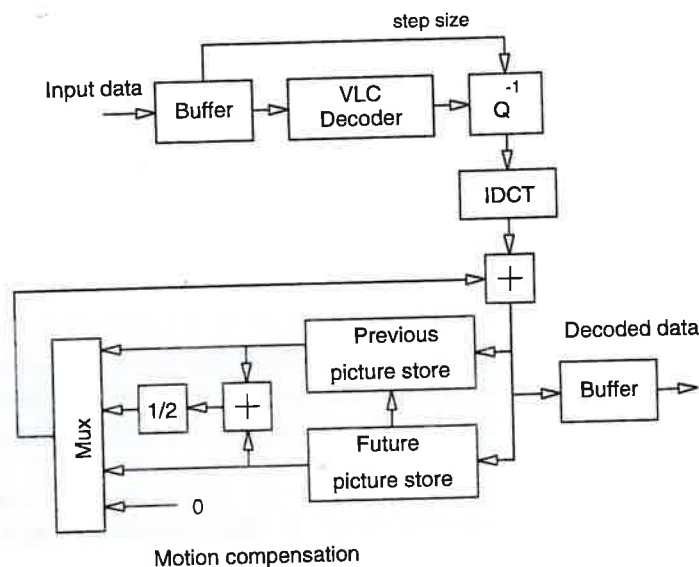


Figure 6.8 Block diagram of an MPEG decoder.

3. Input p_2 (B-picture). Compute the IDCT and perform bidirectional motion compensation; that is, using the two motion vectors, access the pixel values in the corresponding regions in the previous and future picture store areas, and compute a prediction for p_2 . Sum the IDCT output and the prediction. Since p_2 is the picture following p_1 in display order, p_2 can be displayed now. Note that, B-pictures need not be stored in the picture store area since they are not used in forming the predictions of subsequent pictures.
4. Input p_3 (B-picture). Repeat the steps performed for decoding p_2 . After displaying p_3 , p_4 (decoded in Step 2) is displayed next.
5. Input p_7 (P-picture). Repeat the steps performed for p_4 . However, the reconstructed picture needs to be saved in the previous picture store; this overwrites p_1 in the previous picture store.
6. Input p_5 and p_6 (B-pictures): Repeat the steps outlined for p_2 . Note that, predictions for p_5 and p_6 use p_4 and p_7 . After p_5 and p_6 are decoded, we first display p_4 and then the pictures p_5 , p_6 , and p_7 .

For low bit-rate applications, in addition to CIF, video coders may also use a quarter-CIF (QCIF) format, which has half the number of pixels and lines required for CIF. Support for CIF coding and decoding is optional; however, all coders must be able to operate using QCIF.

7.2.2 H.261 Encoding

Like MPEG, the H.261 encoding algorithm uses a combination of DCT coding and differential coding. Figure 7.2 shows a block diagram of an H.261 video encoder. The main elements are frame prediction, DCT transformation,

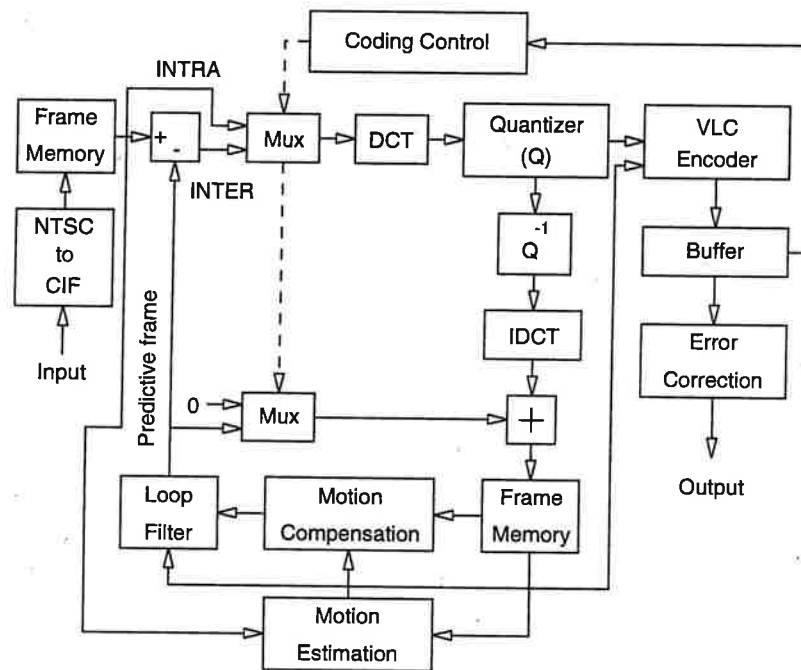


Figure 7.2 Block diagram of an H.261 video encoder.

quantization (Q), and variable length coding (VLC).

The DCT coding path is similar to the one used in JPEG and MPEG. A video frame is first translated into a CIF frame and then stored into frame memory. Noise filtering or other signal preprocessing can also be performed at this stage. Similarly to the operations in JPEG, the DCT operates on 8×8 picture blocks. Four luminance (Y) blocks and one Cb and one Cr color difference block are combined to form a macroblock.

Note that, frame prediction is done in a manner similar to that in MPEG-1, with the exception that only I-pictures and P-pictures are used. The macroblock organization and the classification of a macroblock as an intra-type or inter-type follows the approach in MPEG-1. Differential coding allows the DCT coder to operate on either input macroblocks (INTRA mode) or the differential macroblocks between the current frame and the prior frame (INTER mode). Not all macroblocks need to be coded and transmitted. For example, at low bit rates, macroblocks (and up to three full frames) can be skipped. The criteria for choosing either to transmit or skip a macroblock or the control mechanism for intra- or interframe coding are not part of the standard and may vary dynamically, depending on the complexity of the input signal and the output data rate constraints.

Following the DCT, the output DCT coefficients are quantized, coded using a variable length coder (a Huffman coder in this case), and stored into an output buffer. By monitoring the capacity of the buffer, the coder can perform data rate control dynamically. The rate control strategy is not specified in the H.261 standard; here, approaches similar to that adopted for MPEG-1 can be used. A proposal for rate-control is included in the informative part of the H.261 document.

Since H.261 is a predictive coder and since transmission errors may cause significant image quality problems, a BCH(511,493) coder is used to add parity bits for error detection and correction. However, the use of error detection and correction is optional for an H.261 decoder.

Because of the use of predictive coding, the encoder needs to monitor the quality of the transmitted image so that it does not diverge from the original signal. After an inverse quantization (Q^{-1}) and an inverse DCT (IDCT), the encoder is able to reconstruct a frame as it will be seen by the decoder. The reconstructed frame is stored into frame memory. Note that, since the VLC

operat
path t

As sh
for m
the lo
the pr
 $\times 8$ b
filter.
are $\frac{1}{4}$
0.124
motic
motic
apprc

The c
group

■
■
■
■

7.2.

Figur
error
leng
quar
or Π
data
con

operation is lossless, there is no need to include the VLC unit into the feedback path that generates the prediction frames.

As shown in Figure 7.2, the prediction path may also include optional circuitry for motion estimation and compensation, and a spatial loop filter. The role of the loop filter is to minimize the prediction error by smoothing the pixels in the previous frame. The loop filter is a separable 2-D filter that operates on 8×8 blocks. The corresponding 1-D filter is a three-tap finite impulse response filter. At block edges, the filter coefficients are 0, 1, and 0. Otherwise, they are $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{1}{4}$. Recent studies suggest that tap weights of 0.1248, 0.7495, and 0.1248 yield nearly a 1 dB improvement in signal quality when an accurate motion-estimation technique is used. On the other hand, for a less accurate motion-estimation method, the H.261 recommended tap weights may be more appropriate.

The compressed data stream is arranged hierarchically into four layers: picture, group of blocks (GOB), macroblock (MB), and block.

- A picture is the top layer.
- Each picture is divided into groups of blocks (GOBs). A GOB is either one-twelfth of a CIF picture or one-third of a QCIF picture.
- Each GOB is divided into 33 macroblocks.
- Each macroblock consists of six 8×8 blocks, that is, four blocks of luminance (Y), one block of Cb, and one block of Cr.

7.2.3 The H.261 Video Decoder

Figure 7.3 shows a block diagram of the H.261 video decoder. After optional error correction, the compressed input is buffered and processed by the variable length decoder. The decoded data are parsed and then processed by an inverse quantizer and an inverse DCT. Depending on the transmission mode (INTRA or INTER), macroblocks from a prior frame may also be added to the current data to form the reconstructed data. The decoded CIF data may then be converted to the appropriate display format.

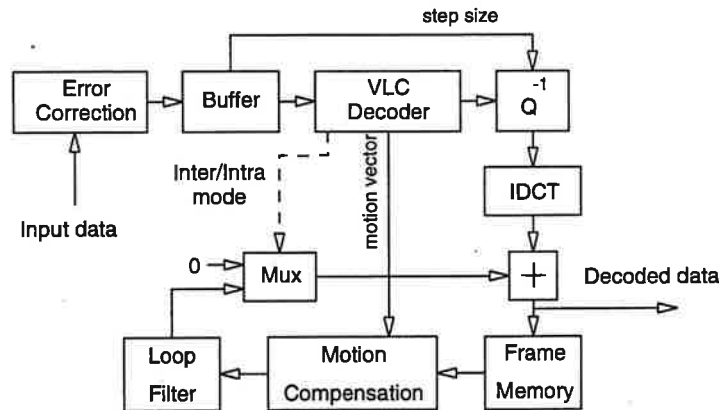


Figure 7.3 Block diagram of an H.261 video decoder.

7.3 IMPLEMENTATION ISSUES

Not all implementation details are specified by H.261. This gives manufacturers the opportunity to tailor their implementations to suit their target applications and to provide for product differentiation. Among the details that are open to variations are rate-control mechanisms, motion-estimation algorithms, and preprocessing and postprocessing algorithms.

In early field trials of H.261 codecs, it was noted that small variations in the implementation of the inverse DCT between two coders may yield noticeable picture artifacts. To resolve this problem, the H.261 Recommendation provides guidelines for measuring the accuracy of a particular implementation of the inverse DCT. The procedure to test the accuracy of an IDCT is outlined in Annex A of the H.261 standard and consists of the following steps: (1) generate random data in a specified range, and form 8×8 data blocks, (2) perform a DCT on each block, (3) perform an IDCT, and (4) measure peak, mean, and mean square error between the original data and the output of the IDCT. To be H.261 compliant, the following conditions must be met:

- For any pixel, the peak error should be less than 1.0, the average error (over 10,000 blocks) should be less than 0.015, and the mean square error should be less than 0.06.