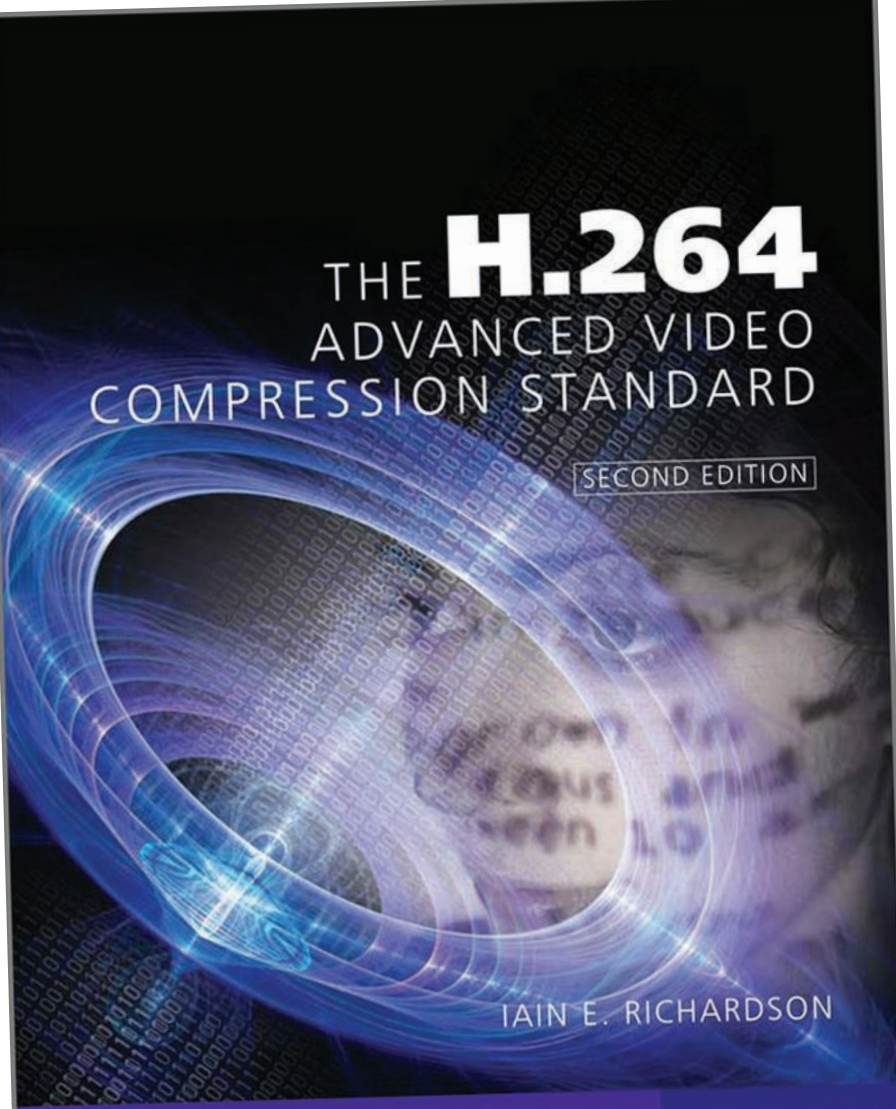
THE H.264 ADVANCED VIDEO COMPRESSION STANDARD

SECOND EDITION



IAIN E. RICHARDSON

WILEY

Nokia Exhibit 2011, p. 1
Amazon.com v. Nokia
IPR2024-00604

THE H.264 ADVANCED VIDEO COMPRESSION STANDARD

Second Edition

Iain E. Richardson
Vcodex Limited, UK

WILEY
A John Wiley and Sons, Ltd., Publication

Nokia Exhibit 2011, p. 2
Amazon.com v. Nokia
IPR2024-00604

This edition first published 2010
© 2010, John Wiley & Sons, Ltd

First Edition published in 2003

Registered office
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom

For details of our global editorial offices, for customer services and for information about how to apply for permission to reuse the copyright material in this book please see our website at www.wiley.com.

The right of the author to be identified as the author of this work has been asserted in accordance with the Copyright, Designs and Patents Act 1988.

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, electronic, mechanical, photocopying, recording or otherwise, except as permitted by the UK Copyright, Designs and Patents Act 1988, without the prior permission of the publisher.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Designations used by companies to distinguish their products are often claimed as trademarks. All brand names and product names used in this book are trade names, service marks, trademarks or registered trademarks of their respective owners. The publisher is not associated with any product or vendor mentioned in this book. This publication is designed to provide accurate and authoritative information in regard to the subject matter covered. It is sold on the understanding that the publisher is not engaged in rendering professional services. If professional advice or other expert assistance is required, the services of a competent professional should be sought.

Library of Congress Cataloguing-in-Publication Data

Richardson, Iain E. G.
The H.264 advanced video compression standard / Iain E. Richardson. – 2nd ed.
p. cm.
Rev. ed. of: H.264 and MPEG-4 video compression. c2003.
Includes bibliographical references and index.
ISBN 978-0-470-51692-8 (pbk.)
1. Digital video–Standards. 2. Video compression–Standards. 3. MPEG (Video coding standard) 4. Multimedia systems. I. Richardson, Iain E. G. H.264 and MPEG-4 video compression. II. Title.
TK6680.5.R52 2010
006.6′96–dc22

2009054387

A catalogue record for this book is available from the British Library.

ISBN: 978-0-470-51692-8

Typeset in 10/12pt Times by Aptara Inc., New Delhi, India
Printed and bound in Great Britain by CPI Autony Rowe, Chippenham, Wiltshire

Nokia Exhibit 2011, p. 3
Amazon.com v. Nokia
IPR2024-00604

parameters, together with previously decoded image pixels, to create a prediction of the current frame and the frame itself is reconstructed by adding the residual frame to this prediction.

3.3 Prediction model

The data to be processed are a set of image samples in the current frame or field. The goal of the prediction model is to reduce redundancy by forming a prediction of the data and subtracting this prediction from the current data. The prediction may be formed from previously coded frames (a **temporal prediction**) or from previously coded image samples in the same frame (a **spatial prediction**). The output of this process is a set of residual or difference samples and the more accurate the prediction process, the less energy is contained in the residual. The residual is encoded and sent to the decoder which re-creates the same prediction so that it can add the decoded residual and reconstruct the current frame. In order that the decoder can create an identical prediction, it is essential that the encoder forms the prediction using only data available to the decoder, i.e. data that has already been coded and transmitted.

3.3.1 Temporal prediction

The predicted frame is created from one or more past or future frames known as **reference frames**. The accuracy of the prediction can usually be improved by compensating for motion between the reference frame(s) and the current frame.

3.3.1.1 Prediction from the previous video frame

The simplest method of temporal prediction is to use the previous frame as the predictor for the current frame. Two successive frames from a video sequence are shown in Figure 3.4 and Figure 3.5. Frame 1 is used as a predictor for frame 2 and the residual formed by subtracting the predictor (frame 1) from the current frame (frame 2) is shown in Figure 3.6. In this image, mid-grey represents a difference of zero and light or dark greys correspond to positive and negative differences respectively. The obvious problem with this simple prediction is that a lot of energy remains in the residual frame (indicated by the light and dark areas) and this means that there is still a significant amount of information to compress after temporal prediction. Much of the residual energy is due to object movements between the two frames and a better prediction may be formed by **compensating** for motion between the two frames.

3.3.1.2 Changes due to motion

Causes of changes between video frames include motion, uncovered regions and lighting changes. Types of motion include rigid object motion, for example a moving car, deformable object motion, for example a person speaking, and camera motion such as panning, tilt, zoom and rotation. An uncovered region may be a portion of the scene background uncovered by a moving object. With the exception of uncovered regions and lighting changes, these differences correspond to pixel movements between frames. It is possible to estimate the trajectory of each pixel between successive video frames, producing a field of pixel trajectories known as the

Nokia Exhibit 2011, p. 4
Amazon.com v. Nokia
IPR2024-00604



Figure 3.4 Frame 1



Figure 3.5 Frame 2

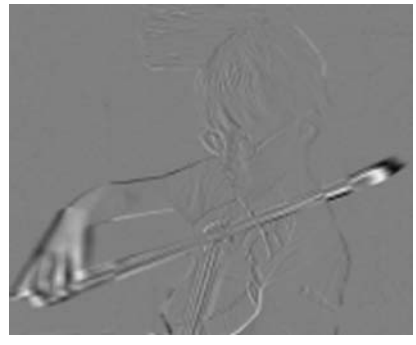


Figure 3.6 Difference

optical flow or optic flow [ii]. Figure 3.7 shows the optical flow field for the frames of Figure 3.4 and Figure 3.5. The complete field contains a flow vector for every pixel position but for clarity, the field is sub-sampled so that only the vector for every 2nd pixel is shown.

If the optical flow field is accurately known, it should be possible to form an accurate prediction of most of the pixels of the current frame by moving each pixel from the reference frame along its optical flow vector. However, this is not a practical method of motion compensation for several reasons. An accurate calculation of optical flow is very computationally intensive,

Nokia Exhibit 2011, p. 5
Amazon.com v. Nokia
IPR2024-00604



Figure 3.7 Optical flow

since the more accurate methods use an iterative procedure for every pixel, and for the decoder to re-create the prediction frame it would be necessary to send the optical flow vector for every pixel to the decoder resulting in a large amount of transmitted data and negating the advantage of a small residual.

3.3.1.3 Block-based motion estimation and compensation

A practical and widely used method of motion compensation is to compensate for movement of rectangular sections or 'blocks' of the current frame. The following procedure is carried out for each block of MxN samples in the current frame:

- Search an area in the reference frame, a past or future frame, to find a similar MxN-sample region. This search may be carried out by comparing the MxN block in the current frame with some or all of the possible MxN regions in a search area, e.g. a region centred on the current block position, and finding the region that gives the 'best' match. A popular matching criterion is the energy in the residual formed by subtracting the candidate region from the current MxN block, so that the candidate region that minimises the residual energy is chosen as the best match. This process of finding the best match is known as **motion estimation**.
- The chosen candidate region becomes the predictor for the current MxN block (a motion compensated prediction) and is subtracted from the current block to form a residual MxN block (**motion compensation**).
- The residual block is encoded and transmitted and the offset between the current block and the position of the candidate region (**motion vector**) is also transmitted.

The decoder uses the received motion vector to re-create the predictor region. It decodes the residual block, adds it to the predictor and reconstructs a version of the original block.

Block-based motion compensation is popular for a number of reasons. It is relatively straightforward and computationally tractable, it fits well with rectangular video frames and with block-based image transforms such as the Discrete Cosine Transform and it provides a

Nokia Exhibit 2011, p. 6
Amazon.com v. Nokia
IPR2024-00604

reasonably effective temporal model for many video sequences. There are however a number of disadvantages. For example, 'real' objects rarely have neat edges that match rectangular boundaries, objects often move by a fractional number of pixel positions between frames and many types of object motion are hard to compensate for using block-based methods, e.g. deformable objects, rotation, warping and complex motion such as a cloud of smoke. Despite these disadvantages, block-based motion compensation is the basis of the temporal prediction model used by all current video coding standards.

3.3.1.4 Motion compensated prediction of a macroblock

The **macroblock**, corresponding to a 16 × 16-pixel region of a frame, is the basic unit for motion compensated prediction in a number of important visual coding standards including MPEG-1, MPEG-2, MPEG-4 Visual, H.261, H.263 and H.264. For source video material in the popular 4:2:0 format (Chapter 2), a macroblock is organised as shown in Figure 3.8. A 16 × 16-pixel region of the source frame is represented by 256 luminance samples arranged in four 8 × 8-sample blocks, 64 red chrominance samples in one 8 × 8 block and 64 blue chrominance samples in one 8 × 8 block, giving a total of six 8 × 8 blocks. An H.264/AVC codec processes each video frame in units of a macroblock.

Motion estimation:
Motion estimation of a macroblock involves finding a 16 × 16-sample region in a reference frame that closely matches the current macroblock. The reference frame is a previously encoded frame from the sequence and may be before or after the current frame in display order. A search area in the reference frame centred on the current macroblock position is searched and the 16 × 16 region within the search area that minimizes a matching criterion is chosen as the 'best match' (Figure 3.9).

Motion compensation:
The luma and chroma samples of the selected 'best' matching region in the reference frame is subtracted from the current macroblock to produce a residual macroblock that is encoded and transmitted together with a motion vector describing the position of the best matching region relative to the current macroblock position.

There are many variations on the basic motion estimation and compensation process. The reference frame may be a previous frame in temporal order, a future frame or a combination

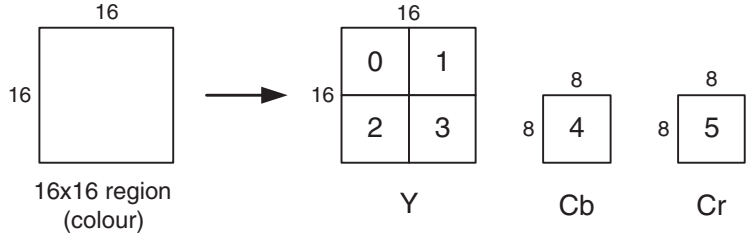


Figure 3.8 Macroblock (4:2:0)

Nokia Exhibit 2011, p. 7
Amazon.com v. Nokia
IPR2024-00604

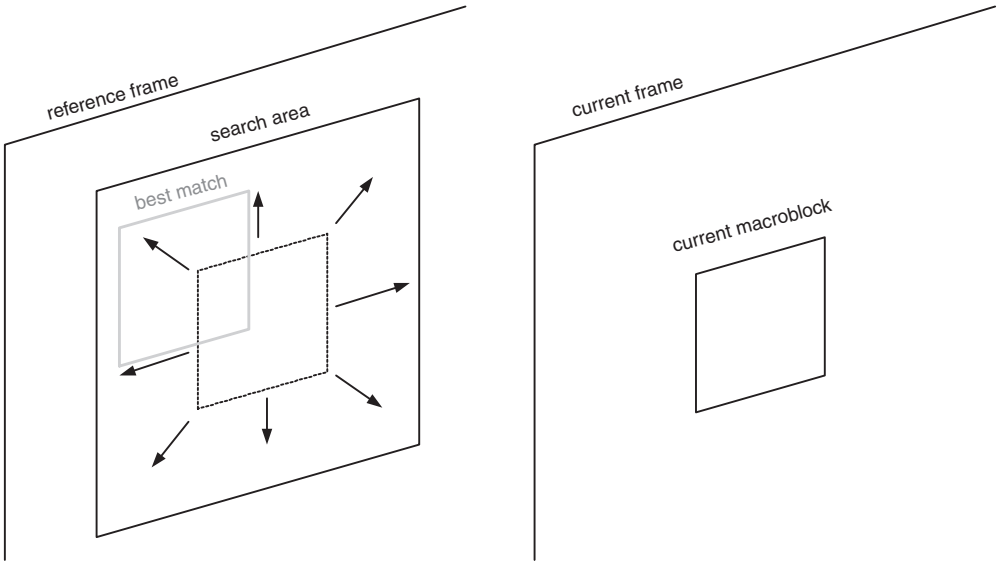


Figure 3.9 Motion estimation

of predictions from two or more previously encoded frames. If a future frame is chosen as the reference, it is necessary to encode this frame before the current frame, i.e. frames must be encoded out of order. Where there is a significant change between the reference and current frames, for example a scene change or an uncovered area, it may be more efficient to encode the macroblock without motion compensation and so an encoder may choose **intra** mode encoding using intra prediction or **inter** mode encoding with motion compensated prediction for each macroblock. Moving objects in a video scene rarely follow 'neat' 16 × 16-pixel boundaries and so it may be more efficient to use a variable block size for motion estimation and compensation. Objects may move by a fractional number of pixels between frames, e.g. 2.78 pixels rather than 2.0 pixels in the horizontal direction, and a better prediction may be formed by interpolating the reference frame to sub-pixel positions before searching these positions for the best match.

3.3.1.5 Motion compensation block size

Two successive frames of a video sequence are shown in Figure 3.10 and Figure 3.11. Frame 1 is subtracted from frame 2 without motion compensation to produce a residual frame (Figure 3.12). The energy in the residual is reduced by motion compensating each 16 × 16 macroblock (Figure 3.13). Motion compensating each 8 × 8 block instead of each 16 × 16 macroblock reduces the residual energy further (Figure 3.14) and motion compensating each 4 × 4 block gives the smallest residual energy of all (Figure 3.15). These examples show that smaller motion compensation block sizes can produce better motion compensation results. However, a smaller block size leads to increased complexity, with more search operations to be carried out, and an increase in the number of motion vectors that need to be transmitted. Sending each

Nokia Exhibit 2011, p. 8
Amazon.com v. Nokia
IPR2024-00604



Figure 3.10 Frame 1



Figure 3.11 Frame 2



Figure 3.12 Residual : no motion compensation

Nokia Exhibit 2011, p. 9
Amazon.com v. Nokia
IPR2024-00604



Figure 3.13 Residual : 16 × 16 block size



Figure 3.14 Residual : 8 × 8 block size

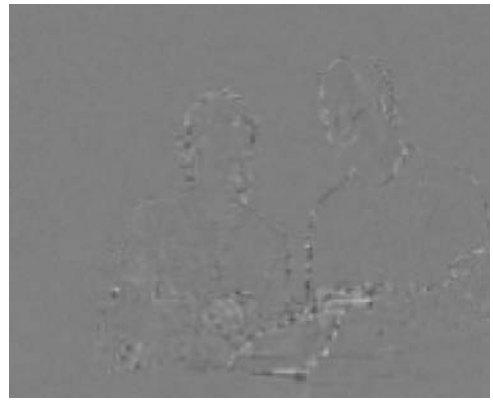


Figure 3.15 Residual : 4 × 4 block size

Nokia Exhibit 2011, p. 10
Amazon.com v. Nokia
IPR2024-00604

motion vector requires bits to be transmitted and the extra overhead for vectors may outweigh the benefit of reduced residual energy. An effective compromise is to adapt the block size to the picture characteristics, for example choosing a large block size in flat, homogeneous regions of a frame and choosing a small block size around areas of high detail and complex motion (Chapter 6).

3.3.1.6 Sub-pixel motion compensation

Figure 3.16 shows a close-up view of part of a reference frame. In some cases, predicting from interpolated sample positions in the reference frame may form a better motion compensated prediction. In Figure 3.17, the reference region pixels are interpolated to half-pixel positions and it may be possible to find a better match for the current macroblock by searching the interpolated samples. Sub-pixel motion estimation and compensation involves searching sub-pixel interpolated positions as well as integer-pixel positions and choosing the position that gives the best match and minimizes the residual energy. Figure 3.18 shows the concept of quarter-pixel motion estimation. In the first stage, motion estimation finds the best match on the integer pixel grid (circles). The encoder searches the half-pixel positions immediately next to this best match (squares) to see whether the match can be improved and if required, the quarter-pixel positions next to the best half-pixel position (triangles) are then searched. The final match, at an integer, half-pixel or quarter-pixel position, is subtracted from the current block or macroblock.

The residual in Figure 3.19 is produced using a block size of 4 × 4 pixels using half-pixel interpolation and has less residual energy than Figure 3.15. This approach may be extended further by interpolation onto a $^1/_4$-pixel grid to give a still smaller residual (Figure 3.20). In

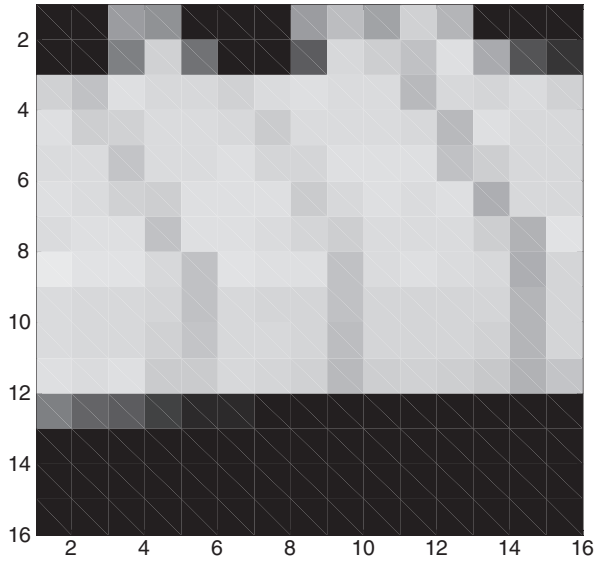


Figure 3.16 Close-up of reference region

Nokia Exhibit 2011, p. 11
Amazon.com v. Nokia
IPR2024-00604

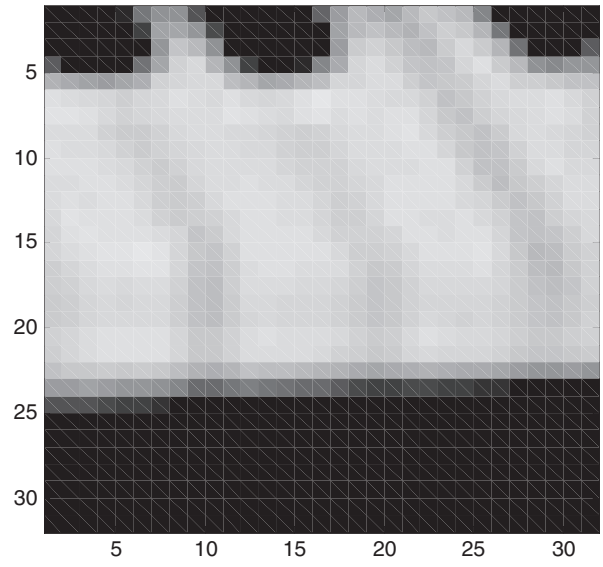


Figure 3.17 Reference region interpolated to half-pixel positions

general, 'finer' interpolation provides better motion compensation performance, producing a smaller residual at the expense of increased complexity. The performance gain tends to diminish as the interpolation steps increase. Half-pixel interpolation gives a significant gain over integer-pixel motion compensation, quarter-pixel interpolation gives a moderate further improvement, eighth-pixel interpolation gives a small further improvement again and so on.

Some examples of the performance achieved by sub-pixel interpolation are given in Table 3.1. A motion-compensated reference frame, the previous frame in the sequence, is subtracted from the current frame and the energy of the residual, approximated by the Sum of Absolute Errors (SAE), is listed in the table. A lower SAE indicates better motion compensation performance. In each case, sub-pixel motion compensation gives improved performance compared with integer-pixel compensation. The improvement from integer to half-pixel is

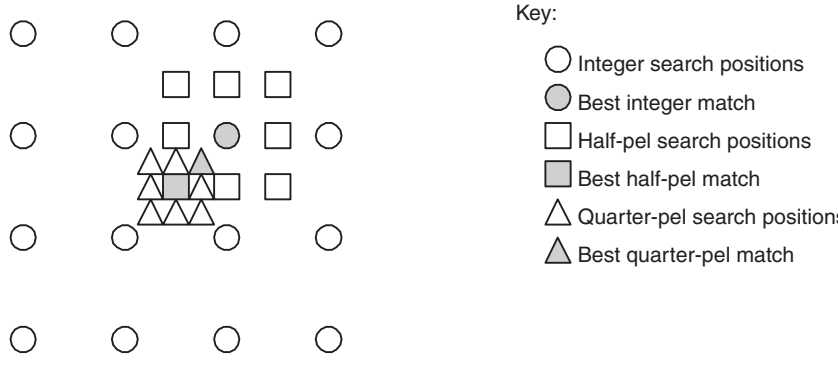


Figure 3.18 Integer, half-pixel and quarter-pixel motion estimation

Nokia Exhibit 2011, p. 12
Amazon.com v. Nokia
IPR2024-00604

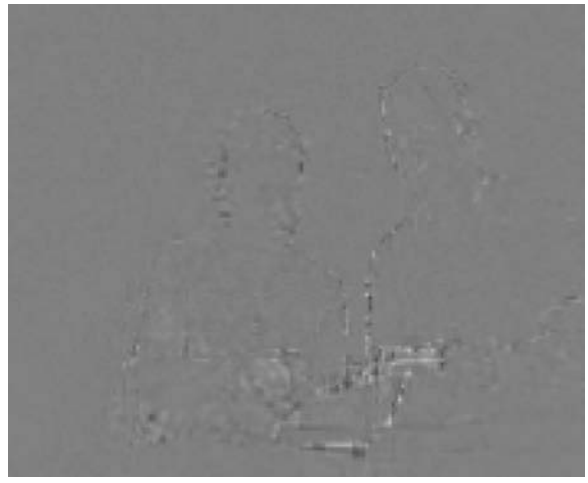


Figure 3.19 Residual : 4 × 4 blocks, 1/2-pixel compensation

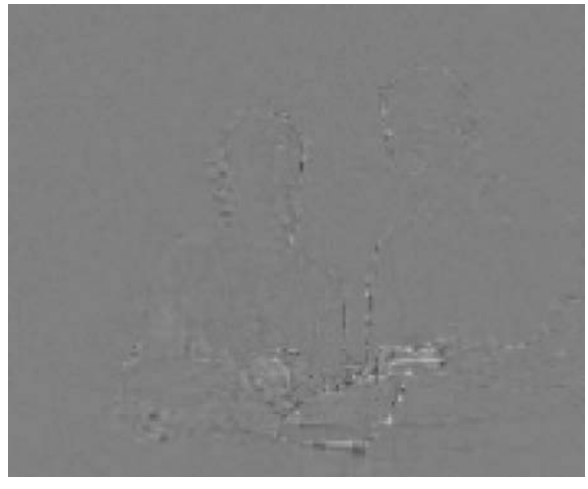


Figure 3.20 Residual : 4 × 4 blocks, 1/4-pixel compensation

Table 3.1 SAE of residual frame after motion compensation, 16 × 16 block size

Sequence	No motion compensation	Integer-pel	Half-pel	Quarter-pel
'Violin', QCIF	171945	153475	128320	113744
'Grasses', QCIF	248316	245784	228952	215585
'Carphone', QCIF	102418	73952	56492	47780

Nokia Exhibit 2011, p. 13
Amazon.com v. Nokia
IPR2024-00604

more significant than the further improvement from half- to quarter-pixel. The sequence 'Grasses' has highly complex motion and is particularly difficult to motion-compensate, hence the large SAE; 'Violin' and 'Carphone' are less complex and motion compensation produces smaller SAE values.

Searching for matching 4 × 4 blocks with ¼-pixel interpolation is considerably more complex than searching for 16 × 16 blocks with no interpolation. In addition to the extra complexity, there is a coding penalty since the vector for every block must be encoded and transmitted to the receiver in order to correctly reconstruct the image. As the block size is reduced, the number of vectors that have to be transmitted increases. More bits are required to represent ½ or ¼-pixel vectors because the fractional part of the vector, e.g. 0.25 or 0.5, must be encoded as well as the integer part. Figure 3.21 plots the integer motion vectors that are transmitted along with the residual of Figure 3.13. The motion vectors required for the residual of Figure 3.20 are plotted in Figure 3.22, in which there are 16 times as many vectors, each represented by two fractional numbers DX and DY with ¼-pixel accuracy. There is therefore a trade-off in compression efficiency associated with more complex motion compensation schemes, since more accurate motion compensation requires more bits to encode the vector field, but fewer bits to encode the residual whereas less accurate motion compensation requires fewer bits for the vector field but more bits for the residual. The efficiency of sub-pixel interpolation schemes can be improved by using sophisticated interpolation filters [iii].

3.3.2 Spatial model: intra prediction

The prediction for the current block of image samples is created from previously-coded samples in the same frame. Figure 3.23 shows a block that is to be predicted, centre, in the current frame. Assuming that the blocks of image samples are coded in raster-scan order, which is not always the case, the upper/left shaded blocks are available for intra prediction. These blocks have already been coded and placed in the output bitstream. When the decoder processes the current block, the shaded upper/left blocks are already decoded and can be used to re-create the prediction.

Many different approaches to intra prediction have been proposed. H.264/AVC uses spatial extrapolation to create an intra prediction for a block or macroblock. Figure 3.24 shows the

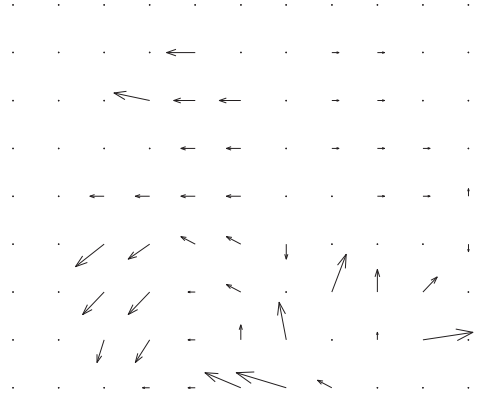


Figure 3.21 Motion vector map : 16 × 16 blocks, integer vectors

Nokia Exhibit 2011, p. 14
Amazon.com v. Nokia
IPR2024-00604

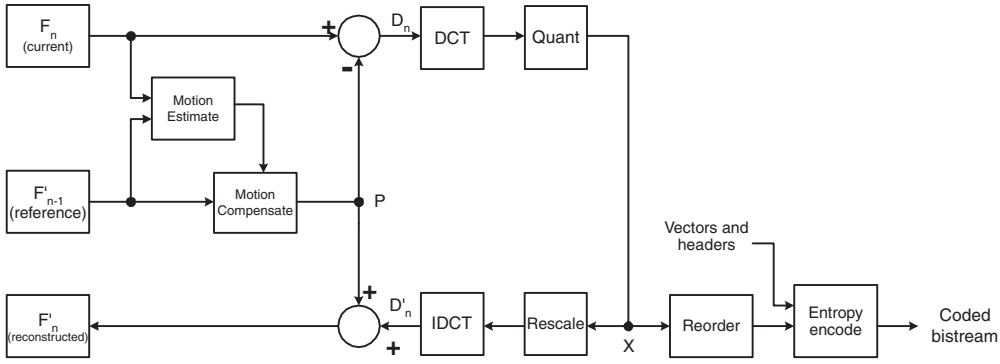


Figure 3.51 DPCM/DCT video encoder

Figure 3.51 and Figure 3.52 show the basic DPCM/DCT hybrid encoder and decoder. In the encoder, video frame n (F_n) is processed to produce a coded or compressed bitstream. In the decoder, the compressed bitstream, shown at the right of the figure, is decoded to produce a reconstructed video frame F'_n. The reconstructed output frame is not usually identical to the source frame. The figures have been deliberately drawn to highlight the common elements within encoder and decoder. Most of the functions of the decoder are actually contained within the encoder, the reason for which will be explained later.

Encoder data flow

There are two main data flow paths in the encoder, left to right (encoding) and right to left (reconstruction). The encoding flow is as follows:

1. An input video frame F_n is presented for encoding and is processed in units of a macroblock, corresponding to a 16 × 16 pixel region of the video image.
2. F_n is compared with a **reference** frame, for example the previous encoded frame (F'_{n-1}). A motion estimation function finds a 16 × 16 region in F'_{n-1} that 'matches' the current macroblock in F_n. The offset between the current macroblock position and the chosen reference region is a motion vector MV.
3. Based on the chosen motion vector MV, a motion compensated prediction P is generated, the 16 × 16 region selected by the motion estimator. P may consist of interpolated sub-pixel data.
4. P is subtracted from the current macroblock to produce a residual or difference macroblock D.

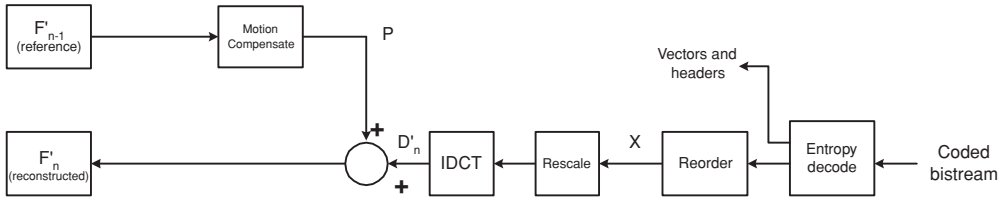


Figure 3.52 DPCM/DCT video decoder

Nokia Exhibit 2011, p. 15
Amazon.com v. Nokia
IPR2024-00604

In this example, the B slices are not used for reference. The POC cycle consists of one reference frame, with an offset from the previous reference frame to the next reference frame of +6 and an offset from the previous reference frame to the next non-reference frame of −4.

Slice	Type	Used for reference	*frame_num*	Delta POC	Offset	POC	Display order
1st	I	Yes	**0**	0	0	0	0
2nd	P	Yes	**1**	0	+6	6	3
3rd	B	No	**2**	0	−4	2	1
4th	B	No	2	2	−4+2	4	2
5th	P	Yes	2	0	+6	12	6
6th	B	No	**3**	0	−4	8	4
7th	B	No	3	2	−4+2	10	5
8th	P	Yes	3	0	+6	18	9
...	...						

The 4th and 7th slices have a delta POC, signalled in the slice header, of +2. The POC for these slices is calculated as: POC = expected POC + 2.

5.3.3 Reference picture lists

A picture that is coded and available for reference is stored in the Decoded Picture Buffer (DPB) and marked as one of the following:

(a) a short term reference picture, indexed according to *frame_num* or POC or,
(b) a long term reference picture, indexed according to *LongTermPicNum*, a reference number derived from the parameter *LongTermFrameIdx* which is assigned when a picture is marked as a long term reference picture.

Short term reference pictures may be assigned a *LongTermFrameIdx*, i.e. changed to a long term reference picture, at a later time.

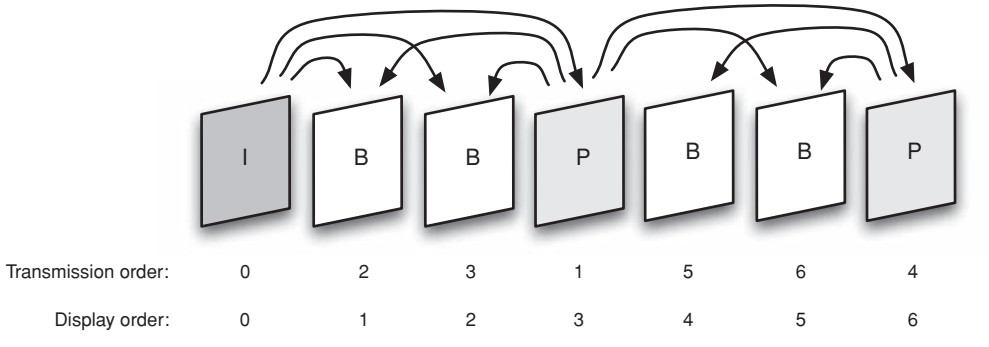


Figure 5.5 Display order: Type 1 example

Nokia Exhibit 2011, p. 16
Amazon.com v. Nokia
IPR2024-00604

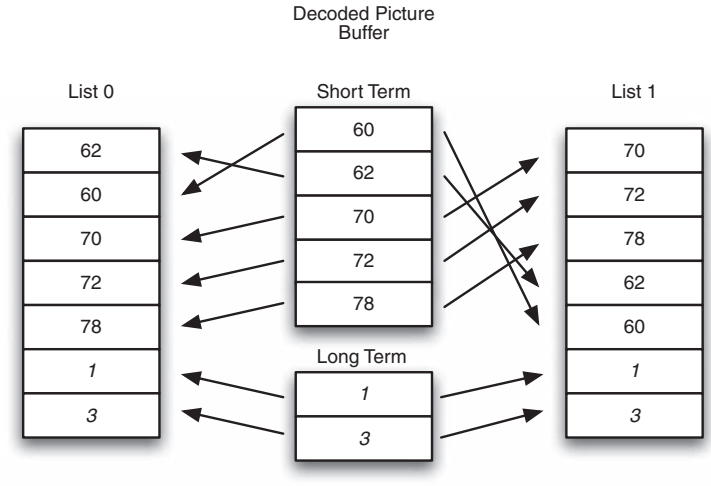


Figure 5.6 List 0 and List 1 ordering: example

Reference Picture Order Example: B slice (list0 and list1)

The DPB contains short term reference pictures with POC 60, 62, 70, 72 and 78 and long term reference pictures with LongTermFrameIdx *1* and *3*. The current POC is 68. The default order of list0 is:

62, 60, 70, 72, 78, *1, 3*

The default order of list1 is:

70, 72, 78, 62, 60, *1, 3*

Figure 5.6 shows how the two lists are created from the sequence of pictures in the DPB. List0 defaults to prediction from 'past' pictures in display order whilst List1 defaults to 'future' pictures in display order. This is shown graphically in Figure 5.7. Note that the first entry in

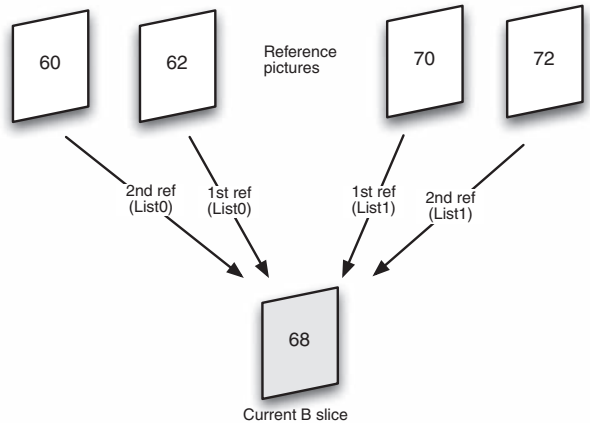


Figure 5.7 Default reference pictures: example

Nokia Exhibit 2011, p. 17
Amazon.com v. Nokia
IPR2024-00604

8

H.264 conformance, transport and licensing

8.1 Introduction

Chapters 4,5,6 and 7 covered the basic concepts of H.264 Advanced Video Compression and the key algorithms that enable an H.264 codec to efficiently code and decode video. The purpose of an industry standard is to enable interoperability between encoders, bitstreams and decoders; i.e. the standard makes it possible for a bitstream encoded by one manufacturer's encoder to be decoded by a different manufacturer's decoder. H.264/AVC defines Profiles and Levels to place operational limits on (a) the particular coding tools and (b) the computational capacity and storage required to decode a sequence. Conformance is verified using a theoretical 'model', a Hypothetical Reference Decoder.

Practical applications of H.264/AVC involve transmitting and/or storing coded video information. The standard includes a number of features designed to support efficient, robust transport of the coded bitstream, including Parameter Sets and NAL Units, described in Chapter 5, and specific transport tools, described in this chapter. To help support an increasingly diverse range of video content and display types, 'side' information including Supplemental Enhancement Information and Video Usability Information may be transmitted along with the coded video data.

Video coding is big business, with many worldwide industries relying on video compression to enable digital media products and services. With many thousands of published patents in the field of video coding, licensing of H.264/AVC implementations is an important issue for commercial digital video applications.

8.2 Conforming to the Standard

H.264/AVC specifies many syntax options and decoding algorithms [i] which cover a wide range of potential video coding scenarios. The standard is designed to support video coding for applications such as small hand-held devices with limited display resolution and minimal computational capacity, through to high-definition decoders with large amounts of memory

The H.264 Advanced Video Compression Standard, Second Edition Iain E. Richardson
© 2010 John Wiley & Sons, Ltd

Nokia Exhibit 2011, p. 18
Amazon.com v. Nokia
IPR2024-00604

and computing resources. The standard describes the various syntax elements that may occur in a bitstream and specifies exactly how each syntax element should be processed and decoded in order to produce an output video sequence.

It is important to know whether a particular decoder can handle a particular coded sequence, i.e. whether the decoding and display operations are within the decoder's capabilities. This is achieved by specifying a **profile** and **level** for every coded sequence. The profile places algorithmic constraints on the decoder, determining which decoding tools the decoder should be capable of handling, whilst the level places data processing and storage constraints on the decoder, determining how much data the decoder should be capable of storing, processing and outputting to a display. An H.264 decoder can immediately determine whether it is capable of decoding a particular bitstream by extracting the Profile and Level parameters and determining whether these are supported by the decoder's capabilities.

8.2.1 Profiles

The H.264/AVC standard specifies a number of **Profiles**, each specifying a subset of the coding tools available in the H.264 standard. A Profile places limits on the algorithmic capabilities required of an H.264 decoder. Hence a decoder conforming to the Main Profile of H.264 only needs to support the tools contained within the Main Profile; a High Profile decoder needs to support further coding tools; and so on. Each Profile is intended to be useful to a class of applications. For example, the Baseline Profile may be useful for low-delay, 'conversational' applications such as video conferencing, with relatively low computational requirements. The Main Profile may be suitable for basic television/entertainment applications such as Standard Definition TV services. The High Profiles add tools to the Main Profile which can improve compression efficiency especially for higher spatial resolution services, e.g. High Definition TV.

8.2.1.1 Baseline, Constrained Baseline, Extended and Main Profiles

Figure 8.1 shows the tools supported by the Baseline, Constrained Baseline, Extended and Main Profiles. The **Baseline Profile** was originally intended to be suitable for low complexity, low delay applications such as conversational or mobile video transmission. It includes I and P slice types, allowing intra prediction and motion compensated prediction from a single reference, the basic 4×4 integer transform and CAVLC entropy coding. It also supports three tools for improved transport efficiency, FMO, ASO and Redundant Slices (section 8.3). However, these last three tools have not tended to be popular with codec manufacturers and most implementations of H.264/AVC do not support FMO, ASO or Redundant Slices. In recognition of this, a recent amendment to the standard includes the **Constrained Baseline Profile** which excludes these tools [ii].

The **Extended Profile** is a superset of the Baseline Profile, adding further tools that may be useful for efficient network streaming of H.264 data (section 8.3). The **Main Profile** is a superset of the Constrained Baseline Profile and adds coding tools that may be suitable for broadcast and entertainment applications such as digital TV and DVD playback, namely CABAC entropy coding and bipredicted B slices with prediction modes such as Weighted Prediction for better coding efficiency and frame/field coding support for interlaced video content.

Nokia Exhibit 2011, p. 19
Amazon.com v. Nokia
IPR2024-00604

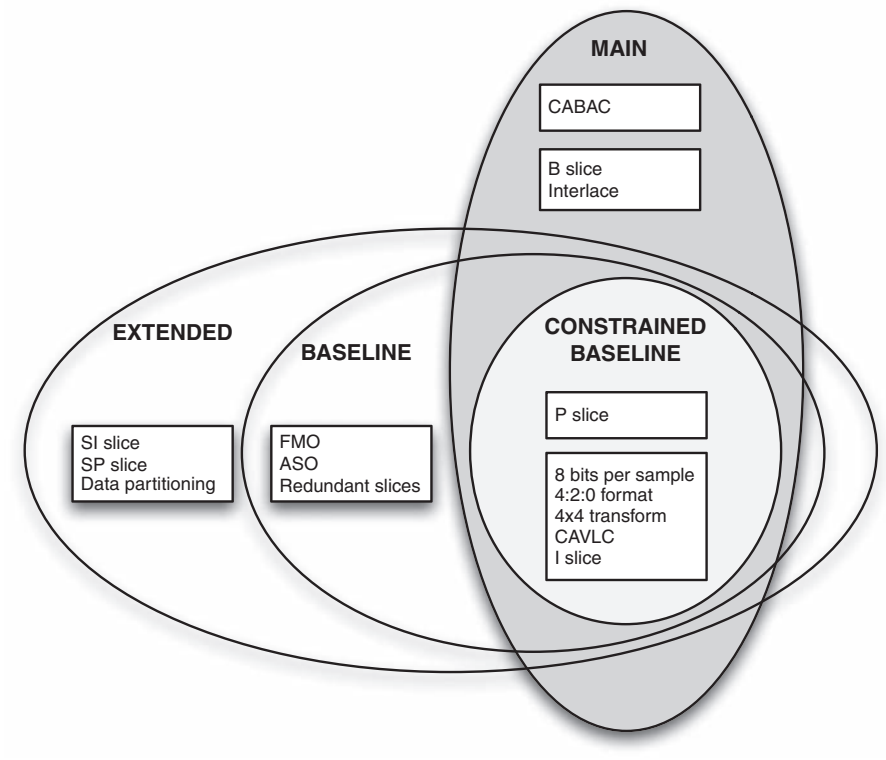


Figure 8.1 Baseline, Constrained Baseline, Extended and Main Profiles

8.2.1.2 High Profiles

Four High Profiles are shown in Figure 8.2, together with the Main Profile for comparison. Each of these Profiles adds coding tools that support higher-quality applications – High Definition, extended bit depths, higher colour depths – at the expense of greater decoding complexity. The High Profile is a superset of the Main Profile and adds the following tools: 8×8 transform and 8×8 inter prediction for better coding performance, especially at higher spatial resolutions, quantizer scale matrices which support frequency-dependent quantizer weightings, separate quantizer parameters for Cr and Cb and support for monochrome video (4:0:0 format). The High Profile makes it possible to use a higher coded data rate for the same Level (see section 8.2.2). The High Profile may be particularly useful for High Definition applications.

Further profiles add more sophisticated tools that may be necessary or useful for 'professional' applications such as content distribution, archiving, etc. The maximum number of bits per sample is extended to 10 bits in the High10 profile and to 14 bits in the High444Pred profile. High422 Profile adds support for 4:2:2 video, i.e. higher Chroma resolution, and High444 Profile extends this to 4:4:4 video giving equal resolution in Luma and Chroma components and adds separate coding for each colour component and a further lossless coding mode that uses predictive coding (Chapter 7).

Nokia Exhibit 2011, p. 20
Amazon.com v. Nokia
IPR2024-00604

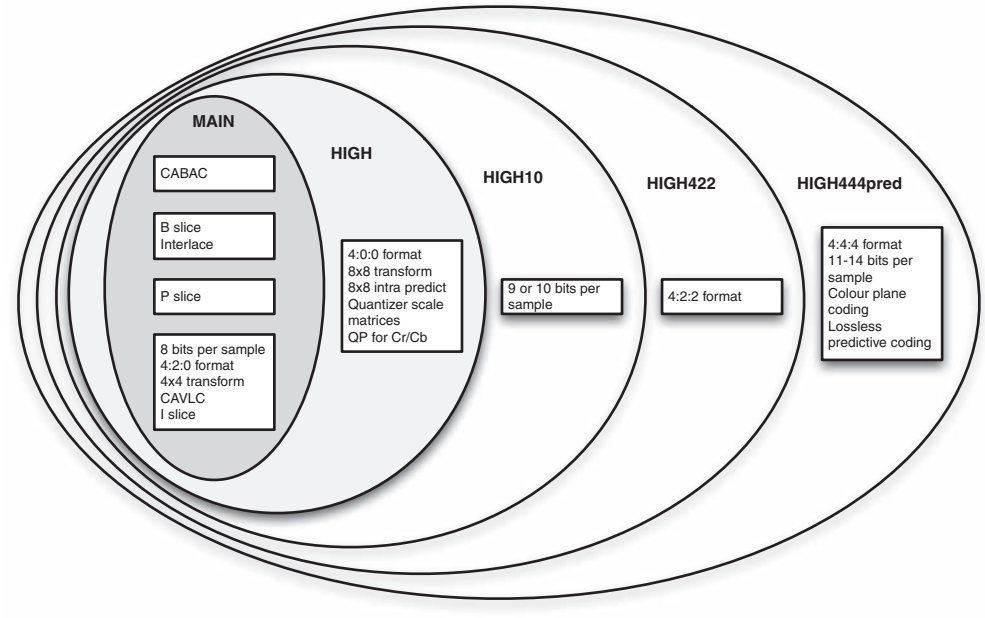


Figure 8.2 Main and High Profiles

8.2.1.3 Intra Profiles

Figure 8.3 shows the Main Profile together with four Intra Profiles. Each of these includes selected tools contained in the High Profiles of Figure 8.2, but without Inter coding support, i.e. no P or B slices, or Interlace support. These Intra Profiles may be useful for applications such as video editing which require efficient coding of individual frames but also require complete random access to coded frames and hence do not require inter coding.

8.2.2 Levels

The Sequence Parameter Set defines a Level for the coded bitstream, a set of constraints imposed on values of the syntax elements in the H.264/AVC bitstream. The combination of Profile and Level constrains the maximum computational and memory requirements that will be placed on the decoder. The main Level constraints are as follows:

- Maximum macroblock processing rate (MaxMBPS): the maximum number of macroblocks, 16×16 luma and associated chroma, that a decoder must handle per second.
- Maximum frame size (MaxFS): the maximum number of macroblocks in a decoded frame.
- Maximum Decoded Picture Buffer size (MaxDPB): the maximum memory space required to store decoded pictures at the decoder.
- Maximum video bit rate (MaxBR): the maximum coded video bitrate.
- Maximum Coded Picture Buffer size (MaxCBP): the maximum memory space required to store (buffer) coded data prior to decoding.

Nokia Exhibit 2011, p. 21
Amazon.com v. Nokia
IPR2024-00604

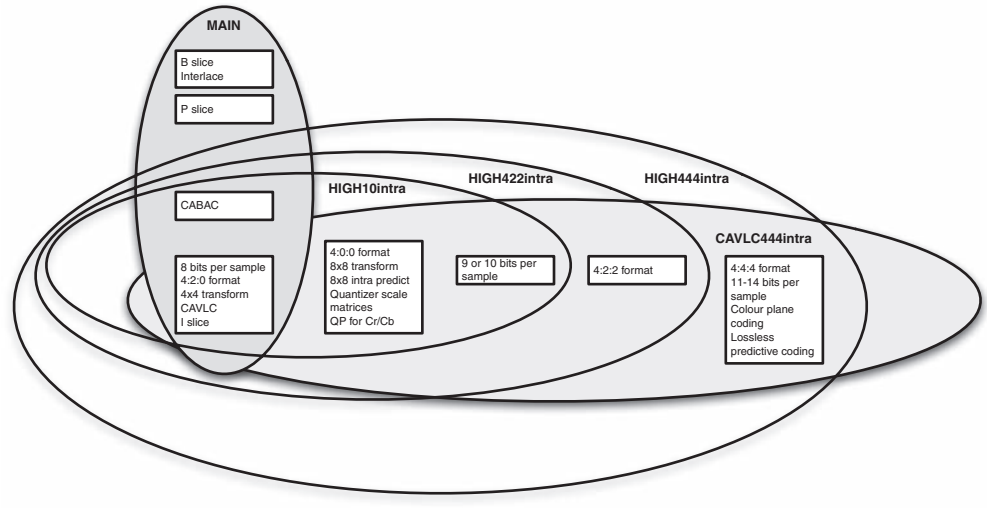


Figure 8.3 Main and Intra Profiles

- Vertical motion vector range (MaxVmvR): the maximum range (+/−) of a vertical motion vector.
- Minimum Compression Ratio (MinCR): the minimum ratio between uncompressed video frames and compressed or coded data size.
- Maximum motion vectors per two consecutive macroblocks (MaxMvsPer2Mb): specified for levels above 3, a constraint on the number of motion vectors (MVx, MVy) that may occur in any two consecutive decoded macroblocks.

In the present version of the standard [i] level numbers range from 1 to 5 with intermediate steps 1.1, 1.2, 1.3, 2.1, etc. A decoder operating at a particular level is expected to be able to handle any of the level constraints at or below that level. For example, a Level 2.1 decoder can handle levels 1, 1.1, 1.2, 1.3, 2 and 2.1. Selected level constraints are shown graphically in Figure 8.4. It is clear that these range from very low, suitable for low-complexity decoders with limited display resolutions, e.g. handheld devices, to very high, suitable for Full High Definition decoders with high resolution displays and significant processing resources.

The parameter MaxFS defines the maximum decoded picture size in macroblocks. This implies certain maximum display resolutions, depending on the aspect ratio. Figure 8.5 shows some examples. For example, MaxFS at Level 1 is equal to 99 macroblocks, which can correspond to 11 × 9 MB or 176 × 144 luma samples, i.e. QCIF resolution. At Levels 2.2 and 3, MaxFS is 1620 macroblocks which can correspond to 45 × 36 MB or 720 × 576 luma samples, '625' Standard Definition. At Levels 4 and 4.1, MaxFS is 8192 macroblocks, which corresponds to approximately 120 × 68 MB or 1920 × 1080 luma samples, '1080p' High Definition. Note that many other aspect ratios are possible within these constraints.

Nokia Exhibit 2011, p. 22
Amazon.com v. Nokia
IPR2024-00604

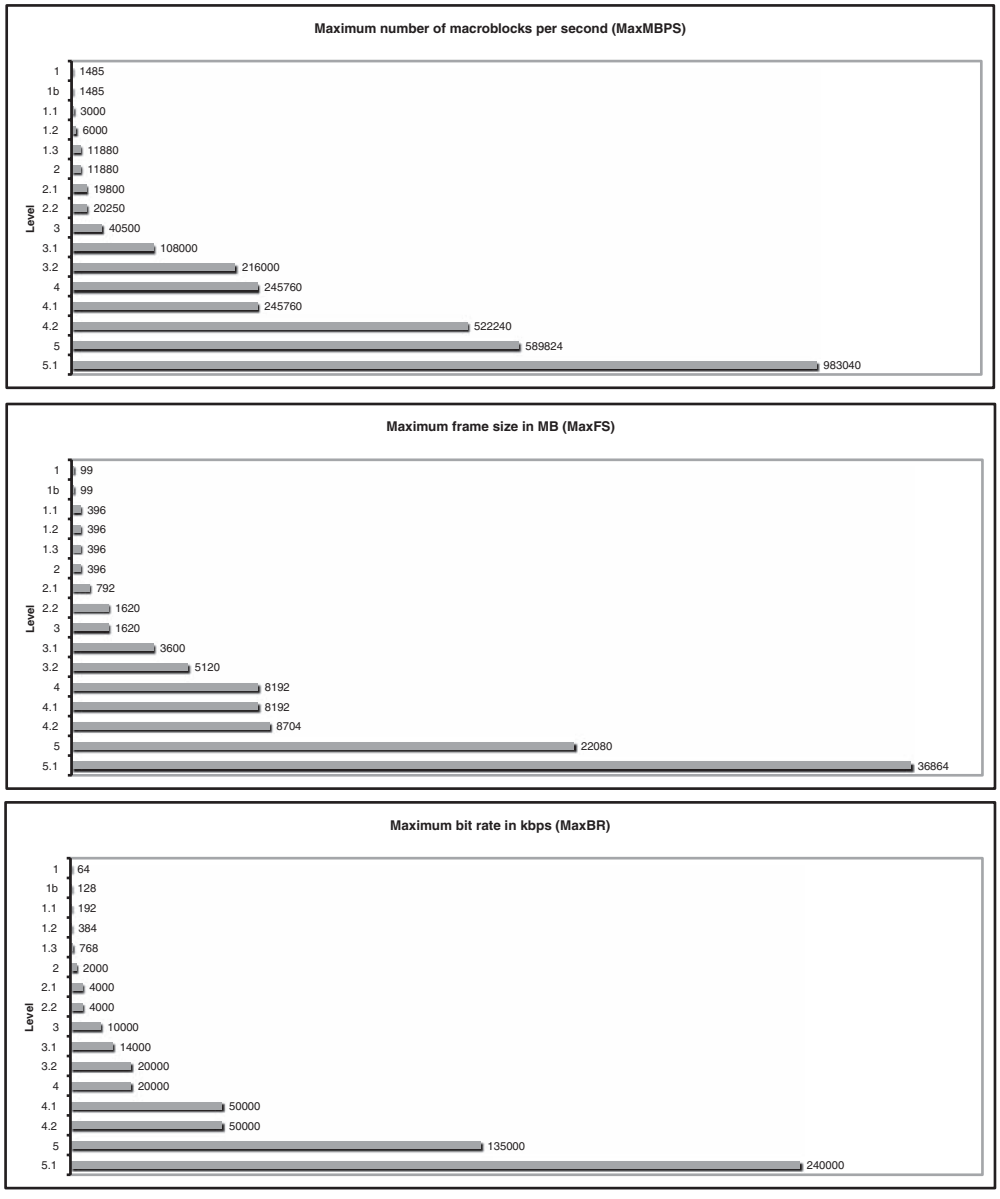


Figure 8.4 Selected Level constraints

The combination of MaxFS, the frame size in macroblocks, and MaxMBPS, the number of macroblocks per second, places a constraint on the maximum frame rate at a particular frame resolution. Table 8.1 lists some examples. Level 1.2 corresponds to CIF resolution at a maximum of 15 frames per second **or** QCIF at a maximum of 60 frames per second. Level 4 corresponds to 1080p High Definition at a maximum of 30 frames per second, **or** 720p High Definition at a maximum of 68 frames per second, and so on.

Nokia Exhibit 2011, p. 23
Amazon.com v. Nokia
IPR2024-00604

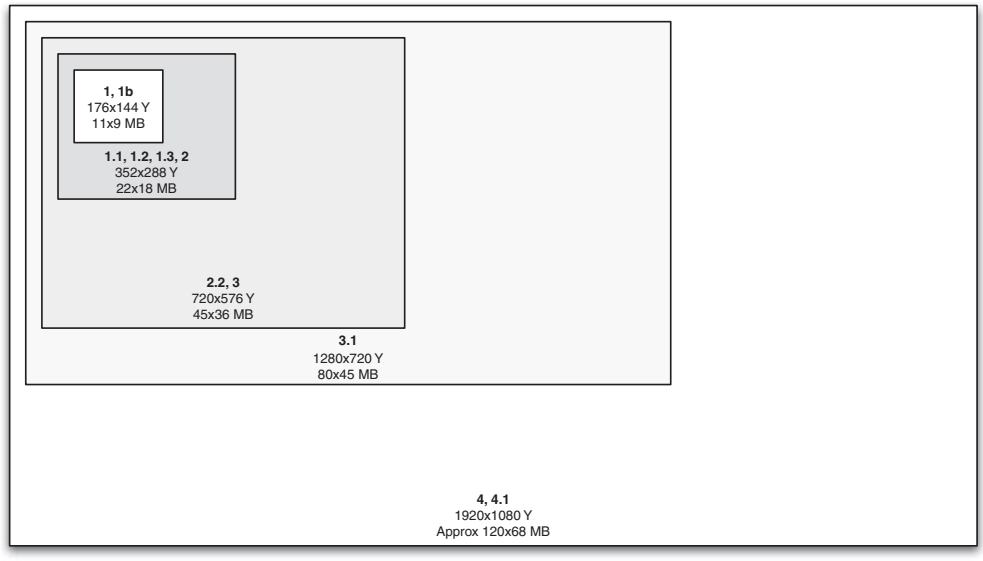


Figure 8.5 Selected display resolutions

Table 8.1 Selected formats, frame rates and levels

Format (luma resolution)	Max frames per second	Level
QCIF (176x144)	15	1, 1b
	30	1.1
CIF (352x288)	15	1.2
	30	1.3, 2
525 SD (720x480)	30	3
625 SD (720x576)	25	3
720p HD (1280x720)	30	3.1
1080p HD (1920x1080)	30	4, 4.1
	60	4.2
4Kx2K (4096x2048)	30	5.1

Nokia Exhibit 2011, p. 24
Amazon.com v. Nokia
IPR2024-00604