PERFORMANCE ANALYSIS AND COMPARISON OF DIRAC VIDEO CODEC WITH H.264 / MPEG-4 PART 10 AVC

by

ARUNA RAVI

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2009

Dedicated to God & to Guru Govardhanji

ACKNOWLEDGEMENTS

I am truly grateful to my thesis advisor, Dr. K. R. Rao, for his guidance and encouragement throughout this thesis. He has been a constant source of inspiration and has helped me improve my work on a regular basis.

I thank Dr. Tim Borer, Lead Engineer, BBC R&D for his valuable advice, motivation and insightful comments towards this research.

I extend my sincere gratitude to Mr. Saurabh Gupta of Nokia Siemens Networks for his advice and support in this research.

I thank Dr. Do Nyeon Kim, Mr. Subrahmanya Samanth and other MPL members for their guidance and suggestions.

I would like to acknowledge Mr. Hemant Malhotra, Mr. Dzung Hoang of JVT-Experts group for their help in obtaining the utility to extract the test sequences.

I thank my graduate advisor, Dr. A. Davis and Dr. K. Alavi, Associate Chairman, Electrical Engineering for being a part of my thesis committee.

Finally, I am indebted my parents, my brother, Gokul and my friend, Ashish for their continuous encouragement and support in all my endeavors. This thesis would not have been possible without them.

July 17, 2009

ABSTRACT

PERFORMANCE ANALYSIS AND COMPARISON OF DIRAC VIDEO CODEC WITH H.264 / MPEG-4 PART 10 AVC

Aruna Ravi , M.S.

The University of Texas at Arlington, 2009

Supervising Professor: K. R. Rao

Dirac is a hybrid motion-compensated state-of-the-art video codec that uses modern techniques such as wavelet transforms and arithmetic coding. [1] It is an open technology, designed to avoid patent infringement and can be used without the payment of license fees. It is well suited to the business model of public service broadcasters as it can be easily recreated for new platforms. Dirac is aimed at applications ranging from HDTV (high definition television) to web streaming.

H.264, MPEG-4 part-10 or AVC [12 - 16], is the latest digital video codec standard which has proven to be superior than earlier standards in terms of compression ratio, quality, bit rates and error resilience. However unlike Dirac, it requires payment of license / patent fees.

The objective of this thesis is to implement Dirac video codec (encoder and decoder) [1] based on several input test sequences, and compare its performance with H.264 / MPEG-4 Part 10 AVC. Analysis has been done on Dirac and H.264 using QCIF, CIF and SDTV video test sequences as input and the results have been recorded graphically for various parameters including compression ratio, bit rate, PSNR, SSIM and MSE. In these tests, encoding and decoding has been performed for quality factor ranging from 0 – 10 and for lossless compression. Apart from this, comparison between Dirac and H.264's performance has been implemented at various constant 'target' bit rates ranging from 10KBps to 200KBps.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

ACRONYMS AND ABBREVIATIONS

AAC: Advanced audio coding

AFF: Advanced file format

AVC: Advanced video coding

ATSC: Advanced television systems committee

BBC: British Broadcasting Corporation

CABAC: Context-based adaptive binary arithmetic coding

CAVLC: Context-based adaptive variable length coding

CBR: Constant bit rate

CIC: Cascaded Integrator-Comb

CIF: Common intermediate format

DCT: Discrete cosine transform

EBU: European Broadcasting Union

FIR: Finite impulse response

GOP: Group of picture(s)

HDTV: High definition television

HF: High frequency

HH: High - high

HL: High - low

IEC: International Engineering Consortium

IIR: Infinite impulse response

ISO: International Standards Organization

ITU: International Telecommunication Union

JVT: Joint Video Team

LF: Low frequency

LH: Low - high

LL: Low - low

MBAFF: MB-Level adaptive frame / field

MCPE: Motion-compensated prediction error

MPEG: Moving Picture Experts Group

MSE: Mean square error

MV: Motion vector

OBMC: Overlapped block-based motion compensation

PCM: Pulse-code modulation

QCIF: Quarter common intermediate format

QF: Quality factor

RDO: Rate distortion optimization

SAD: Sum of the absolute difference

SDTV: Standard definition television

SMPTE: Society of Motion Picture and TV Engineers

SNR: Signal to noise ratio

SSIM: Structural similarity index

VC: Video codec

VCEG: Video Coding Experts Group

WMV: Windows Media Video

YUV: Luminance and chrominance color components

CHAPTER 1

INTRODUCTION

1.1 Introduction

In this era of digital communications, video compression has become a necessity and plays an integral role that aids the widespread use of multimedia over the Internet. In the digital world, with the rapid growth of the Internet and the introduction of high-definition media, compression is used to exploit limited storage and transmission capacity as efficiently as possible.

Dirac is an open and royalty-free video codec developed by the BBC, named in honor of the British scientist Paul Dirac. It aims to provide high-quality video compression from web video up to HD,[4] and as such competes with existing formats such as H.264 [12 - 16] and WMV [24] [25]. Dirac was used internally by the BBC to transmit HDTV pictures at the Beijing Olympics in 2008. [18] [19]

Dirac can compress any size of picture from low-resolution QCIF (176x144 pixels) to HDTV (1920x1080) and beyond, similar to common video codecs such as the ISO/IEC Moving Picture Experts Group (MPEG)'s MPEG-4 Part 2 [26][27] and Microsoft's WMV 9 [24] [25]. However, it promises significant savings in data rate and improvements in quality over these codecs, by some claims even superior to those promised by the latest generation of codecs such as H.264/MPEG-4 AVC or SMPTE's VC-1. [20]

Dirac's implementers make the preliminary claim of "a two-fold reduction in bit rate over MPEG-2 for high definition video",[2] an estimate which would put the design in about the same class of compression capability as the latest standards such as H.264/MPEG-4 AVC and VC-1 [29].

Dirac employs wavelet compression, instead of the discrete cosine transforms used in most other codecs (such as H.264/MPEG-4 AVC or SMPTE's VC-1). Dirac is one of several projects that have applied wavelets to video compression. Wavelet compression has already proven its viability in the JPEG 2000 compression standard for photographic images. [20]

There are two parts in the Dirac development process: (A) a compression specification for the bit stream and decoder, and (B) software for compression and decompression. The software [4] [28] is not intended simply to provide reference coding and decoding - It is a prototype implementation that can freely be modified and deployed. The decoder in particular is designed to be fast and more agile. The resulting specification is simple and straightforward to implement and optimized for real-time performance. [1]

In this thesis, the performance of the Dirac video codec has been thoroughly analyzed for QCIF, CIF and SDTV input video streams and compared with the performance of H.264 video codec at constant bit rates.

1.2 Thesis Outline

Chapter 2 and chapter 3 give an overview of the Dirac video codec and the H.264 video codec. The architecture and the various features of each codec along with the advantages, disadvantages, features and applications are discussed in detail.

Chapter 4 explains the process of encoding the input video sequence and decoding of the compressed video by the Dirac encoder and decoder.

Chapter 5 presents the test data, results and conclusions obtained using the proposed method of implementation. Suggestions for future work are also outlined.

CHAPTER 2

OVERVIEW OF DIRAC VIDEO CODEC

2.1 Dirac architecture

In the Dirac codec, image motion is tracked and the motion information is used to make a prediction of a later frame. A transform is applied to the prediction error between the current frame and the previous frame aided by motion compensation and the transform coefficients are quantized and entropy coded. [1]

Temporal and spatial redundancies are removed by motion estimation, motion compensation and discrete wavelet transform respectively. Dirac uses a more flexible and efficient form of entropy coding called arithmetic coding which packs the bits efficiently into the bit stream. [1]

2.2 Dirac encoder

The Dirac encoder [1] [30] has the architecture as shown in Fig. 2.1. The entire compressed data is packaged in a simple bytestream. This has synchronization, permitting access to any frame quickly and efficiently - making editing simple. The structure is such that the entire bytestream can be packaged in many of the existing transport streams. This feature allows a wide range of coding options, as well as easy access to all the other data transport systems required for production or broadcast metadata.

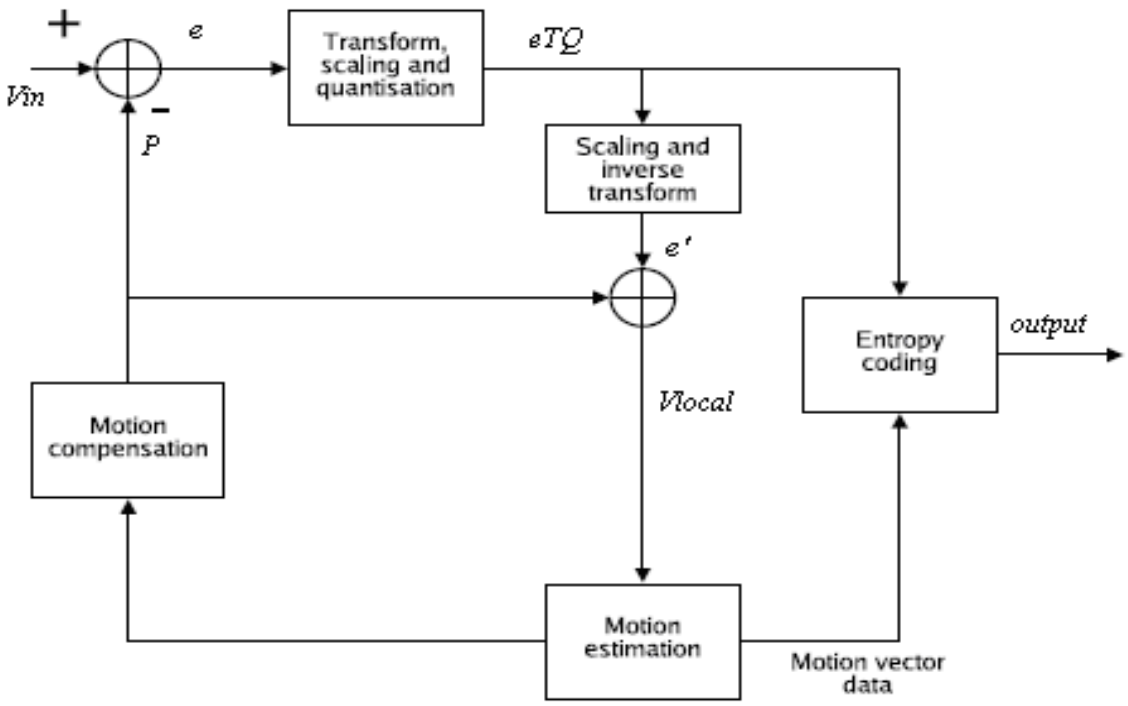


Fig. 2.1: Dirac encoder architecture [1] [2]

Notations: V_{in} : Input video frame; P: Previous frame (Motion compensated); Motion compensated prediction error(MCPE): $e = V_{in} - P$; e_{TQ} : MCPE after wavelet transformation and quantization; e' : MCPE after inverse transformation; $V_{local} = P + e'$

2.3 Dirac decoder

The Dirac decoder [1] [30] performs the inverse operations of the encoder as depicted in Fig. 2.2.

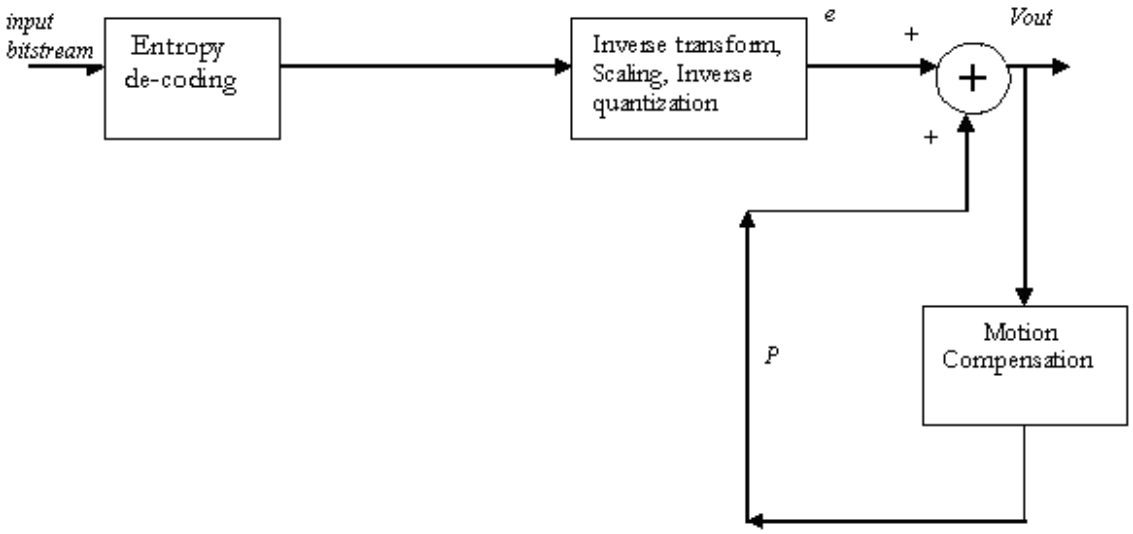


Fig. 2.2: Dirac decoder architecture

Notations: *P*: Motion compensated previous frame, *e:* Motion compensated prediction error (MCPE): Prediction error after inverse wavelet transformation and quantization; V_{out}: Output video frame

2.4 Features of Dirac [4]

The features of Dirac are listed below:

- Direct support of multiple picture formats
- 4K e-cinema through to quarter common intermediate format (QCIF)
- Supports I-frame only upto long group of picture (GOP) structures
- Direct support of multiple chroma formats e.g. 4:4:4/4:2:2/4:2:0
- Direct support of multiple bit depths e.g. 8 bit to 16 bit
- Direct support of interlace via metadata
- Direct support of multiple frame rates from 23.97 fps to 60fps
- Definable pixel aspect ratios
- Definable "Clean area" for inputs within larger containers
- Definable signal ranges and offsets
- Multiple color spaces with metadata
- Color primaries and color matrices
- 32 bit frame numbers (at 60 fps) in I frame
- Choice of multiple wavelet filters (including filters optimized for down-conversion)
- Definable wavelet depth

Dirac 1.0.2 was released on February 12, 2009. [4] The new features of this release are:

- Compliance with Dirac bytestream specification 2.2.3
- Improved motion estimation.
- Improved compression efficiency at low bit-rates and small pictures
- Fixed bugs in DC sub-band handling
- New 4:2:2 <-> 4:2:0 conversion utilities for interlaced video.
- DirectShow Filter to play back Dirac v2.2.3 files in Window Media Player and MPlayer Classic [4]

2.5 Advantages of Dirac [2]

The advantages of Dirac are listed below:

- Dirac is an open technology - removing licensing costs on software, hardware and content flow.
- Dirac's technical flexibility offers a versatile package, facilitating ease of operation over many applications and therefore saving money.
- Capability of compressing high resolution files for production, broadcast content, and web 2.0 applications. Compression can be either lossless or visually lossless.
- Can exploit lossy compression using long group of picture (GOP) formats for broadcasting and other distribution outlets.

- The quality or compression efficiency is similar to that of AVC/H.264 [12] but without the encoding complexity or license burden.
- Can achieve good subjective compression performance by using a few tools guided by psycho visual criteria.
- More freedom for future implementations and therefore more scope to improve performance
- Can improve quality by concentrating more on large errors and de-emphasizing high frequency errors.
- Very comparable to other state-of-the-art codecs despite simple toolset.
- Relative simplicity and clean architecture support high performance.
- Offers good quality at low bit rates, leading to lower costs.
- Offers good quality with low delay - ideal for live broadcast applications in studios and outside broadcasts.
- Suitable for everything from internet streaming to HDTV and electronic cinema.
- Speed of playback can be improved as the software is optimized.

2.6 Applications of Dirac [2]

The applications of Dirac are listed below:

- Broadcasting
- Clip distribution
- Live streaming video
- Pod casting
- Creative archive
- Peer to peer transfers
- HDTV with SD (standard definition) simulcast capability
- Higher density channel packing
- Desktop production
- News links
- Archive storage
- Digital intermediate film out file storage
- PVRs (personal video recorder)
- Multilevel Mezzanine coding (aka SMPTE VC-2): Eg - 3 GBit/s into 1.5 GBit/s, 1.5 GBit/s into 270 MBit/s

Table 2.1: Algorithmic element comparison of Dirac with other codecs [35]

Algorithmic Element	MPEG-2 Video	MPEG-4 AVC	SMPTE VC-1	Dirac
	(H.262)	*(H.264)*	*(Windows Media Video 9)*	
Intra Prediction	None: MB encoded DC predictors	4x4 spatial 16x16 spatial I-PCM	Frequency domain coefficient	4x4 spatial
Picture coding type	Frame Field Picture AFF	Frame Field Picture AFF MB AFF	Frame Field Picture AFF MB AFF	Frame
Motion compensation block size	16x16 16x8, 8x16	16x16 16x8, 8x16 8x8 8x4, 4x8 4x4	16x16 8x8	4x4
Motion vector precision	Full pel Half pel	Full pel Half pel Quarter pel	Full pel Half pel Quarter pel	1/8 pel
P Frame type	Single reference	Single reference Multiple reference	Single reference Intensity compensation	Single reference Multiple reference
B Frame type	One reference each way	One reference each way Multiple reference Direct & Spatial direct Weighted prediction	One reference each way	One reference each way, Multiple reference
In-loop filters	None	De-blocking	De-blocking Overlap transform	None
Entropy coding	VLC	CAVLC, CABAC	Adaptive VLC	Arithmetic coding
Transform	8x8 DCT	4x4 integer DCT 8X8 integer DCT	4x4 integer DCT 8X8 integer DCT 8x4 & 4x8 integer DCT	4x4 Wavelet transform
Other	Quantization scaling matrices	Quantization scaling matrices	Range reduction Instream post-processing control	Quantization scaling matrices

2.7 Summary

In this chapter, an overview of the Dirac video codec was presented. The architecture of the encoder and the decoder were discussed in detail. The various features, advantages and the numerous applications of Dirac were also discussed. A detailed comparison of the algorithmic elements of Dirac with other state-of-the-art codecs such as MPEG-2, MPEG-4 and SMPTE VC-1 was also presented.

CHAPTER 3

OVERVIEW OF H.264 VIDEO CODEC

3.1 H.264/AVC - Introduction

H.264 is an open, licensed standard that supports the most efficient video compression techniques available today, and is equivalent to MPEG-4 Part 10, or MPEG-4 AVC (Advanced Video Coding) [45]. As of May 2009, it is the latest block-oriented motion-compensation-based codec standard developed by the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG), and it is the product of a partnership effort known as the Joint Video Team (JVT). It is a standard capable of providing good video quality at substantially lower bit rates, promises better rate-distortion performance and compression efficiency than previous standards such as MPEG-2, H.263, or MPEG-4 Part 2 Visual without increasing the complexity of design.[45] It has simple syntax specifications, yet enough flexibility to be applied to a wide variety of applications such as video broadcasting, video streaming, video conferencing, D-Cinema, HDTV. It is network friendly and demonstrates a balance between coding efficiency, implementation complexity and cost - based on state-of the-art in VLSI design technology. [12] H.264 uses hybrid block based video compression techniques such as transformation for reduction of spatial correlation, quantization for bit-rate control, motion compensated prediction for reduction of temporal correlation and entropy coding for reduction in statistical correlation.

It includes intra-picture prediction, a 4x4 integer transform, multiple reference pictures, variable block sizes, a quarter pel precision for motion compensation, an in-loop de-blocking filter, and improved entropy coding. [46]

H.264 / MPEG-4 has a layered structure that consists of two layers: Network Abstraction Layer (NAL) and Video Coding Layer (VCL). The NAL abstracts the VCL data and has header information about the VCL format. It is appropriate for conveyance by the transport layers or storage media. The NAL unit (NALU) defines a generic format for use in both packet based and bit-streaming systems. The VCL is a core coding layer and concentrates on attaining maximum coding efficiency.

The layered structure supports 4:2:0 chroma sampling picture format including QCIF and CIF formats shown in Fig. 3.1.

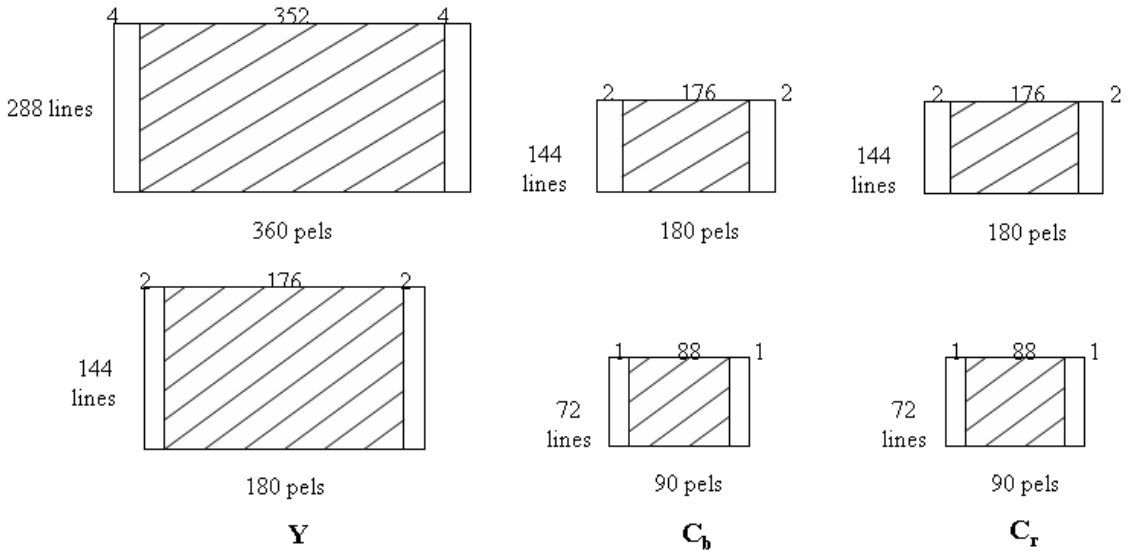


Fig. 3.1: Supporting picture format – 4:2:0 chroma sampling, Top: CIF, Bottom: QCIF [12]

3.2 H.264/AVC profiles [12] [45]

The H.264 standard is a "family of standards", that include the following sets of capabilities, referred to as "profiles", targeting specific classes of applications:

- Baseline Profile (BP): The baseline profile includes I- and P-slice coding, enhanced error resilience tools (flexible macro-block ordering (FMO), arbitrary slices and redundant slices), and CAVLC. Primarily for lower-cost applications with limited computing resources, this profile is used widely in videoconferencing and mobile applications. Among the three profiles, it offers the least coding efficiency.
- Main Profile (MP): Originally intended as the mainstream consumer profile for broadcast and storage applications. The Main profile includes I-, P- and B-slices, interlaced coding, CAVLC and CABAC. This profile is designed to provide the highest possible coding efficiency.
- Extended Profile (XP): The extended profile is a superset of the baseline profile. It includes B-, SP- (switched prediction) and SI- (switched intra) slices, data partitioning, and interlaced coding tools besides tools of the baseline profile. It is thus more complex but also provides better coding efficiency. It is intended as the streaming video profile, has relatively high compression capability and robustness to data losses and server stream switching.

- High Profile (HiP): The primary profile for broadcast and disc storage applications, particularly for high-definition television. There are four High Profiles (Fidelity range extensions). [50]They are:

– High Profile - to support the 8-bit video with 4:2:0 sampling for applications using high resolution.
– High 10 Profile - to support the 4:2:0 sampling with up to 10 bits of representation accuracy per sample.
– High 4:2:2 Profile - to support up to 4:2:2 chroma sampling and up to 10 bits per sample.
– High 4:4:4 Profile - to support up to 4:4:4 chroma sampling, up to 12 bits per sample, and integer residual color transform for coding RGB signal.

Table 3.1: H.264 profiles for particular applications [12]

Profile	Applications
Baseline	Video Conferencing Videophone
Main	Digital Storage Media Television Broadcasting
Extended	Streaming Video
High	Content contribution Content distribution Studio editing Post processing

Table 3.2: H.264 / MPEG-4 Part 10 profile specifications [12]

	Baseline	Main	Extended	High
I & P Slices	X	X	X	X
Deblocking Filter	X	X	X	X
¼ Pel Motion Compensation	X	X	X	X
Variable Block Size (16x16 to 4x4)	X	X	X	X
CAVLC/UVLC	X	X	X	X
Error Resilience Tools – Flexible MB Order, ASO, Red. Slices	X		X	
SP/SI Slices			X	X
B Slice		X	X	X
Interlaced Coding		X	X	X
CABAC		X		X
Data Partitioning			X	

The common coding parts for the profiles are listed below: [12]

- I slice (Intra-coded slice): coded by using prediction only from decoded samples within the same slice
- P slice (Predictive-coded slice) : coded by using inter prediction from previously decoded reference pictures, using at most one motion vector and reference index to predict the sample values of each block
- CAVLC (Context-based Adaptive Variable Length Coding) for entropy coding

The common coding parts for the baseline profile are listed below:

- Common parts : I slice, P slice, CAVLC
- FMO Flexible macro block order: macro-blocks may not necessarily be in the raster scan order. The map assigns macro-blocks to a slice group
- ASO Arbitrary slice order : the macro-block address of the first macro-block of a slice of a picture may be smaller than the macro-block address of the first macro-block of some other preceding slice of the same coded picture
- RS Redundant slice : This slice belongs to the redundant coded data obtained by same or different coding rate, in comparison with previous coded data of same slice

The common coding parts for the main profile are listed below:

- Common parts : I slice, P slice, CAVLC
- B slice (Bi-directionally predictive-coded slice) : the coded slice by using inter prediction from previously-decoded reference pictures, using at most two motion vectors and reference indices to predict the sample values of each block
- Weighted prediction : scaling operation by applying a weighting factor to the samples of motion-compensated prediction data in P or B slice
- CABAC (Context-based Adaptive Binary Arithmetic Coding) for entropy coding

The common coding parts for the extended profile are listed below:

- Common parts : I slice, P slice, CAVLC
- SP slice : specially coded for efficient switching between video streams, similar to coding of a P slice
- SI slice : switched, similar to coding of an I slice
- Data partition : the coded data is placed in separate data partitions, each partition can be placed in different layer unit
- Flexible macro-block order (FMO), arbitrary slice order (ASO)
- Redundant slice (RS), B slice
- Weighted prediction

The specific coding parts for the various profiles are represented by Fig. 3.2 shown below:

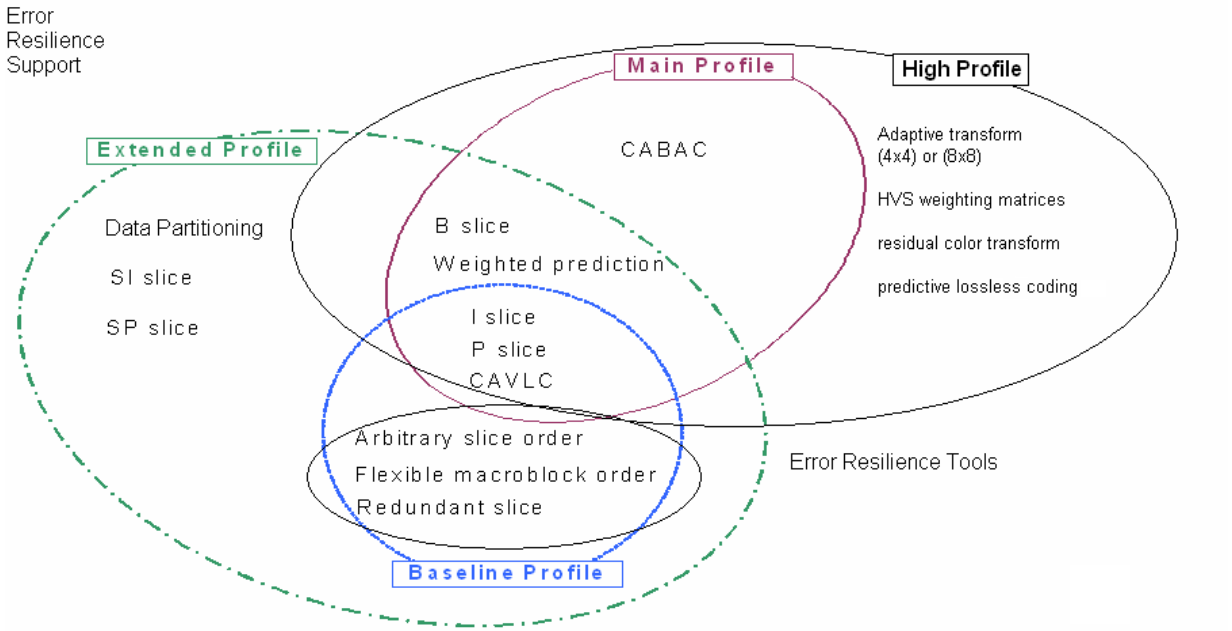


Fig. 3.2: Specific coding parts for H.264 profiles [12]

3.3 H.264 encoder and decoder

The H.264 / MPEG-4 AVC architecture is shown in Fig. 3.3.

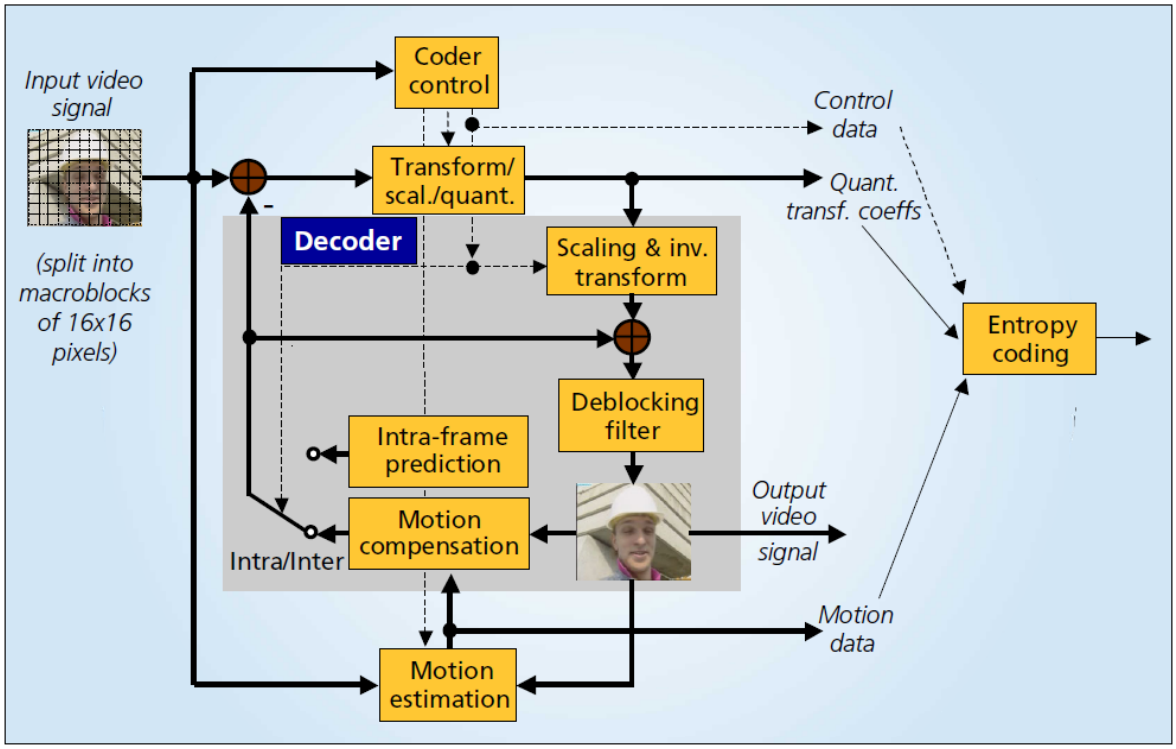


Fig. 3.3: Basic H.264 / AVC macro-block coding structure [51]

In a series of frames, video data can be reduced by methods such as difference coding, which is used by most video compression standards including H.264. In difference coding, a frame is compared with a reference frame and only pixels that have changed with respect to the reference frame are coded. In this way, the number of pixel values that are coded and sent is reduced.

The amount of encoding can be further reduced if detection and encoding of differences is based on blocks of pixels (macro-blocks) rather than individual pixels; therefore, bigger areas are compared and only blocks that are significantly different are coded. The overhead associated with indicating the location of areas to be changed is also reduced.

The basic encoding procedure is shown in the form of block diagram in Fig. 3.4.

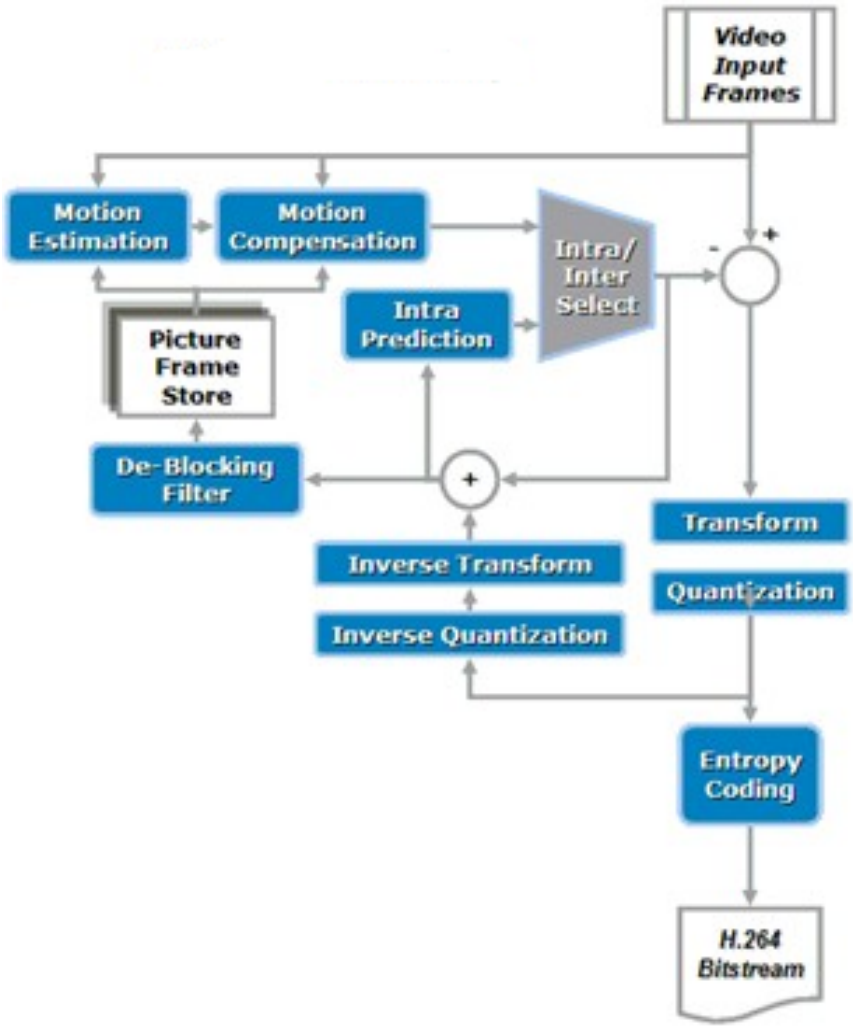


Fig. 3.4: Basic block diagram of H.264 encoder [47]

The functions of different blocks of the H.264 encoder are described below:

Transform: A 4x4 multiplier-free integer transform is used and the transform coefficients are explicitly specified in AVC and allow it to be perfectly invertible. Its hierarchical structure is a 4 x 4 Integer DCT and Hadamard transform. The Hadamard transform is applied only when (16x16) intra prediction mode is used with (4x4) integer DCT. MB size for chroma depends on 4:2:0, 4:2:2 and 4:4:4 formats (see Fig. 3.5). [12]

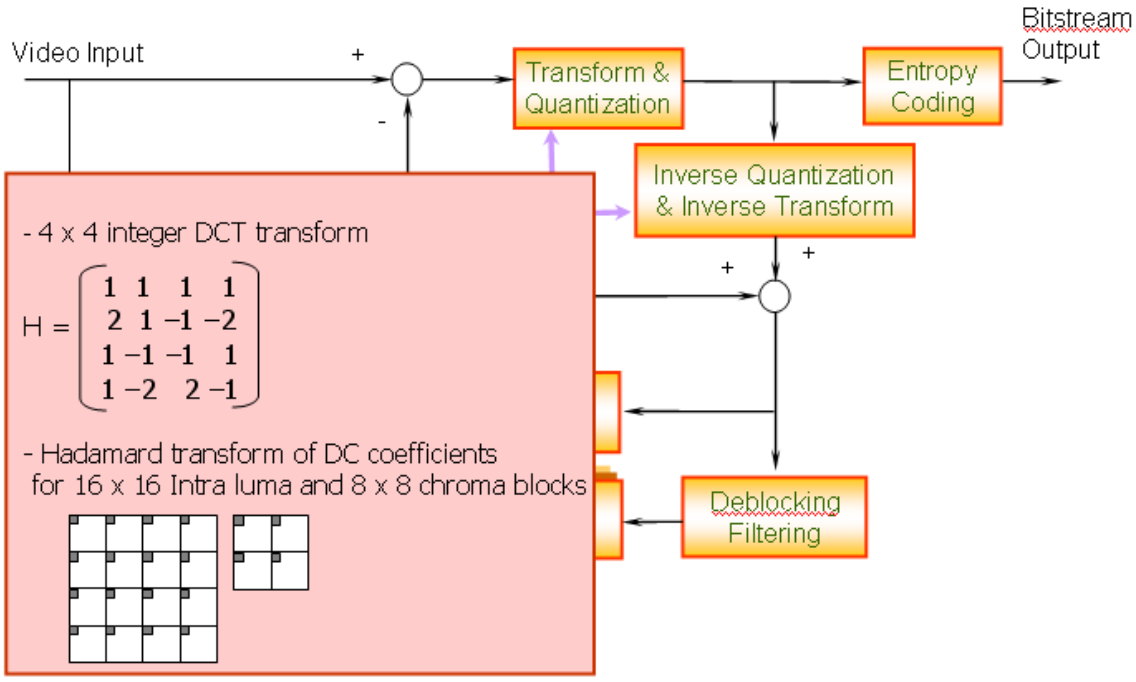


Fig. 3.5: Block diagram emphasizing transform [12]

Fig. 3.6 shows the assignment of the DC indices to the 4 x 4 luma block. The numbers 0, 1, …15 are the coding order for (4x4) integer DCT and (0,0), (0,1), (0,2), …, (3,3) are the DC coefficients of each 4x4 block.

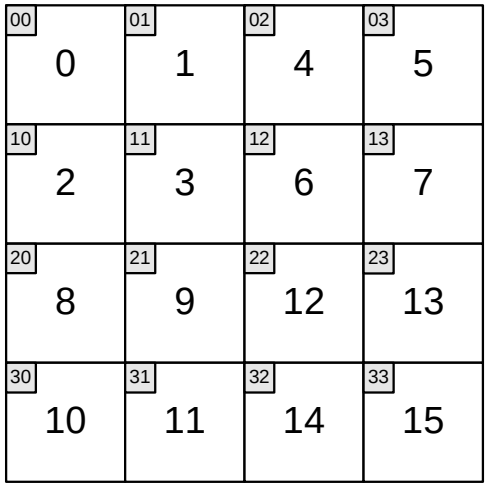


Fig. 3.6: Assignment of indices of the DC (dark samples) to luma 4 x 4 blocks [12]

The 4x4 IntDCT (see Fig. 3.7) can be applied as shown below: [12]

$$Y=(C_f \times C_f^{T}) \otimes E_f \qquad (3.1)$$

$$Y=\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

$$a=\frac{1}{2}, b=\sqrt{\frac{2}{5}}, d=\frac{1}{2}$$

where X : input pixels, Y : output coefficients, $\otimes$ represents element by element multiplication.

The inverse 4x4 DCT can be represented by the following equation: [12]

$$X' = C_i^T (Y \otimes E_i) C_i \tag{3.2}$$

$$where\ C_i = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}; E_i = \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

The 16 DC coefficients of the 16 (4x4) blocks are transformed using Walsh Hadamard transform as shown below: [12]

$$Y_D = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_{D00} & x_{D01} & x_{D02} & x_{D03} \\ x_{D10} & x_{D11} & x_{D12} & x_{D13} \\ x_{D20} & x_{D21} & x_{D22} & x_{D23} \\ x_{D30} & x_{D31} & x_{D32} & x_{D33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \right) // 2 \tag{3.3}$$

where // represents rounding to the nearest integer.

The Walsh – Hadamard transform can be represented as follows:

$$Y_D = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} DC_{00} & DC_{01} \\ DC_{10} & DC_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \text{4:2:0} \tag{3.4}$$

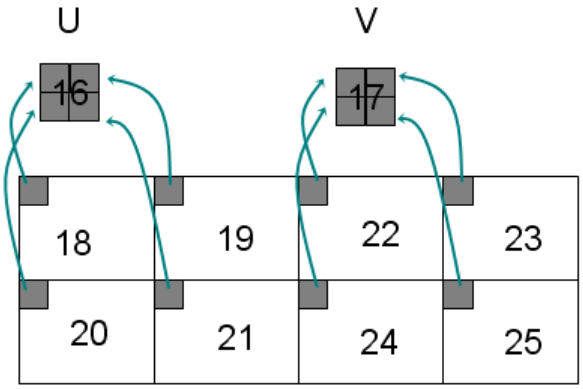


Fig. 3.7: Chroma DC coefficients for 4x4 IntDCT [12]

Scaling and Quantization: Multiplication operation for the exact transform is combined with the multiplication of scalar quantization. The scale factor for each element in each sub-block varies as a function of the quantization parameter associated with the macro-block that contains the sub block, and as a function of the position of the element within the sub-block. The rate-control algorithm in the encoder controls the value of quantization parameter. The encoder performs post-scaling and quantization.

Quantization and scaling at the encoder can be represented by the following equation: [12]

$$B_{ij} = A_{ij}\, round\left(\frac{SF_{ij}}{Qstep}\right) \tag{3.5}$$

where A : quantizer input, B : quantizer output, $Qstep$: quantization parameter, SF : scaling term

Fig. 3.8 shows the transform, scaling and quantization blocks at the encoder part of H.264 / MPEG-4.

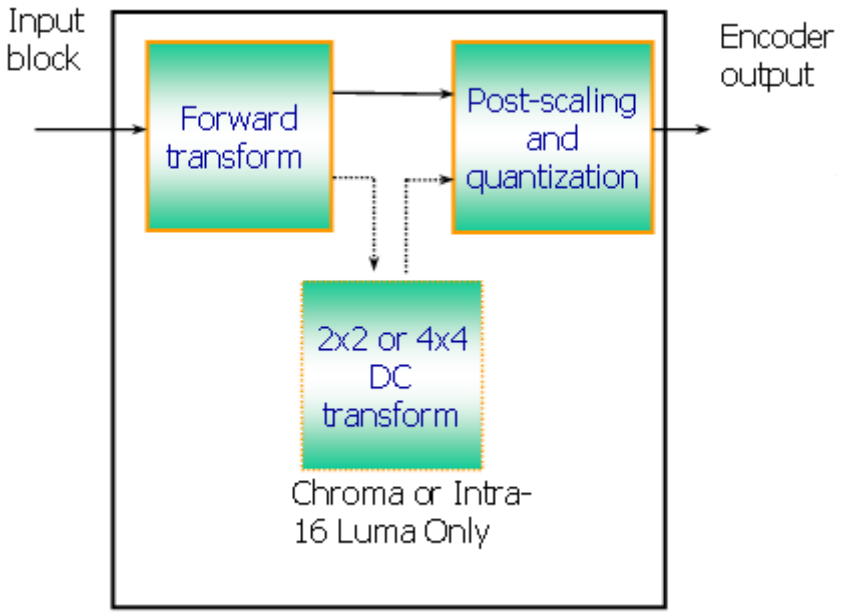


Fig. 3.8: Transform, scaling and quantization at H.264 encoder [12]

Entropy coding: The H.264 AVC includes two different entropy coding methods for coding quantized transform coefficients namely CAVLC (Context-based Adaptive Variable Length Coding) and CABAC (Context-based Adaptive Binary Arithmetic Coding).

CAVLC handles the zero and +/-1 coefficients based on the levels of the coefficients. The total numbers of zeros and +/-1 are coded. For the other coefficients, their levels are coded. [12] Context adaptive VLC of residual coefficients make use of run-length encoding.

CABAC on the other hand, utilizes arithmetic coding. Also, in order to achieve good compression, the probability model for each symbol element is updated. Both MV and residual transform coefficients are coded by CABAC. [12]

Adaptive probability models are used and are restricted to binary arithmetic coding for simple and fast adaptation mechanism. Symbol correlations are exploited by using contexts.

CABAC increases compression efficiency by 10% over CAVLC but computationally more intensive.[12] Typically CABAC provides 10-15 % reduction in bit rate compared to CAVLC, for the same PSNR [50] as shown in Fig. 3.9.

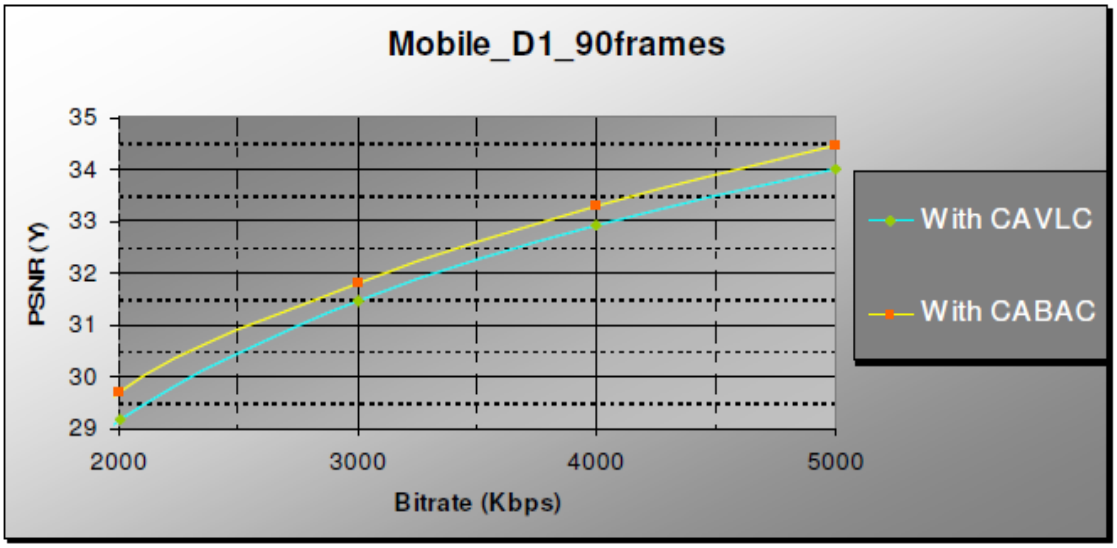


Fig. 3.9: Comparison of CAVLC and CABAC [50]

All syntax elements other than residual transform coefficients are encoded by the Exp-Golomb codes (Universal Variable Length Codes (UVLC)). Some of the features of the Exp-Golomb codes [50] are:

- Variable length codes with a regular construction
- Easy to parse, optimal for one / two-sided geometric pdfs (probability density functions)
- Syntax elements covered – mb_type, sub_mbtype, MVD, ref_idx etc.

The UVLC can be represented as [M Zeroes] [1] [M bits of INFO]. The first codeword has no leading zeroes or trailing information. Code words 1 and 2 have a single-bit INFO field, code words 3-6 have a two-bit INFO field and so on (where INFO = code_num + 1 – 2M). The length of each Exp-Golomb codeword is (2M+1) bits where M = Floor (Log_2 [code_num + 1]). [12]

There are two types of scan orders to read the residual data (quantized transform coefficients) namely, zig-zag and alternate scan as shown in Fig. 3.10. [12]

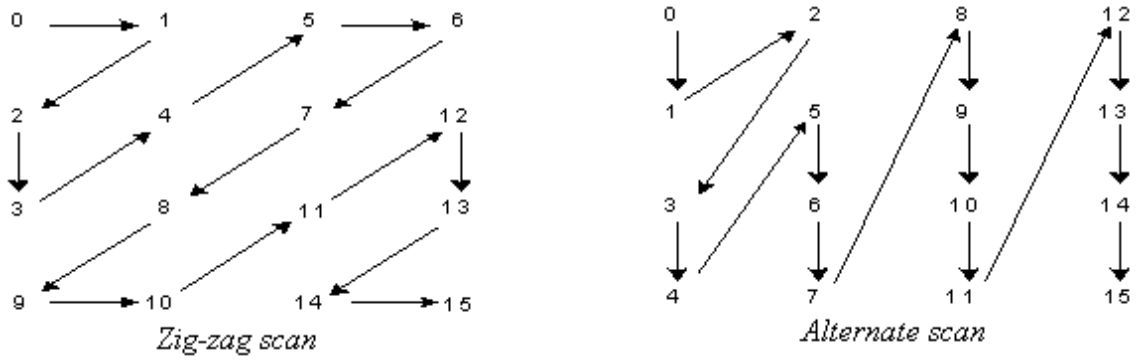


Fig. 3.10: H.264 scan orders to read residual data [12]

Deblocking filter: Coarse quantization of the block-based image transform produces disturbing blocking artifacts at the block boundaries of the image. Motion compensation of the macro-block by interpolation of data from previous reference frames might never give a perfect match and discontinuities appear at the edges of the copied blocks. When the later P/B frames reference these images having blocky edges, the blocking artifacts further propagate to the interiors of the current block worsening the situation further. [50]

The best way to deal with these artifacts is to filter the blocky edges to have a smoothed edge. This filtering process is known as the “deblock” filtering. The In-Loop deblock filter not only smoothens the blocky edges but also helps to increase the rate-distortion performance. After this, the frame decode process is carried out which ensures that all the top/left neighbors have been fully reconstructed and available as inputs for de-blocking the current MB. This is applied to all 4x4 blocks except at the boundaries of the picture. Filtering for block edges of any slice can be selectively disabled by means of flags. [50] Vertical edges are filtered first (left to right) followed by the horizontal edges (top to bottom) as shown in Fig. 3.11.

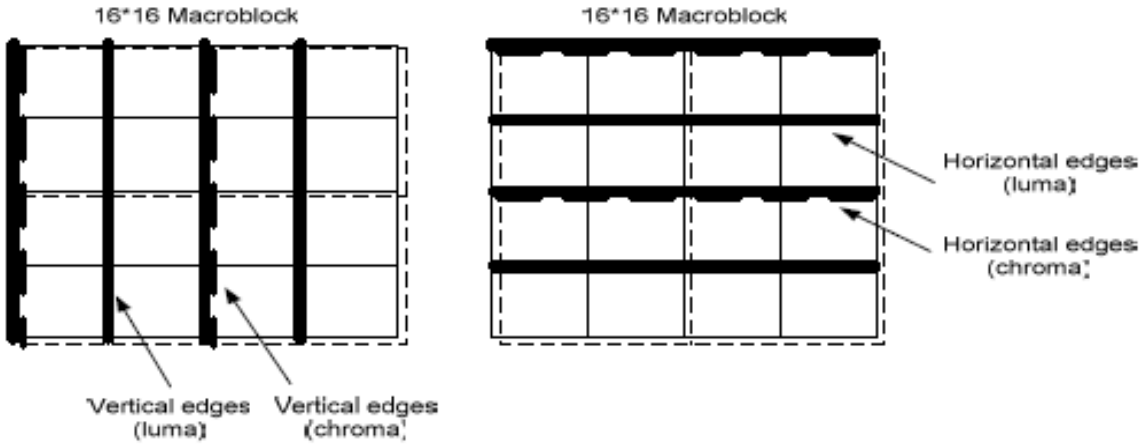


Fig. 3.11: De-blocking filter process [50]

This filter operates on a macro-block after motion compensation and residual coding, or on a macro-block after intra-prediction and residual coding, depending whether the macro-block is inter-coded or intra-coded. The results of the loop filtering operation are stored as a reference picture.

The loop filter operation is adaptive in response to several factors such as the quantization parameter of the current and neighboring macro-blocks, the magnitude of the motion vector and the macro-block coding type. [50]

Mode decision: It determines the coding mode for each macro-block. Mode decision to achieve high efficiency may use rate distortion optimization. Mode decision works with rate control algorithm and the outcome is the best-selected coding mode for a macro-block.

Intra prediction: Intra prediction is a process of using the pixel data predicted from the neighboring blocks for the purpose of sending information. The H.264/AVC does prediction in the spatial domain rather than in frequency domain. It uses the reconstructed but unfiltered macro-block data from the neighboring macro-blocks to predict the current macro-block coefficients. Intuitively, the prediction of pixels from the neighboring pixels (top/left) of macro-blocks in the raster scan would be much efficient as compared to the prediction of the transform domain values. Prediction from samples in the pixel domain helps in better compression for intra blocks in an inter frame. It also offers flexible bit-rate control by providing the flexibility to eliminate redundancies across multiple directions. [50] In High Profile, there is also adaptive directional intra prediction for 8x8 blocks. The intra prediction block is shown in Fig. 3.12.

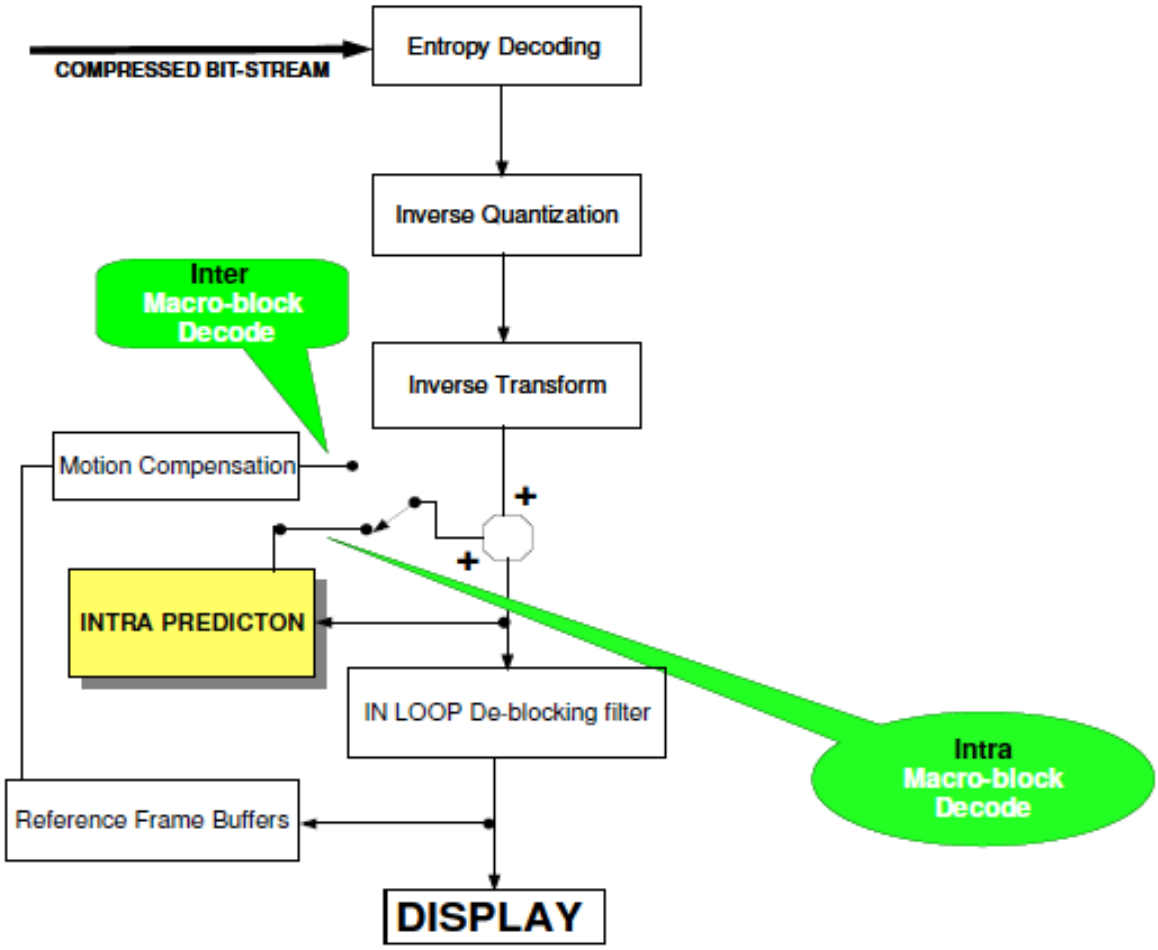


Fig. 3.12: Intra-prediction block [50]

For luminance pixels with 4x4 partitions, 9 intra-prediction modes are defined and when a 16x16 partition is used, four intra-prediction modes are defined. The 16 x 16 mode is aimed to provide better compression for flat regions of a picture at lower computational costs. This mode is also helpful to avoid the gradients that show up in flat regions of the picture quantized with high quantization parameters.

The mode decisions for 4x4 blocks are shown in Fig. 3.13.

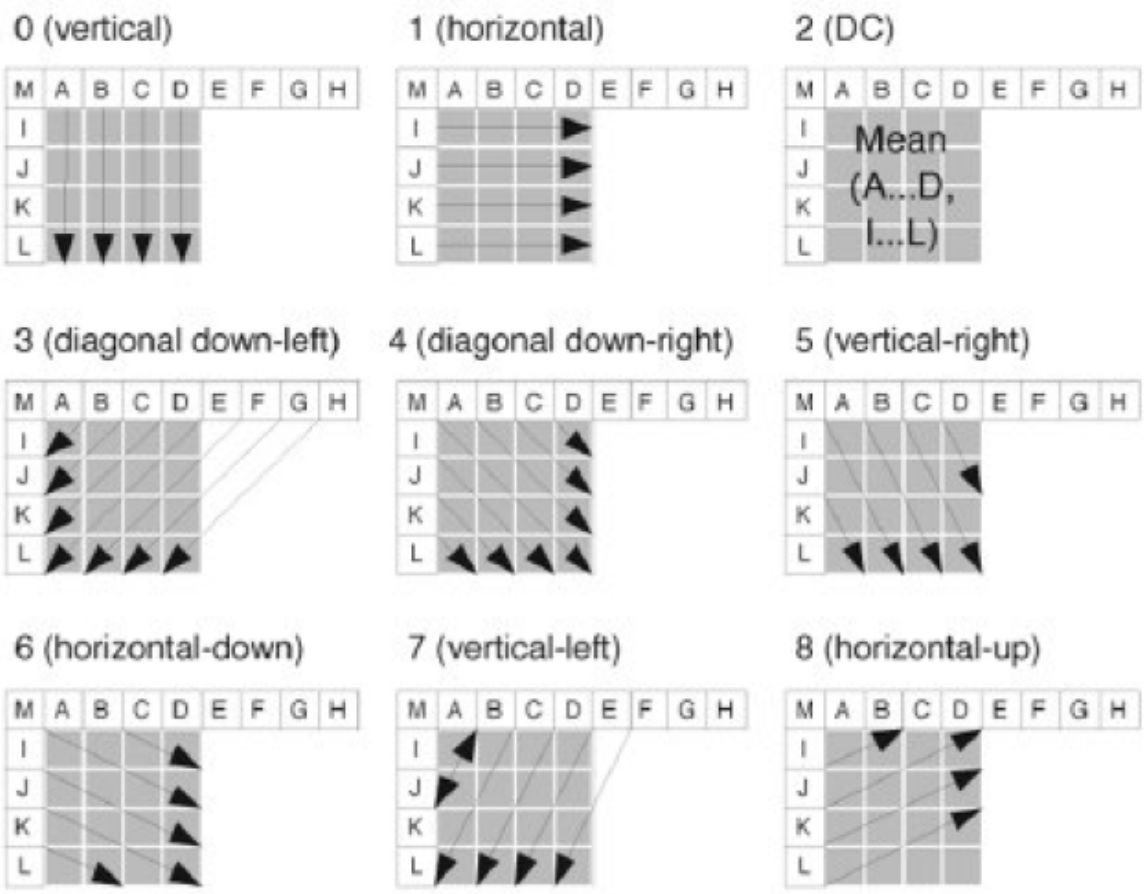


Fig. 3.13: Mode decisions for intra prediction [13]

Inter prediction: The inter prediction block includes both motion estimation (ME) and motion compensation (MC). It generates a predicted version of a rectangular array of pixels, by choosing similarly sized rectangular arrays of pixels from previously decoded reference pictures and translating the reference arrays to the positions of the current rectangular array. Fig. 3.14 depicts inter-prediction.

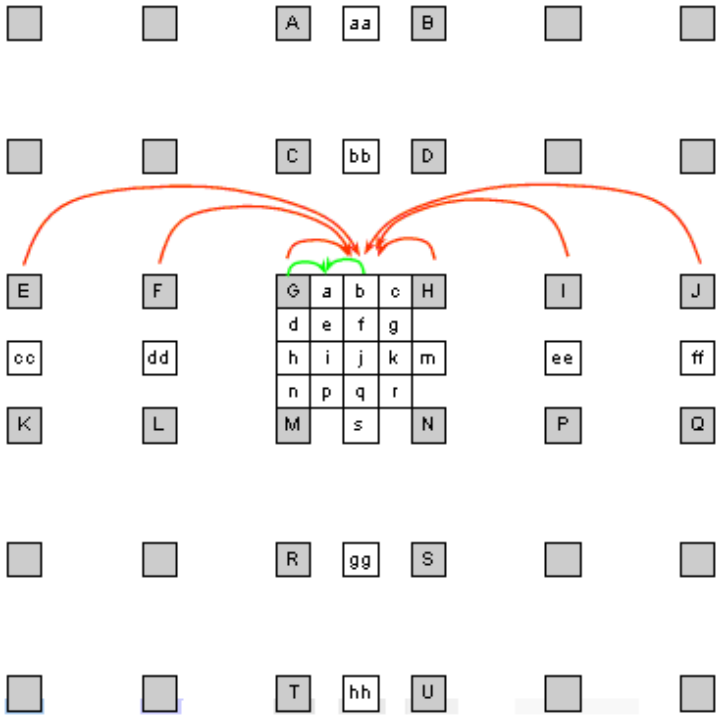


Fig. 3.14: Inter prediction in H.264 [12]

In Fig. 3.14, half-pel is interpolated from neighboring integer-pel samples using a 6-tap Finite Impulse Response filter with weights (1, -5, 20, 20, -5, 1) / 32, quarter-pel is produced using bilinear interpolation between neighboring half- or integer-pel samples. [12]

In the AVC, the rectangular arrays of pixels that are predicted using MC can have the following sizes: 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16pixels. The translation from other positions of the array in the reference picture is specified with quarter pixel precision. In case of 4:2:0 format, the chroma MVs have a resolution of 1/8 of a pixel. They are derived from transmitted luma MVs of 1/4 pixel resolution, and simpler filters are used for chroma as compared to luma. Fig. 3.15 illustrates the partitioning of the macro-block for motion compensation.

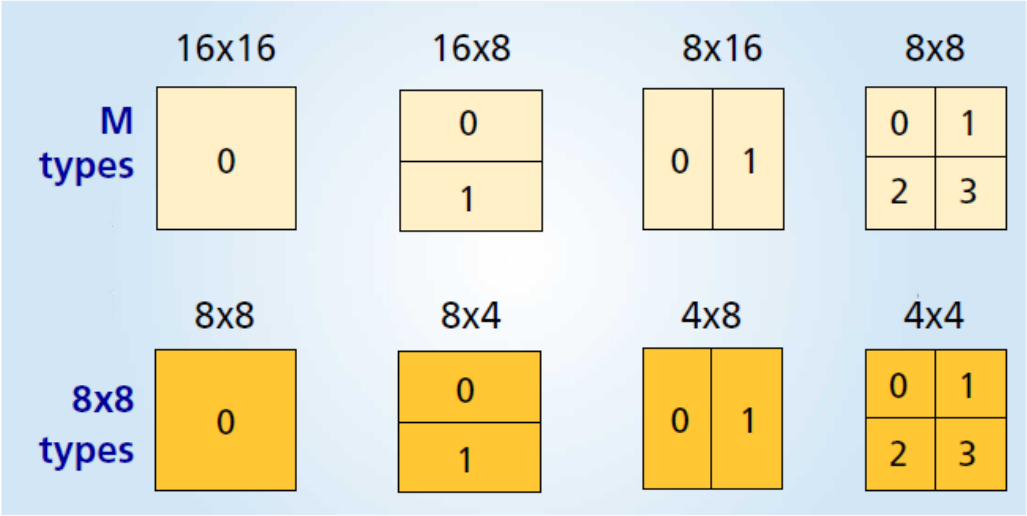


Fig. 3.15: Segmentations of the macro-block for motion compensation [51]

Fig. 3.16 depicts sub-pel motion compensation block of the H.264/AVC encoder.

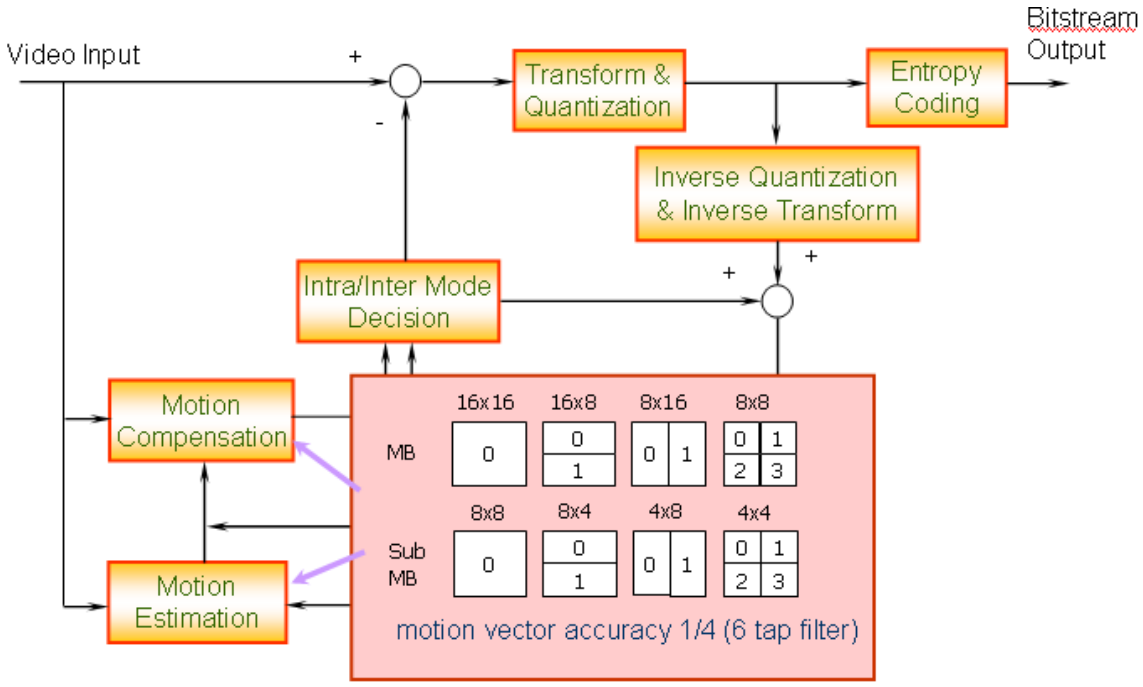


Fig. 3.16: Block diagram emphasizing sub-pel motion compensation [12]

H.264/AVC supports multi-picture motion-compensated prediction. That is, more than one prior-coded picture can be used as a reference for motion-compensated prediction as shown in Fig. 3.17. In addition to the motion vector, the picture reference parameters (Δ) are also transmitted. Both the encoder and decoder have to store the reference pictures used for Inter-picture prediction in a multi-picture buffer. The decoder replicates the multi-picture buffer of the encoder, according to the reference picture buffering type and any memory management control operations that are specified in the bit stream. [51]

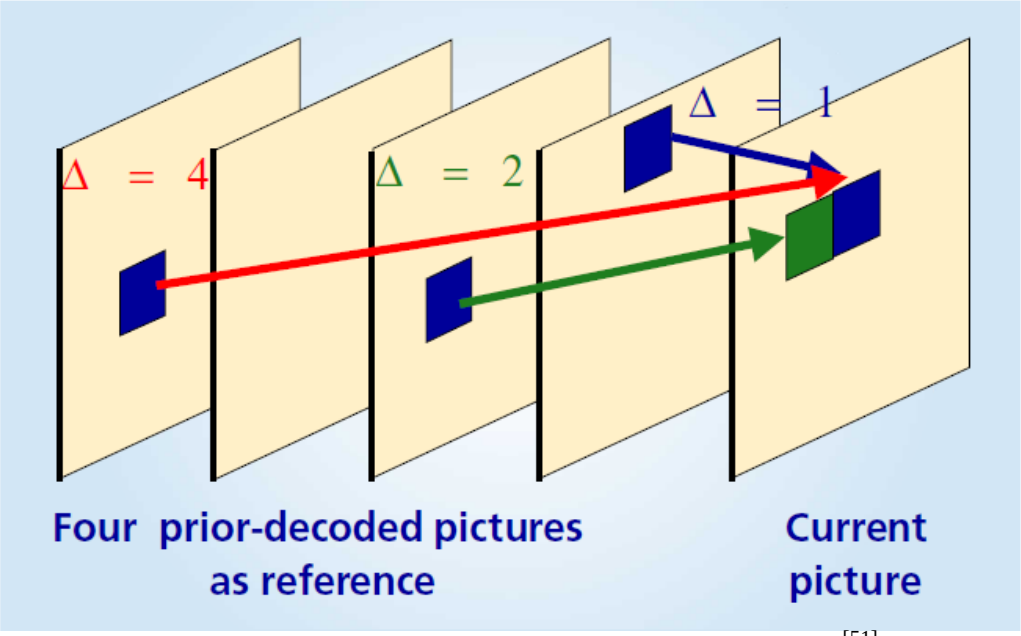


Fig. 3.17: Multi-frame motion compensation in H.264 [51]

The H.264 / MPEG-4 AVC decoder takes in the encoded bit stream as input and gives raw YUV video frames as output. The header or syntax information and slice data with motion vectors is extracted by the entropy decoder block through which the bit stream is passed. Next the residual block data is extracted by means of inverse scan and inverse quantizer. Inverse transform is carried out on all the blocks in order to map them from transform domain to pixel domain. Predicted block is formed using motion vectors and previously decoded reference frames if the block is found to be inter coded. Then the predicted block and residual block are combined to reconstruct the complete frame. This decoded frame is then presented to the user after it is passed through a de-blocking filter.

The decoding procedure is illustrated in the block diagram in Fig. 3.18.

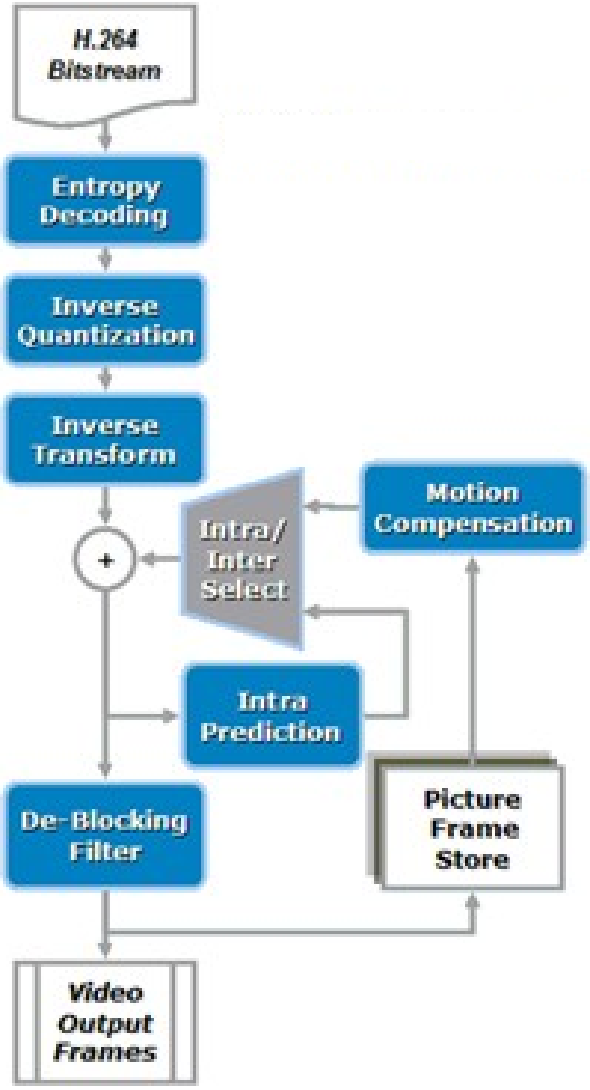


Fig. 3.18: Basic block diagram of H.264 decoder [47]

The decoder performs inverse quantization and pre-scaling as represented in the following equation: [12]

$$A'_{ij} = B_{ij} \bullet Qstep \bullet SF_{ij} \tag{3.6}$$

where *B*: inverse quantizer input, *A'*: inverse quantizer output, *Qstep :* quantization parameter, *SF* : scaling term

Fig. 3.19 shows the transform, scaling and quantization blocks at the decoder part of H.264 / MPEG-4 Part 10.

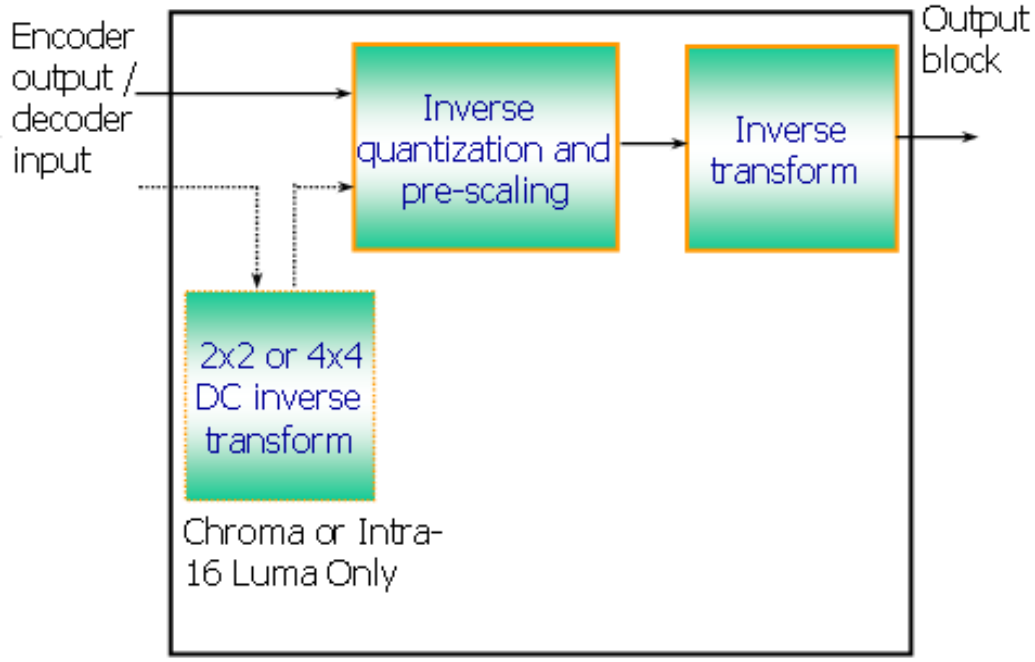


Fig. 3.19: Transform, scaling and quantization at H.264 decoder [12]

3.4 Features of H.264 [45] [50]

H.264/AVC/MPEG-4 Part 10 [13 - 16] contains a number of new features that allow it to compress video much more effectively than previous standards and to provide more flexibility for applications in a wide variety of network environments. Some of the features are listed below:

- Multi-picture and inter-picture prediction including - variable block size motion compensation, weighted prediction, quarter-pixel accuracy for motion compensation, unrestricted / multiple motion vectors per macro-block, six-tap filtering for half-pel luma sample prediction, support for multiple reference pictures
- Intra prediction in spatial domain
- 4*4 transform for 16 bit word length
- Arithmetic coding - used to create a very powerful technique CABAC
- Parameter set structure
- Flexible slice size - increases coding efficiency by reducing the header data
- PAFF (Picture adaptive frame/field)
- MBAFF (Macro block adaptive frame/field)
- Lossless macro-block coding and flexible macro-block ordering
- Arbitrary slice ordering
- Flexible interlaced-scan video coding
- New transform design features including - exact match integer 4×4 and 8×8 spatial block transforms, secondary Hadamard transform for high compression in smooth regions.

- A quantization design including - logarithmic step size control for easier bit rate management and in-loop de-blocking filter to avoid blocking artifacts.
- Entropy coding design including - Context-adaptive binary arithmetic coding (CABAC), Context-adaptive variable-length coding (CAVLC) and Variable length coding (VLC) techniques.
- Loss resilience features including - Network Abstraction Layer (NAL), Flexible macro-block ordering (FMO), Data partitioning (DP), Redundant slices (RS), Frame numbering.
- Support of monochrome, 4:2:0, 4:2:2, and 4:4:4 chroma sub-sampling
- Support of sample bit depth precision ranging from 8 to 12 bits per sample
- Picture order count

3.5 Applications of H.264 / MPEG-4 Part 10 [12]

The H.264 / MPEG-4 part 10 AVC has a broad range of applications for video content. Some of them are listed below:

- Video streaming over the internet
- Cable TV on optical networks
- Direct broadcast satellite video services
- Digital subscriber line (DSL) video services
- Digital terrestrial television broadcasting, cable modem (DSL)
- Interactive storage media
- Multimedia mailing

- Multimedia services over packet networks
- Real-time conversational services (videoconferencing, videophone, etc.)
- Remote video surveillance
- Serial storage media (digital VTR, etc.)
- D Cinema content distribution, studio editing, post processing
- Scalable video coding (SVC) and multi-view video coding (MVC) have been adopted as extensions to the H.264.

3.6 Summary

In this chapter, an overview of the H.264 / MPEG-4 Part-10 AVC is presented. The various profiles of the encoder and the encoding and decoding procedures are discussed in detail. The various features and applications of H.264 are also discussed.

CHAPTER 4

ENCODING AND DECODING IN DIRAC

4.1 Need for encoding

Video encoding is the process of preparing the video for output, where the digital video is encoded to meet proper formats and specifications for recording and playback through the use of video encoder software.[21] Streaming video quality is dependent in part upon the video encoding process and the amount of bandwidth required for it to be viewed properly. It is usually standard practice to produce or encode streaming video for delivery over the Internet at a minimum of standard 56K modem speed since many users are still connected at these speeds. While encoding a video, a high degree of compression is applied to both the video and audio tracks so that it will stream at this speed. [22]

4.2 Wavelet transform

The 2D discrete wavelet transform provides Dirac with the flexibility to operate at a range of resolutions. This is because wavelets operate on the entire picture at once, rather than focusing on small areas at a time. In Dirac, the discrete wavelet transform plays the same role of the DCT in MPEG-2 in de-correlating data in a roughly frequency-sensitive way, whilst having the advantage of preserving fine details better. In one dimension, it consists of the iterated application of a complementary pair of half-band filters followed by sub-sampling by a factor 2 as shown in Fig. 4.1. [41]

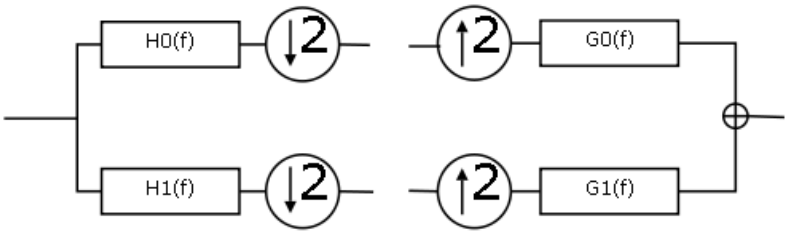


Fig. 4.1: Perfect reconstruction analysis and synthesis filter pairs [41]

The synthesis filters can undo the aliasing introduced by critical sampling and perfectly reconstruct the input. The filters split the signal into a LH (low-high), high-frequency (HF) part and the wavelet transform then iteratively decomposes the low-frequency (LF) component to produce an octave-band decomposition of the signal. [41]

The wavelet transform is constructed by repeated filtering of signals into low- and high-frequency parts. For two-dimensional signals, this filtering occurs both horizontally and vertically. At each stage, the low horizontal / low vertical frequency sub-band is split further, resulting in logarithmic frequency decomposition into sub-bands. Wavelet transforms have been proven to provide a more efficient technique than block transforms with still images. Within the Dirac wavelet filters, the data is encoded in 3 stages as shown in Fig. 4.2.

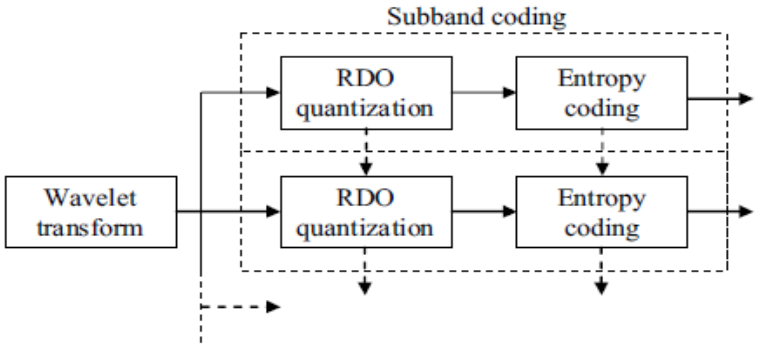


Fig. 4.2: Dirac's wavelet transform architecture [5]

Daubechies wavelet filters [39] [40] are used to transform and divide the data in sub-bands which then are quantized with the corresponding RDO (rate distortion optimization) parameters and then variable length encoded. These three stages are then reversed at the decoder. [5]

The choice of wavelet filters has an impact on compression performance. Filters are required to have compact impulse response in order to reduce ringing artefacts and other effects so as to represent smooth areas compactly. It also has an impact on encoding and decoding speed in software. There are numerous filters supported by Dirac to allow a tradeoff between complexity and performance. These are configurable in the reference software. [41]

One filter available in Dirac is an approximation of the Daubechies (9, 7) low pass wavelet filter: [41]

$$
\begin{aligned}
s_n^0 &= x_{2n} \\
d_n^0 &= x_{2n+1} \\
d_n^1 &= d_n^0 - (6497 \cdot (s_n^0 + s_{n+1}^0)) / 4096 \\
s_n^1 &= s_n^0 - (217 \cdot (d_n^1 + d_{n-1}^1)) / 4096 \\
d_n^2 &= d_n^1 + (3616 \cdot (s_n^1 + s_{n+1}^1)) / 4096 \\
s_n^2 &= s_n^1 + (1817 \cdot (d_n^2 + d_{n-1}^2)) / 4096
\end{aligned}
\tag{4.1}
$$

where s denotes sum and d denotes difference.

The numbers are integer approximations of the Daubechies lifting coefficients. This makes the transform fully invertible. The implementation ignores scaling coefficients, since these can be taken into account in quantizer selection by weighting the quantizer noise appropriately. The problem with this filter is that it has four lifting stages, and so it takes longer time in software. [41]

At the other extreme is the (5, 3) Daubechies high pass filter: [41]

$$d_n^1 = d_n^0 - (s_n^0 + s_{n+1}^0)/2$$
$$s_n^1 = s_n^0 + (d_n^1 + d_{n-1}^1)/4 \quad (4.2)$$

The discrete wavelet transform packs most of the information into only a few sub-bands (at low frequency) as shown in Fig. 4.3, which allows compression to be achieved.

Fig. 4.3: Stages of wavelet transform [1]

This process can be repeated to achieve higher levels of wavelet transform. In case of two-dimensional images, wavelet filters are normally applied in both vertical and horizontal directions to each image component to produce four so-called sub-bands termed Low-Low (LL), Low-High (LH), High-Low (HL) and High-High (HH). In the case of two dimensions, only the LL band is iteratively decomposed to obtain the decomposition of the two-dimensional spectrum as shown in Fig. 4.4. [41]

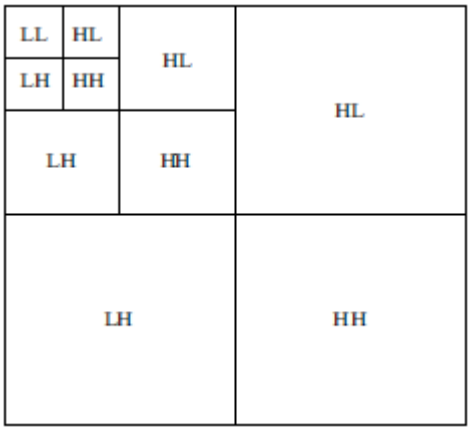


Fig. 4.4: Wavelet transform frequency decomposition [5]

A Dirac-coded picture is free from block artifacts and is clearly superior in the case of moving images. [1]

4.3 Scaling and Quantization

Scaling involves taking frame data after application of wavelet transform and scaling the coefficients to perform quantization. Quantization employs a rate distortion optimization algorithm to strip information from the frame data that results in as little visual distortion as possible. Dirac uses a dead-zone quantization as shown in Fig. 4.5 which differs from orthodox quantization by making the first set of quantization step twice as wide. This allows Dirac to perform coarser quantization on smaller values. [5]

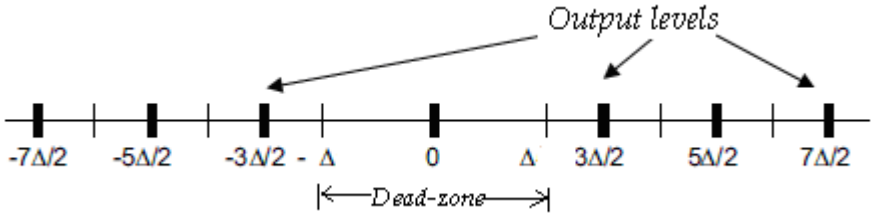


Fig. 4.5: Dead-zone quantizer with quality factor (QF) [5]

4.4 Entropy Coding

Entropy coding is applied after wavelet transform to minimize the number of bits used. It consists of three stages: binarization, context modeling and arithmetic coding [5] as shown in Fig. 4.6. The purpose of the first stage is to provide a bit stream with easily analyzable statistics that can be encoded using arithmetic coding, which can adapt to those statistics, reflecting any local statistical features. The context modeling in Dirac is based on the principle that whether a coefficient is small or not is well-predicted by its neighbors and its parents. [3] Arithmetic coding performs lossless compression and is both flexible and efficient.

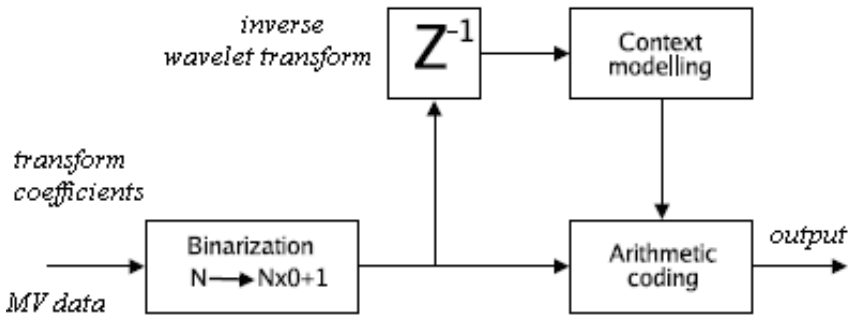


Fig. 4.6: Dirac's entropy coding architecture [6]

The non-zero values in the higher frequency sub-bands of the wavelet transform are often in the same part of the picture as they are in lower frequency sub-bands. Dirac creates statistical models of these correlations and arithmetic coding allows us to exploit these correlations to achieve better compression. The motion information estimated at the encoder also uses statistical modeling and arithmetic coding to compress it into the fewest number of bits. This compressed data is put into the bit stream, to be used by the decoder as part of the compressed video.

4.5 Motion Estimation

Motion estimation exploits temporal redundancy in video streams by looking for similarities between adjacent frames. Dirac implements hierarchical motion estimation (Fig. 4.7) in three distinct stages.

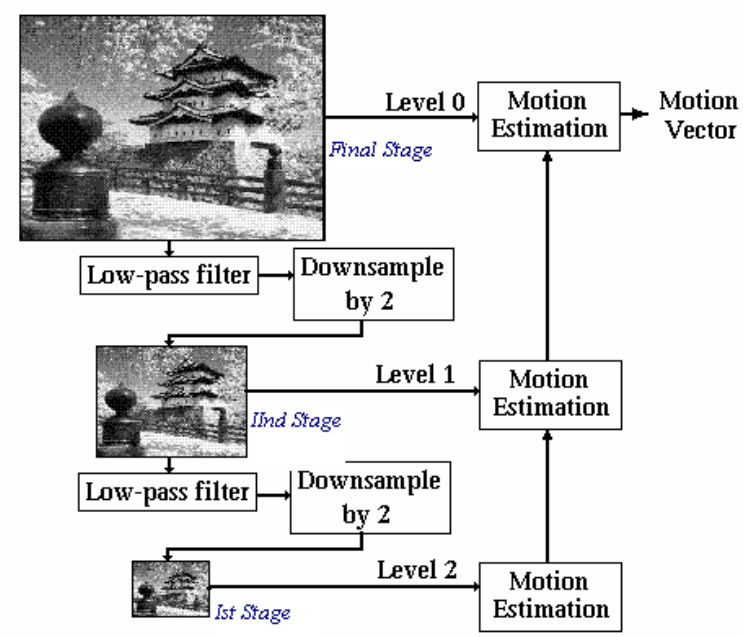


Fig. 4.7: Hierarchical motion estimation [10]

In the first stage, pixel accurate motion vectors are determined for each block and each reference frame by hierarchical block matching. In the second stage, these pixel-accurate vectors are refined by searching sub-pixel values in the immediate neighborhood. In the final stage, mode decisions are made for each macro-block, determining the macro-block splitting level and the prediction mode used for each prediction unit. This last stage involves further block matching as block motion vectors are used as candidates for higher-level prediction units. [8]

In its hierarchical motion estimation, Dirac first down converts the size of the current and reference of all types of inter frames (both P and B) using the 12 taps down conversion filter. [9] Down conversion filters are lowpass filters that pass only the desired signal and also perform anti-alias filtering prior to decimation. Any suitable lowpass filter can be used including FIR, IIR and CIC filters. [43] The number of down conversion levels depends upon the frame format. [9]

The number of down conversion levels is 4 and 6 for the frame format CIF and HD (1920x1080) respectively. In the down conversion process, the dimension (both height and width) of the frames are reduced by the factor of two in each level. The motion estimation is performed first in the lowest resolution (smallest frame) level and gradually increased to the higher resolution levels until it reaches the original frame size. The search pattern used in lowest level is diamond shape [31] with the search range 5 and all other levels except the lowest level use square shape search pattern with search range 1. Fig. 4.8 shows both search patterns where there are altogether 61 search points in diamond shape and 9 points in square shape. [9]

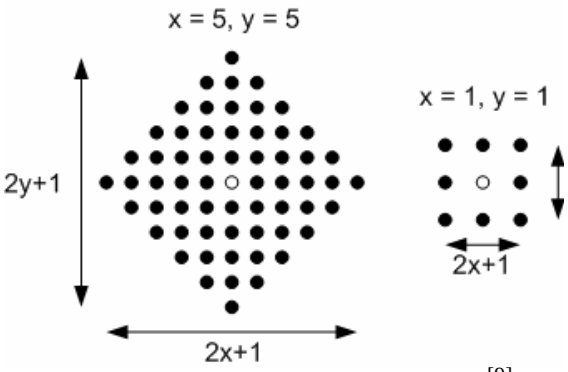


Fig. 4.8: Search patterns in Dirac [9]

Initially a list of points to be searched (candidate list) is generated. These points follow either a diamond or square pattern and are centered at coordinates pointed by the motion vector (MV). For the lowest search level, two candidate lists are generated that are centered at zero motion vector and predicted motion vector respectively with a diamond search pattern. Predicted motion vector is a spatially predicted MV, which is the median vector of left, top left and top blocks of the current block where motion estimation is carried out. [9] Here sum of the absolute difference (SAD) is used as the cost function. Given two blocks X, Y of samples, SAD is given by: [42]

$$\mathrm{SAD}(X, Y) = \sum_{i,j} |X_{i,j} - Y_{i,j}| \tag{4.3}$$

Initially SAD calculation is carried out only for the center point of diamond pattern in each list and this gives the list with minimum cost. The candidate lists to be searched are chosen by multiplying the minimum cost with 1.5. Next, the lists which are less than 1.5 times the minimum costs are chosen. Hence there can be at most two candidate lists and 122 search points involved in the lowest level search if there is no overlapping between the two lists. After that, SAD calculation is carried out for all chosen search points and the coordinate of the point which gives minimum cost is recorded as the best MV. [9]

The smoothness measure used is based on the difference between the candidate motion vector and the median of the neighboring previously computed motion vectors. Since the blocks are estimated in raster-scan (Fig. 4.9) order then vectors for blocks to the left and above are available for calculating the median vector. [42]

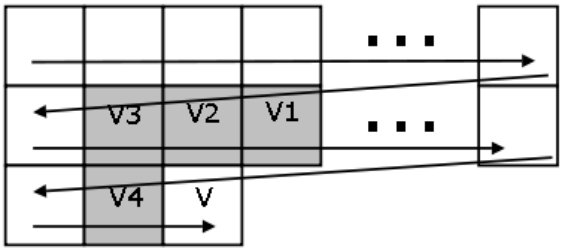


Fig. 4.9: Neighboring motion vectors in raster-scan order for local variance calculation [42]

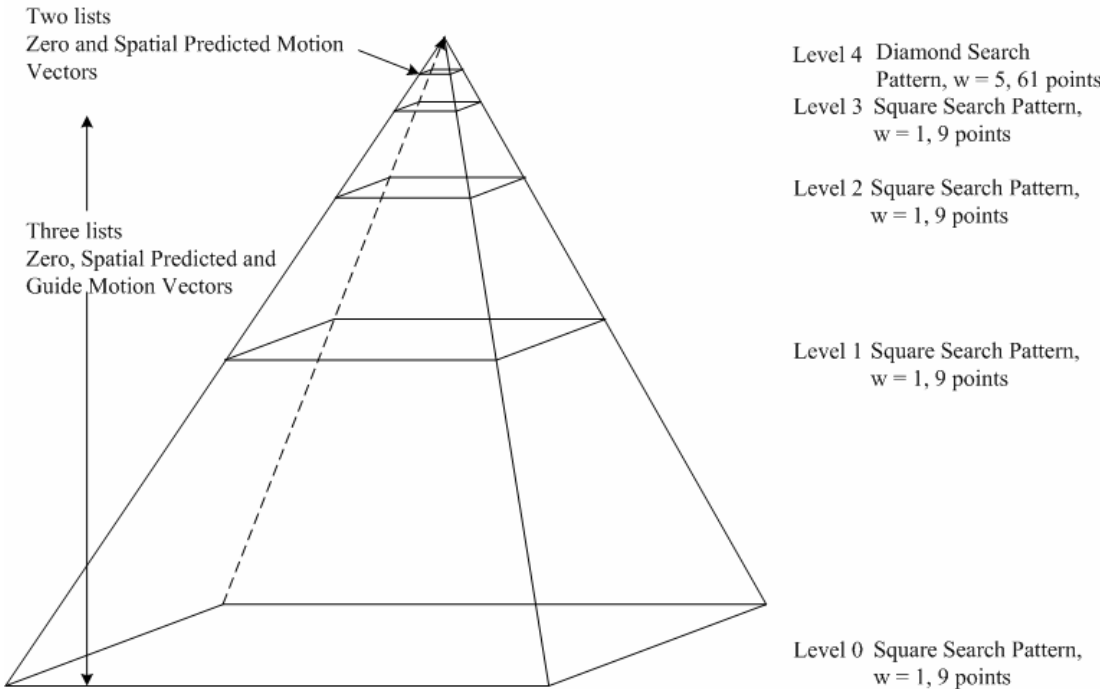


Fig.4.10: Dirac's 4 level hierarchical motion estimation for CIF format, w is search range [9]

Dirac also defines three types of frames. Intra (I) frames are coded without reference to other frames in the sequence. Level 1 (L1) frames and Level 2 (L2) frames are both inter frames, that is, they are coded with reference to other previously coded frames.

The difference between L1 and L2 frames is that L1 frames are also used as temporal references for other frames, whereas L2 frames are not. [3] A prediction structure for frame coding using a standard GOP structure [7] is shown in Fig. 4.11. Each frame in Dirac may be predicted from up to two reference frames. Prediction modes can be varied by prediction unit, and there are four possibilities: Intra, Reference 1 only, Reference 2 only and Reference 1 and 2 (bi-directional prediction). [8]

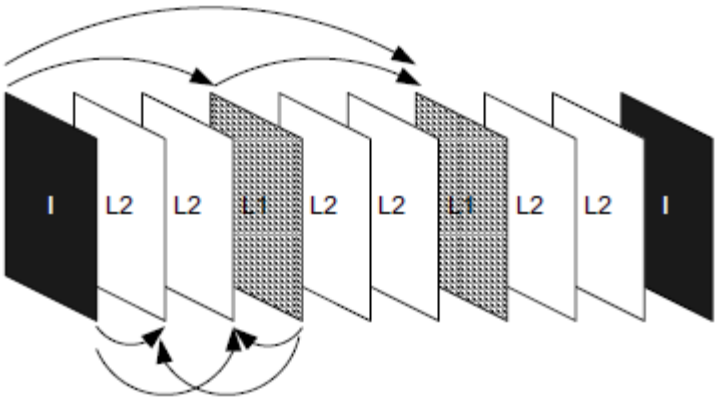


Fig. 4.11: Prediction of L1 and L2 frame in Dirac [7]

4.6 Motion Compensation

Motion compensation is used to predict the present frame. Dirac uses overlapped block-based motion compensation (OBMC) to achieve good compression and avoid block-edge artifacts which would be expensive to code using wavelets. OBMC allows interaction of neighboring blocks. OBMC is performed with basic blocks arranged into macro-blocks consisting of a 4x4 array of blocks. [8]

There should be an exact number of macro-blocks horizontally and vertically. This is achieved by padding the data. Further padding may also be needed because after motion compensation the wavelet transform is applied, which has its own requirements for divisibility. [11]

Although Dirac is not specifically designed to be scalable, the size of blocks is the only non-scalable feature, and for lower resolution frames, smaller blocks can easily be selected. Dirac's OBMC scheme is based on a separable linear ramp mask. This acts as a weight function on the predicting block. Given a pixel p=p(x,y,t) in frame t, p may fall within only one block or in up to four blocks if it lies at the corner of a block as shown in Fig. 4.12 where the darker-shade areas show overlapping areas. [11]

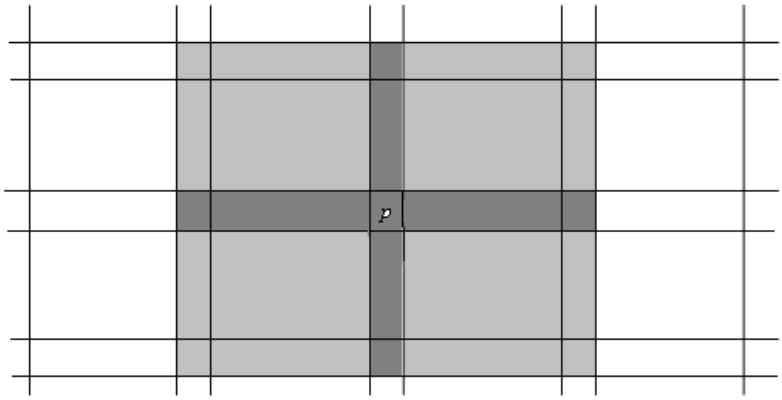


Fig. 4.12: Overlapping blocks in OBMC [11]

Each macro-block may be split in one of three ways into prediction units consisting either of 16 individual blocks, or of an array of 4 mid-size blocks, termed sub-macro-bocks, or of a single macro-block-sized block (Fig. 4.13).

OBMC parameters may be changed frame-by-frame, but defaults exist based on frame sizes. The default for both streaming and standard definition resolution is for 12x12 blocks which are overlapped at intervals of 8 pixels vertically and horizontally (the dimensions are scaled appropriately for chroma components of different resolutions). The OBMC overlapping function used is an integer approximation to the raised-cosine function. [8]

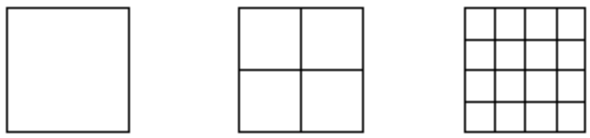


Fig. 4.13: Modes of splitting macro-block into sub-blocks in Dirac [8]

Dirac also provides sub-pixel motion compensation with motion vectors and thereby improves the prediction rate up to 1/8th pixel accuracy. It supports the use of global motion estimates in a few bytes. Techniques such as predicting a frame using only motion information (without transmitting any wavelet coefficients) and predicting a frame to be identical to a previous frame at low bit rates are also supported. It involves using the motion vectors to predict the current frame in such a way as to minimize the cost of encoding residual data. The FrameBuffer class employed within Dirac manages all temporal predictions; this is done by encapsulating an encoded frame within a header. This header is used to define the display order of the frame, the reference frame dependencies and the length of stay within the FrameBuffer. [5]

4.7 Decoder

The Dirac's decoder implementation is designed to provide fast decoding whilst remaining portable across various software platforms. The decoding process is carried out in three stages as shown in Fig. 4.14. At the first stage, the input encoded bit-stream is decoded by variable length decoding. Next, scaling and inverse quantization is performed. In the final stage, inverse transform is applied on the data.

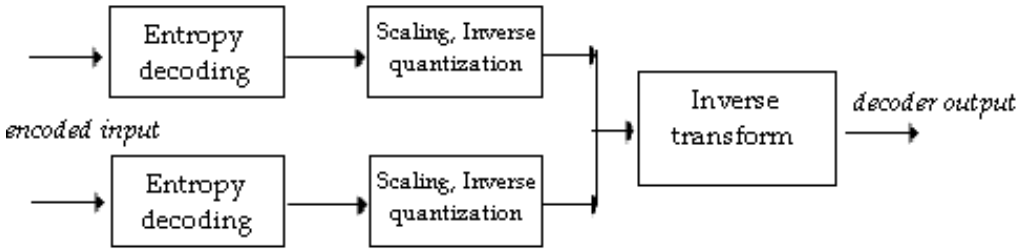


Fig. 4.14: Stages of decoding in Dirac

Based on the information provided by each frame header of the FrameBuffer class, the decoder searches for the required reference frame(s) used for prediction of the current frame, and then produces the output with regard to display order. A trade off is made between accuracy and motion vector bit rate. Such techniques can provide substantial bit rate reductions when only a modest quality is required. [5]

4.8 Summary

In this chapter, each block of the encoder and the decoder of the Dirac video codec are discussed in detail.

CHAPTER 5

RESULTS AND CONCLUSION

5.1 Implementation and results

Dirac can be fully implemented in C++ programming language which allows object oriented development on all common operating systems. The C++ code compiles to produce libraries for common functions, motion estimation, encoding and decoding, which have an interface that allows them to be called from C. An application programmer's interface can be written in C so that it can be kept simple and integrated with various media players, video processing tools and streaming software. [2]

Objective test methods attempt to quantify the error between a reference and encoded bitstream. [5] In the first set of tests, Dirac's QF parameter (which determines the quality scale of the video encoded) has been used. In these tests QF has been varied from 0 – 10 and the corresponding bitrate, compression, SSIM, PSNR and MSE have been recorded.

To ensure the accuracy of the tests, each codec must be encoded using the same bitrate. [5] Since the latest version of Dirac does include a CBR mode, the comparison between Dirac and H.264's performance was produced by encoding several test sequences at different bitrates. Utilizing the CBR mode within H.264, we can ensure that H.264 is being encoded at the same bitrate of that of Dirac.

Objective tests are separated into five sections, namely (i) Compression, (ii) Bitrate, (iii) SSIM, (iv) PSNR and (v) MSE.

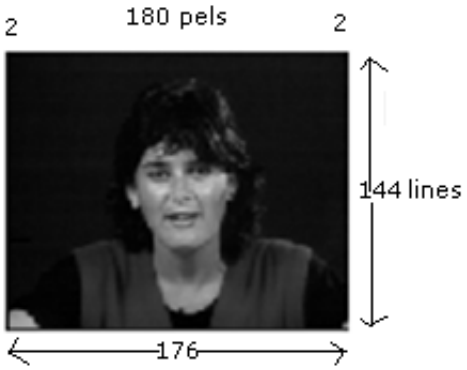
The test sequences “Miss-america” QCIF (176x144) [32], “Stefan” CIF (352x288) [32] and “Susie" standard-definition (SD) (720x480) [33] have been used for the Dirac testing process. In these tests, the video sequences have been encoded and decoded within Dirac at varying the QF.

I. Performance analysis of Dirac on “miss-america.yuv” QCIF sequence

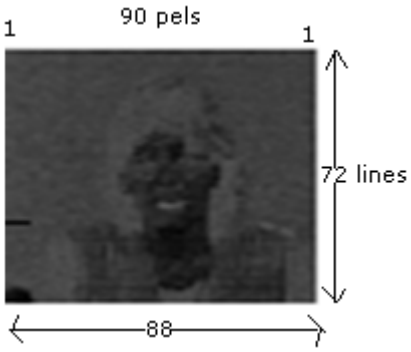
Table 5.1: Basic information for QCIF sequence “miss-america.yuv”

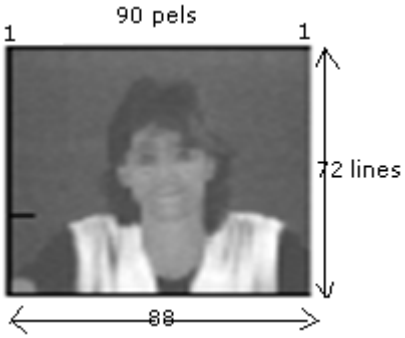
Frame rate (fps)	25
Duration(sec)	6
Width	176
Height	144
Bit rate(KB/sec)	950.4
Size (KB)	5569
Number of frames	150

Fig. 5.1: Original “miss-america.yuv" QCIF sequence



(a)





(b) (c)

Fig. 5.2: “miss-america.yuv” (a) Y component, (b) C_b component and (c) C_r component

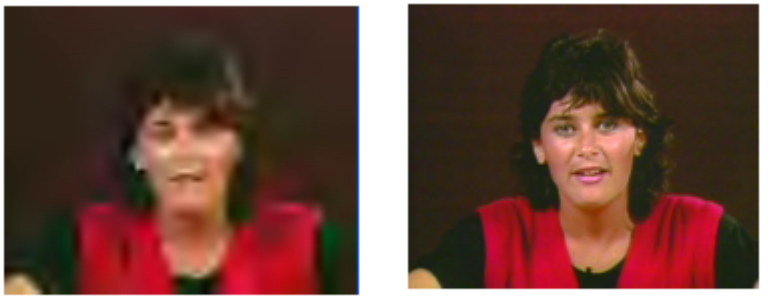


Fig. 5.3: Dirac compressed “miss-america.yuv" QCIF sequence at QF = 0 (left), QF = 10 (right)

Dirac output sequence, lossless encoding
PSNR: 100.00, Bitrate: 482.3KBps

Fig. 5.4: Dirac compressed “miss-america.yuv" QCIF sequence (lossless encoding)

Table 5.2: Performance analysis of Dirac at different QF for QCIF sequence

QF	Size(KB)*	Bitrate(KBps)	Compression ratio	PSNR-Y(dB)	PSNR-U(dB)	PSNR-V(dB)	SSIM	MSE-Y	MSE-U	MSE-V
0	30	4.963	192	30.627	34.100	31.187	0.882	56.287	25.299	49.479
1	31	5.255	181	31.883	34.620	32.406	0.902	42.144	22.444	37.362
2	33	5.586	170	32.977	35.006	33.160	0.917	32.760	20.533	31.414
3	36	6.020	158	33.664	35.289	33.640	0.928	27.968	19.241	28.122
4	41	6.895	138	35.269	35.452	34.652	0.944	19.329	18.531	22.281
5	49	8.211	116	37.035	36.348	35.959	0.957	12.870	15.077	16.489
6	61	10.381	92	39.149	37.070	37.879	0.968	7.910	12.766	10.597
7	81	13.702	69	41.341	37.975	39.759	0.976	4.775	10.366	6.873
8	115	19.601	48	44.480	39.093	41.568	0.982	3.218	8.013	4.532
9	186	31.635	30	44.224	40.235	43.152	0.985	2.458	6.160	3.147
10	308	52.531	18	45.643	41.244	44.476	0.987	1.773	4.883	2.320
Lossless	2827	482.3	2	100.000	100.000	100.000	1.000	0.000	0.000	0.000

** indicates encoded file size including all 150 frames after compression*

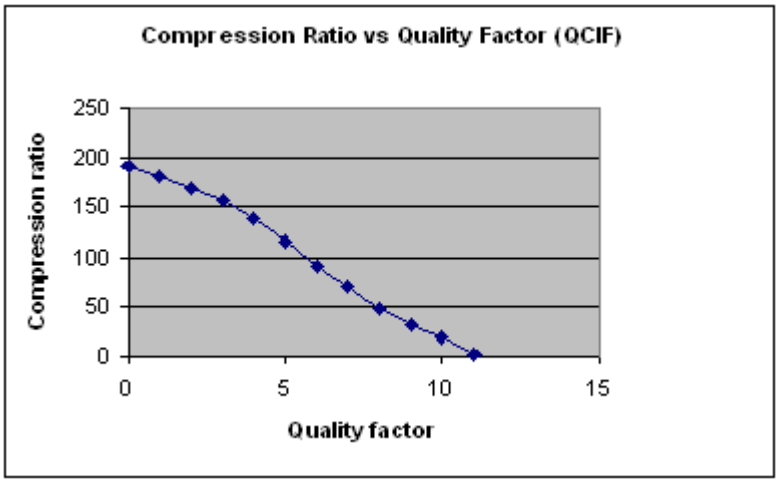


Fig. 5.5: Compression achieved by Dirac for “miss-america” QCIF sequence

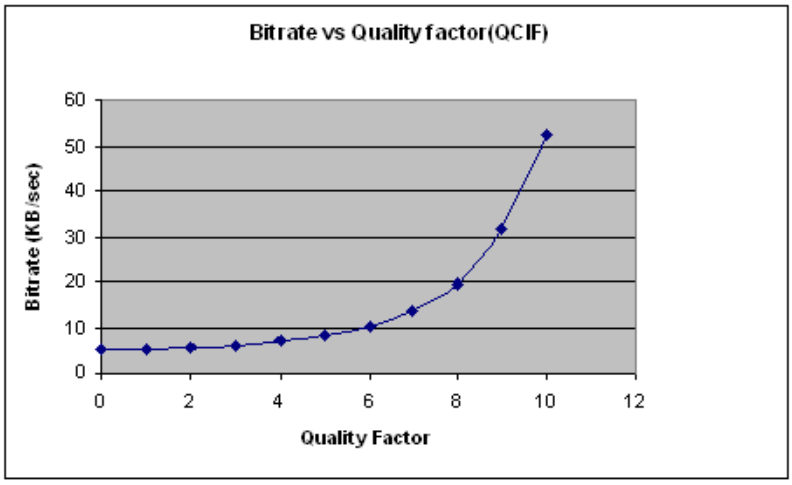


Fig. 5.6: Bitrate achieved by Dirac for “miss-america” QCIF sequence

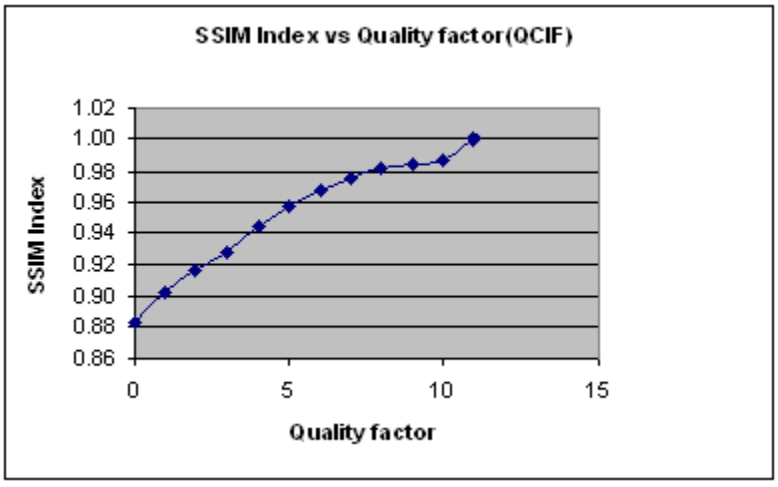


Fig. 5.7: SSIM achieved by Dirac for “miss-america” QCIF sequence

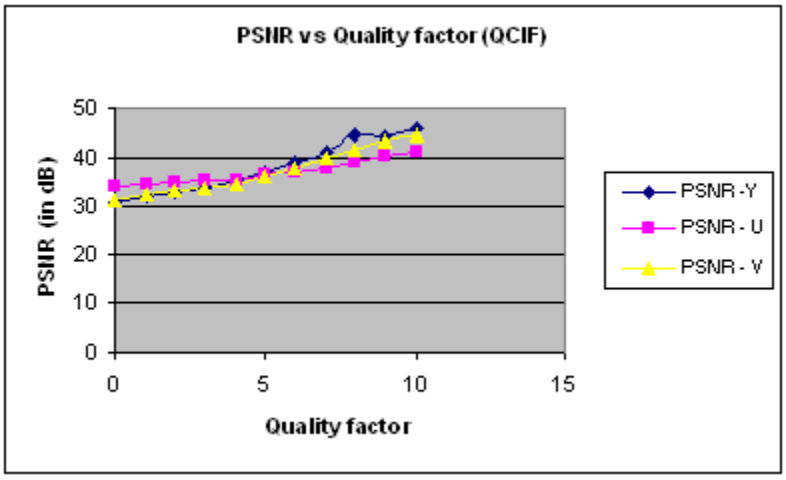


Fig. 5.8: PSNR achieved by Dirac for “miss-america” QCIF sequence

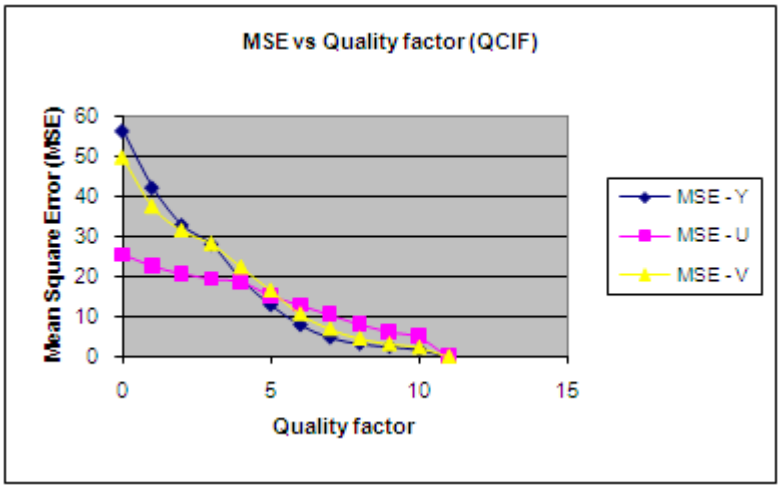
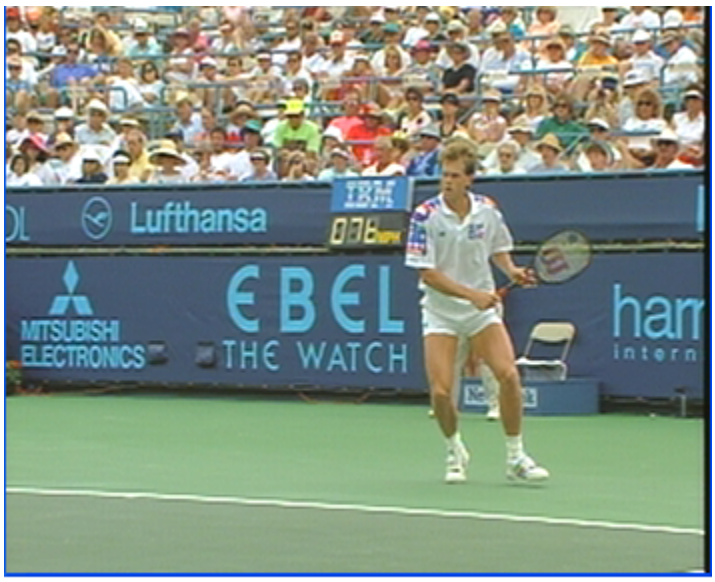


Fig. 5.9: MSE achieved by Dirac for “miss-america” QCIF sequence

II. Performance analysis of Dirac on “stefan.yuv” CIF sequence

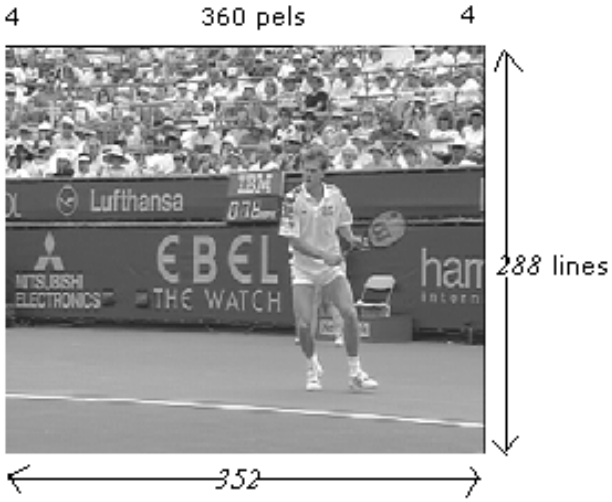
Table 5.3: Basic information for CIF sequence “stefan.yuv”

Frame rate (fps)	6.4285
Duration(sec)	14
Width	352
Height	288
Bit rate(KB/sec)	977.55
Size (KB)	13365
Number of frames	90

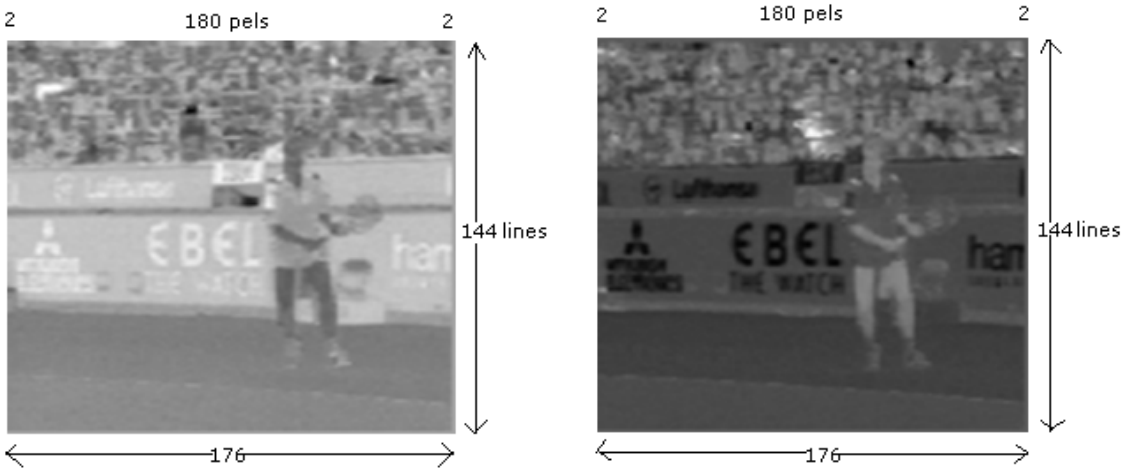


Original "stefan_cif.yuv" sequence
PSNR: 100.00, Bitrate: 977.55KBps

Fig. 5.10: Original “stefan.yuv" CIF sequence

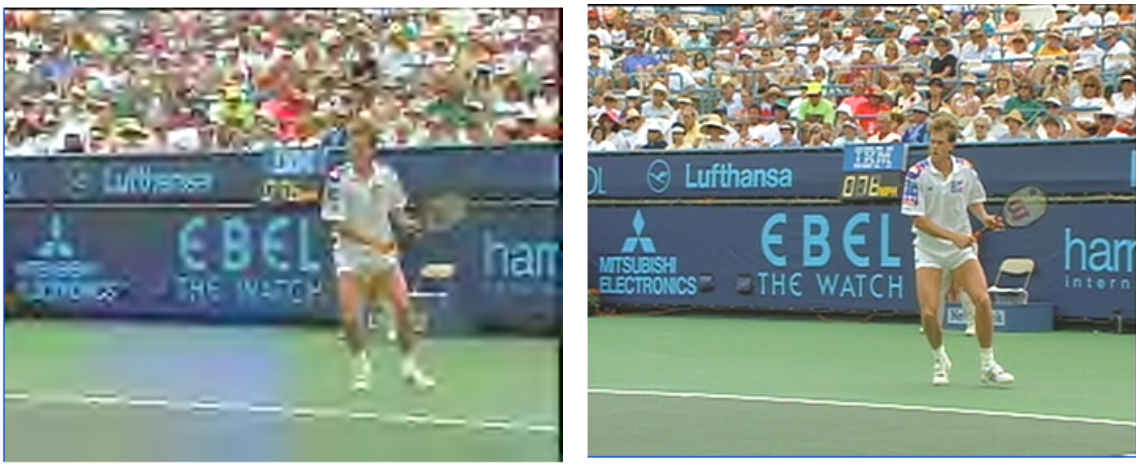


(a)



(b) (c)

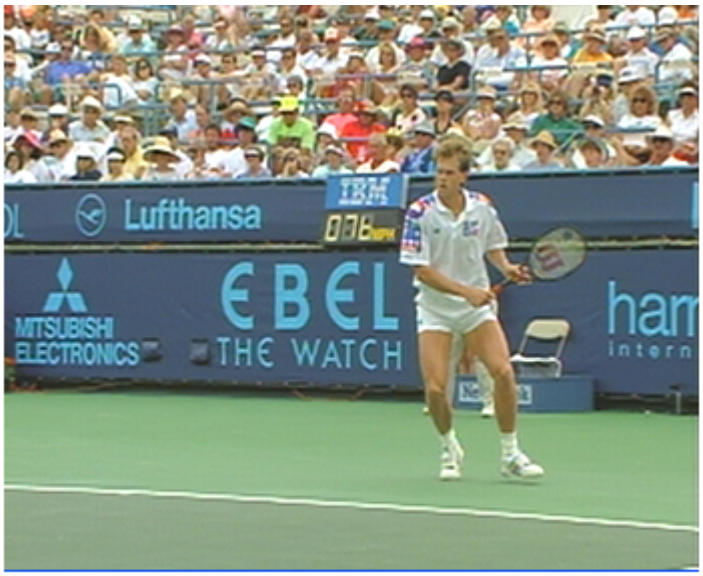
Fig. 5.11: “stefan.yuv” (a) Y component, (b) C_b component and (c) C_r component



Dirac output sequence, QF = 0
PSNR: 21.327, Bitrate: 5.808KBps

Dirac output sequence, QF = 10
PSNR: 42.667, Bitrate: 136.740KBps

Fig. 5.12: Dirac compressed “stefan.yuv" CIF sequence at QF = 0 (left), QF = 10 (right)



Dirac output sequence, lossless encoding
PSNR: 100.00, Bitrate: 821.337KBps

Fig. 5.13: Dirac compressed “stefan.yuv" CIF sequence (lossless encoding)

Table 5.4: Performance analysis of Dirac at different QF for CIF sequence

QF	Size(KB)*	Bitrate(KBps)	Compression ratio	PSNR-Y(dB)	PSNR-U(dB)	PSNR-V(dB)	SSIM	MSE-Y	MSE-U	MSE-V
0	80	1.333	140	21.327	29.174	29.845	0.666	479.000	78.646	67.392
1	96	1.603	116	23.361	30.425	30.406	0.773	299.928	58.969	59.227
2	117	1.961	95	25.281	30.757	30.945	0.840	192.739	54.629	52.305
3	151	2.521	74	27.192	31.372	31.722	0.887	124.125	47.408	43.738
4	201	3.360	55	29.413	32.163	33.430	0.926	74.436	39.515	29.515
5	277	4.640	40	31.600	33.849	35.308	0.950	44.985	26.805	19.154
6	397	6.650	28	33.629	35.721	37.337	0.966	28.198	17.419	12.007
7	583	9.786	19	35.624	37.519	38.950	0.976	17.812	11.514	8.281
8	869	14.580	13	37.787	38.846	40.157	0.983	10.824	8.481	6.272
9	1270	21.318	9	40.031	40.231	41.877	0.987	6.457	6.165	4.221
10	1870	31.383	6	42.667	42.396	44.420	0.991	3.519	3.745	2.350
Lossless	11230	821.337	0	100.000	100.000	100.000	1.000	0.000	0.000	0.000

** indicates encoded file size including all 90 frames after compression*

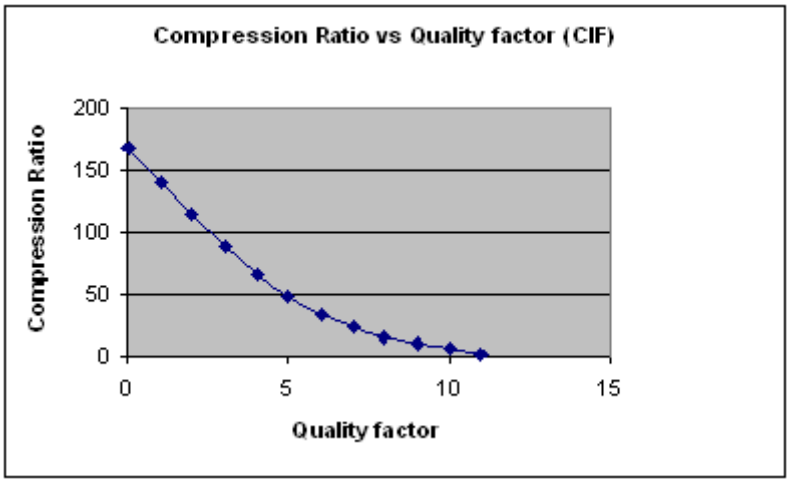


Fig. 5.14: Compression achieved by Dirac for “stefan” CIF sequence

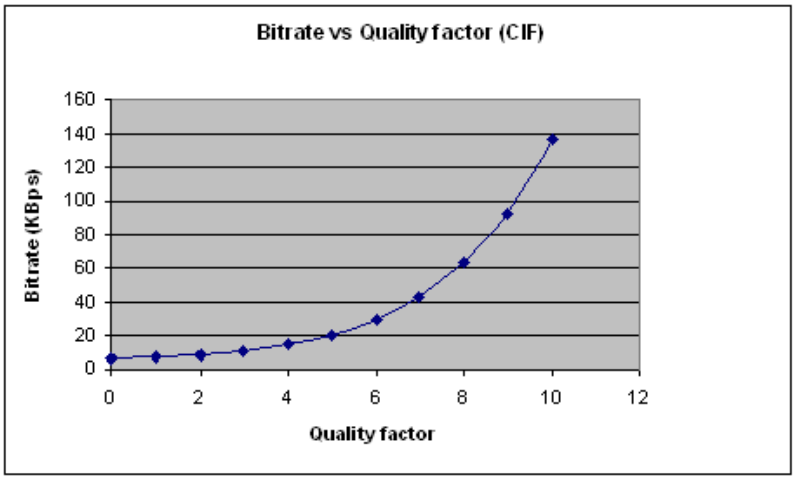


Fig. 5.15: Bitrate achieved by Dirac for “stefan” CIF sequence

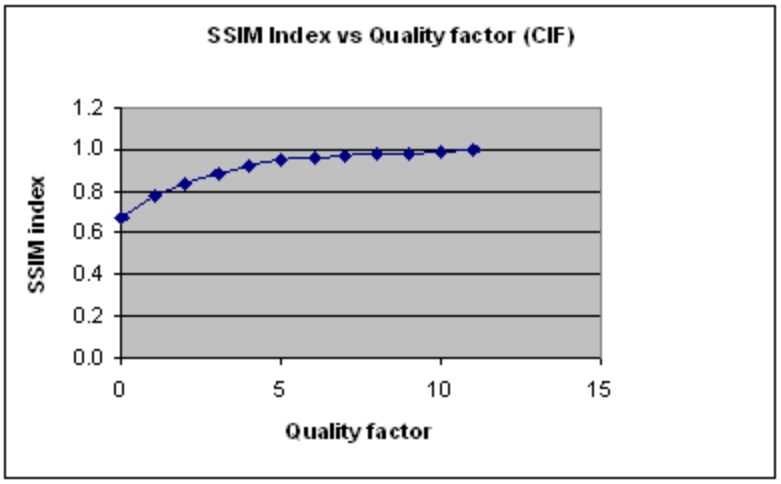


Fig. 5.16: SSIM achieved by Dirac for “stefan” CIF sequence

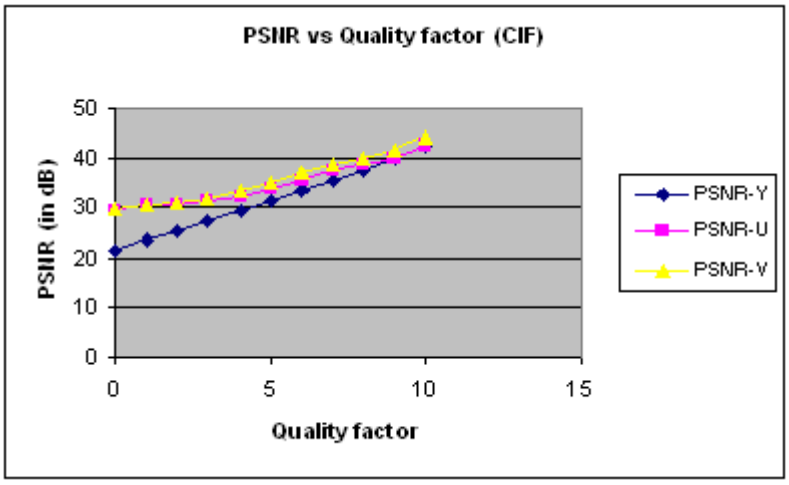


Fig. 5.17: PSNR achieved by Dirac for “stefan” CIF sequence

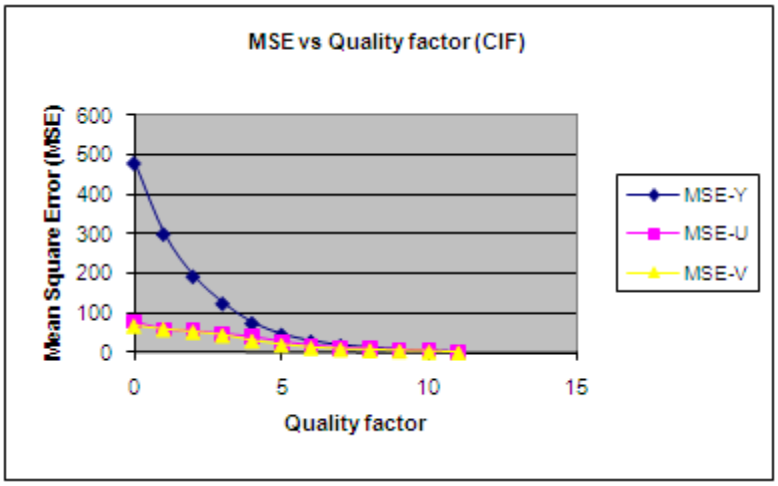


Fig. 5.18: MSE achieved by Dirac for “stefan” CIF sequence

III. Performance analysis of Dirac on “susie.yuv” SDTV sequence

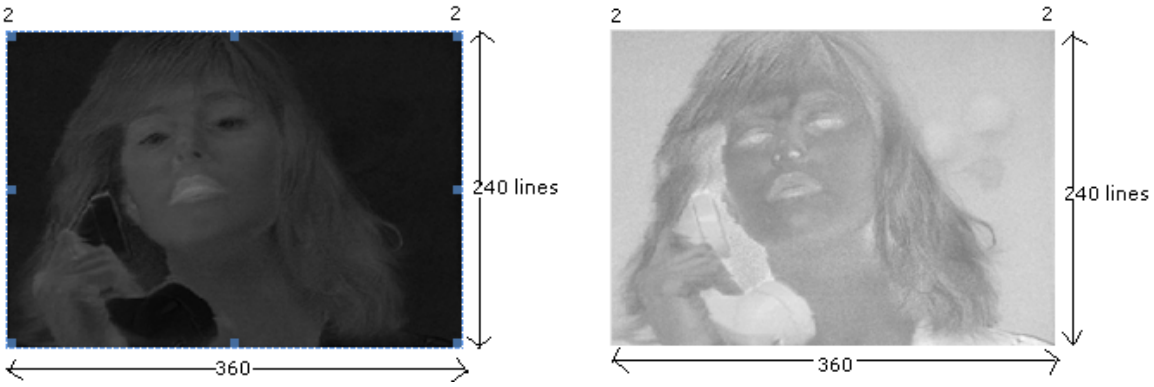
Table 5.5: Basic information for SDTV sequence “susie.yuv”

Frame rate (fps)	1.389
Duration(sec)	18
Width	720
Height	480
Bit rate(KB/sec)	960.077
Size (KB)	16875
Number of frames	25

Original “susie.yuv” SDTV sequence
PSNR: 100.00, Bitrate: 960.077KBps

Fig. 5.19: Original “susie.yuv" SDTV sequence

(a)



(b) (c)

Fig. 5.20: “susie.yuv” (a) Y component, (b) C_b component and (c) C_r component

Dirac output sequence, QF = 0
PSNR: 33.678, Bitrate: 3.613KBps

Dirac output sequence, QF = 10
PSNR: 44.750, Bitrate: 59.465KBps

Fig. 5.21: Dirac compressed “susie.yuv" SDTV sequence at QF = 0 (left), QF = 10 (right)

Dirac output sequence, lossless encoding
PSNR: 100.00, Bitrate: 513.995KBps

Fig. 5.22: Dirac compressed “susie.yuv" SDTV sequence (lossless encoding)

Table 5.6: Performance analysis of Dirac at different QF for SDTV sequence

QF	Size(KB)*	Bitrate(KBps)	Compression ratio	PSNR-Y(dB)	PSNR-U(dB)	PSNR-V(dB)	SSIM	MSE-Y	MSE-U	MSE-V
0	64	3.613	352	33.678	33.776	33.758	0.892	27.878	27.260	27.371
1	75	4.265	298	34.970	35.016	34.864	0.902	20.707	20.488	21.217
2	92	5.218	243	35.729	35.666	35.482	0.910	17.385	17.638	18.403
3	104	5.886	216	36.351	35.666	36.165	0.915	15.065	15.102	15.726
4	127	7.221	176	37.615	37.261	36.990	0.925	11.261	12.218	13.004
5	159	8.990	141	38.479	37.975	37.669	0.932	9.229	10.365	11.121
6	212	12.031	106	39.751	39.041	38.742	0.943	6.886	8.110	8.688
7	303	17.196	74	41.161	40.200	39.893	0.955	4.978	6.209	6.665
8	449	25.503	50	42.540	41.411	40.982	0.965	3.623	4.699	5.187
9	677	38.465	33	43.640	42.613	42.123	0.972	2.812	3.562	3.989
10	1046	59.465	21	44.750	43.584	43.144	0.977	2.178	2.849	3.153
Lossless	9036	513.995	2	100.000	100.000	100.000	1.000	0.000	0.000	0.000

** indicates the encoded file size including all 25 frames after compression*

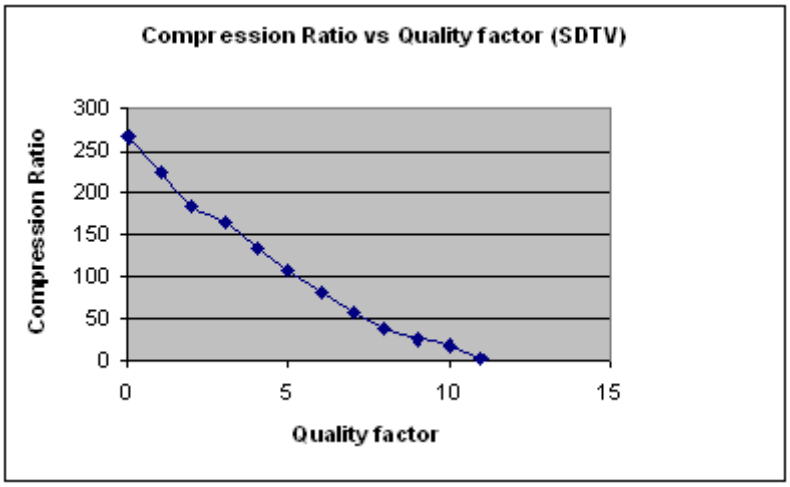


Fig. 5.23: Compression achieved by Dirac for “susie” SDTV sequence

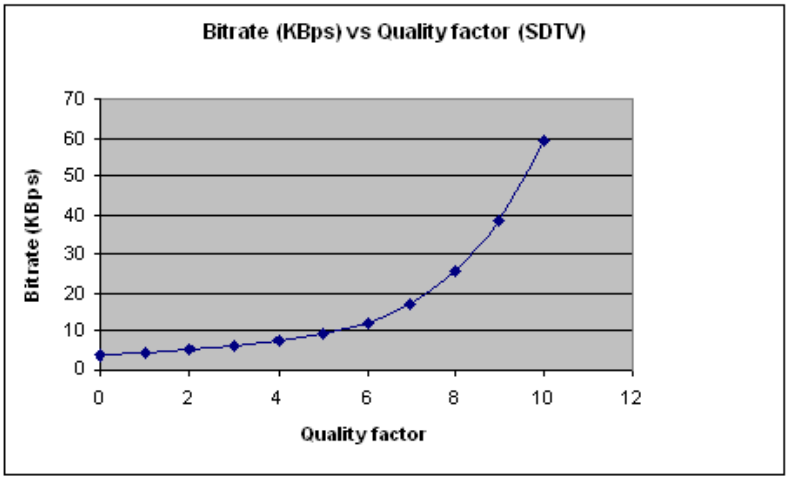


Fig. 5.24: Bitrate achieved by Dirac for “susie” SDTV sequence

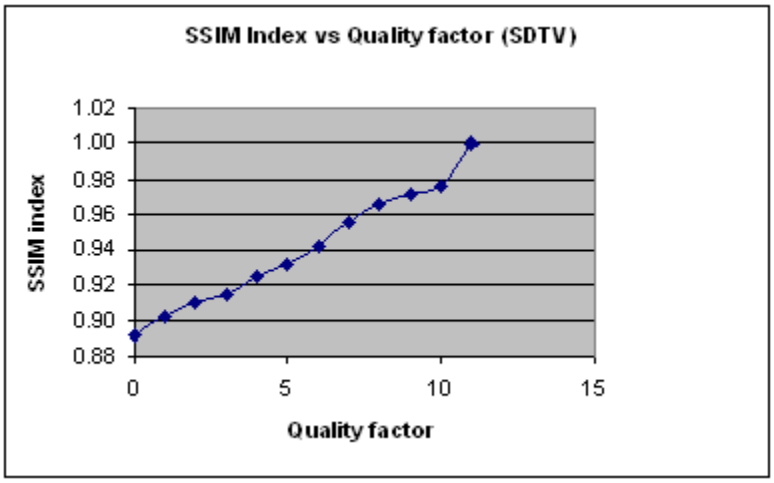


Fig. 5.25: SSIM achieved by Dirac for “susie” SDTV sequence

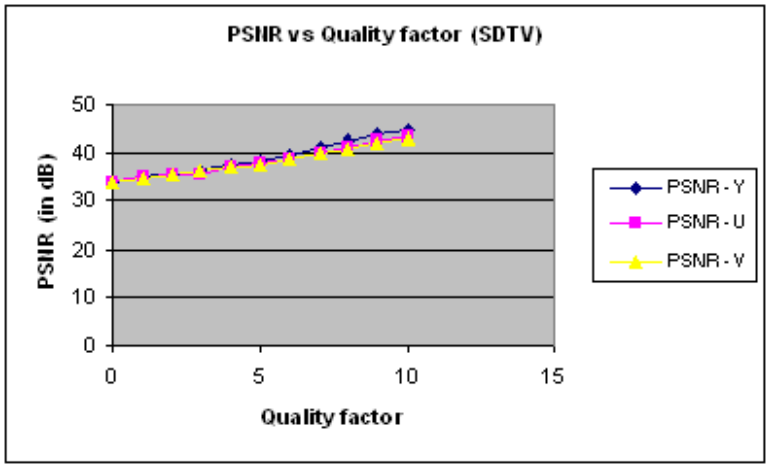


Fig. 5.26: PSNR achieved by Dirac for “susie” SDTV sequence

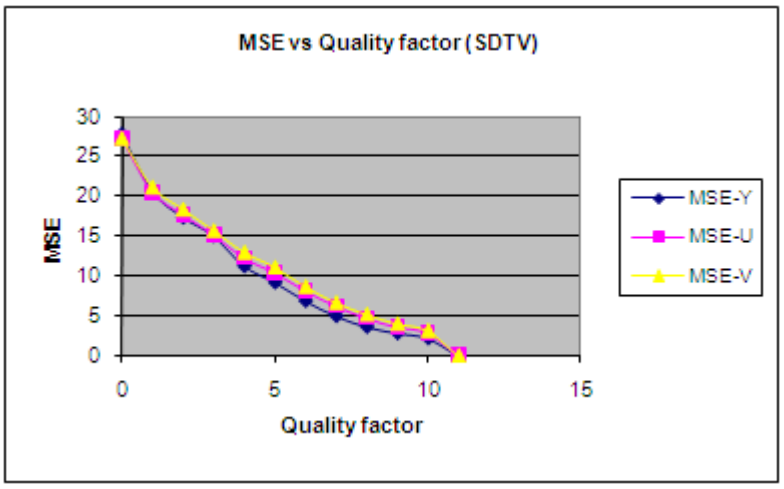


Fig. 5.27: MSE achieved by Dirac for “susie” SDTV sequence

IV. Performance comparison of Dirac and H.264 for “miss-america.yuv” QCIF sequence

Dirac output sequence, CBR=10KBps
PSNR: 38.913, Bitrate: 9.955KBps

H.264 output sequence, CBR=10KBps
PSNR: 44.162, Bitrate: 10.603KBps

Fig. 5.28: Comparison of Dirac and H.264 at CBR = 10KBps, QCIF

Dirac output sequence, CBR=100KBps
PSNR: 46.640, Bitrate: 101.351KBps

H.264 output sequence, CBR=100KBps
PSNR: 49.826, Bitrate: 102.472KBps

Fig. 5.29: Comparison of Dirac and H.264 at CBR = 100KBps, QCIF

Table 5.7: Performance comparison of Dirac with H.264 at CBR for QCIF sequence

	Dirac					H.264				
CBR(KB/s	Size(KB)*	Compression	PSNR-Y(dB)	MSE(Y)	SSIM	Size(KB)*	Compression	PSNR-Y(dB)	MSE(Y)	SSIM
10	59	95	38.913	8.353	0.966	63	90	44.162	2.494	0.983
20	120	46	42.911	3.326	0.981	123	45	45.729	1.739	0.987
40	247	23	44.648	2.230	0.986	243	23	47.257	1.223	0.989
80	477	12	46.180	1.567	0.988	481	12	49.054	0.808	0.992
100	594	9	46.640	1.410	0.989	601	9	49.826	0.677	0.993
160	949	6	47.717	1.100	0.991	911	6	52.073	0.403	0.995
200	1186	5	48.420	0.936	0.992	912	6	52.077	0.403	0.995

** indicates encoded file size including all 150 frames after compression*

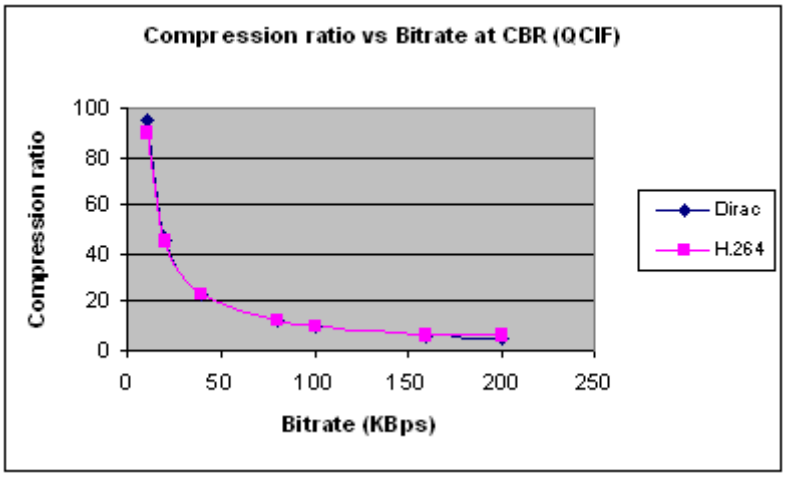


Fig. 5.30: Compression ratio comparison of Dirac and H.264 for “miss-america” QCIF sequence

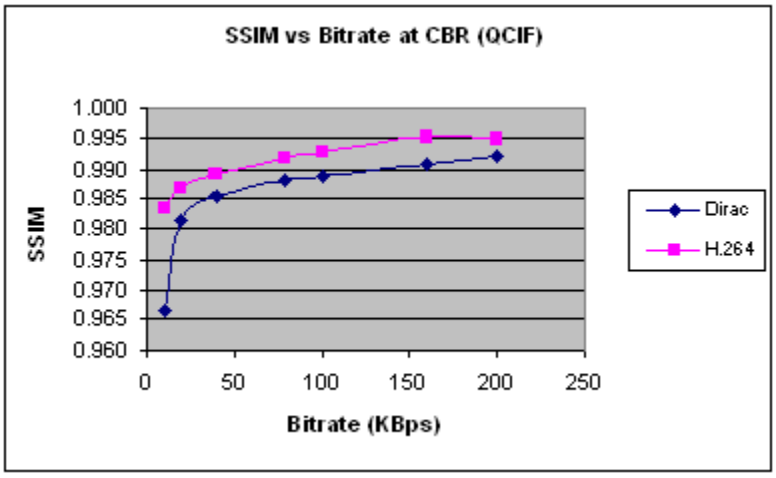


Fig. 5.31: SSIM comparison of Dirac and H.264 for “miss-america” QCIF sequence

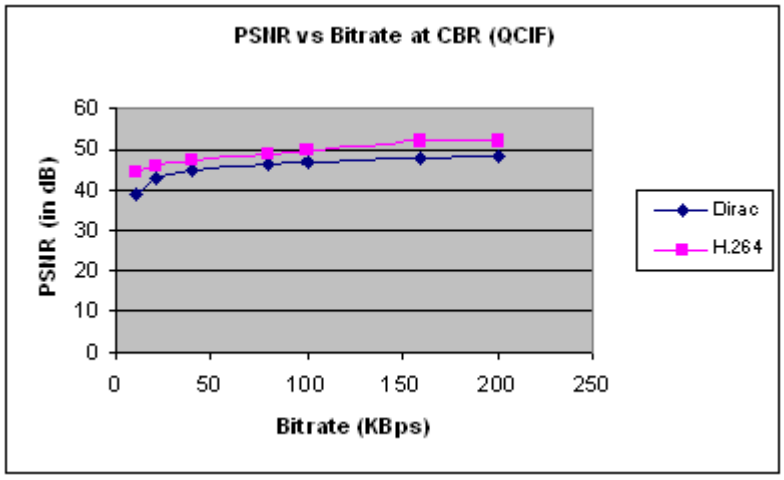


Fig. 5.32: PSNR comparison of Dirac and H.264 for “miss-america” QCIF sequence

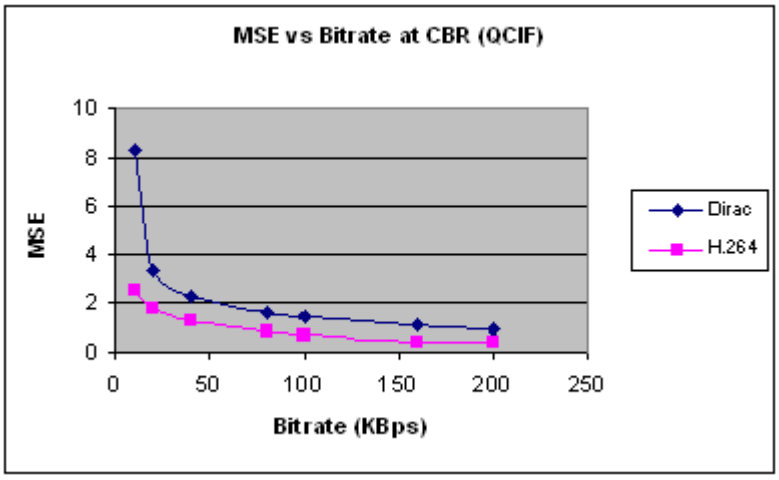
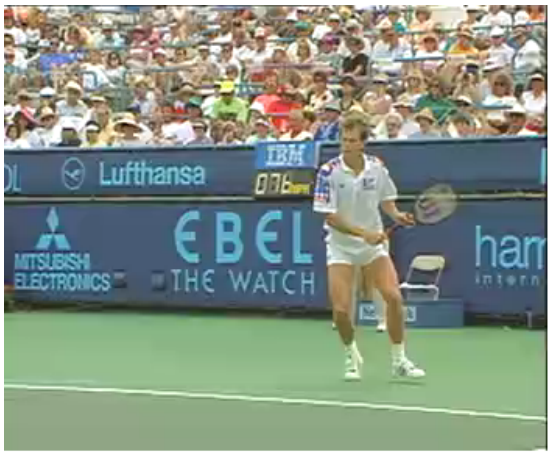


Fig. 5.33: MSE comparison of Dirac and H.264 for “miss-america” QCIF sequence

V. Performance comparison of Dirac and H.264 for “stefan.yuv” CIF sequence

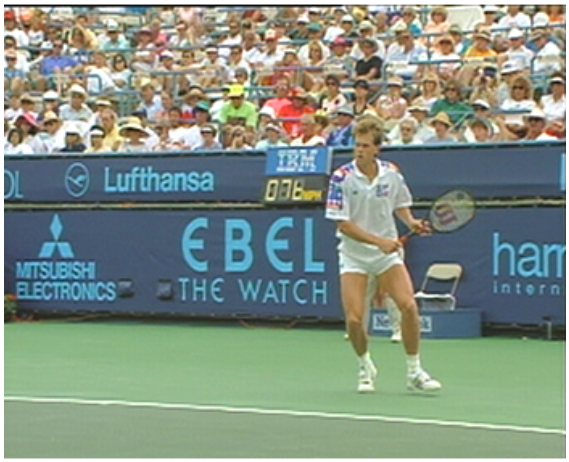
Dirac output sequence, CBR = 10KBps
PSNR: 27.468 , Bitrate: 10.644KBps



H.264 Output sequence, CBR = 10KBps
PSNR: 31.617 , Bitrate: 10.349KBps

Fig. 5.34: Comparison of Dirac and H.264 at CBR = 10KBps, CIF

Dirac output sequence, CBR = 100KBps
PSNR: 40.343, Bitrate: 101.331KBps



H.264 Output sequence, CBR = 100KBps
PSNR: 43.134, Bitrate: 101.546KBps

Fig. 5.35: Comparison of Dirac and H.264 at CBR = 100KBps, CIF

Table 5.8: Performance comparison of Dirac with H.264 at CBR for CIF sequence

	Dirac					H.264				
CBR(KB/s)	Size(KB)*	Compression	PSNR-Y(dB)	MSE(Y)	SSIM	Size(KB)*	Compression	PSNR-Y(dB)	MSE(Y)	SSIM
10	146	92	27.468	116.476	0.896	142	94	31.617	44.815	0.955
20	285	47	31.613	44.851	0.951	282	48	34.650	22.288	0.974
40	559	24	35.296	19.207	0.975	559	24	38.055	10.176	0.984
80	1114	12	39.012	8.164	0.986	1112	12	42.103	4.007	0.991
100	1386	10	40.343	6.008	0.988	1389	10	43.134	3.160	0.992
160	2216	6	43.273	3.061	0.992	2199	6	46.840	1.346	0.995
200	2757	5	44.684	2.211	0.994	2731	5	48.729	0.871	0.997

** indicates encoded file size including all 90 frames after compression*

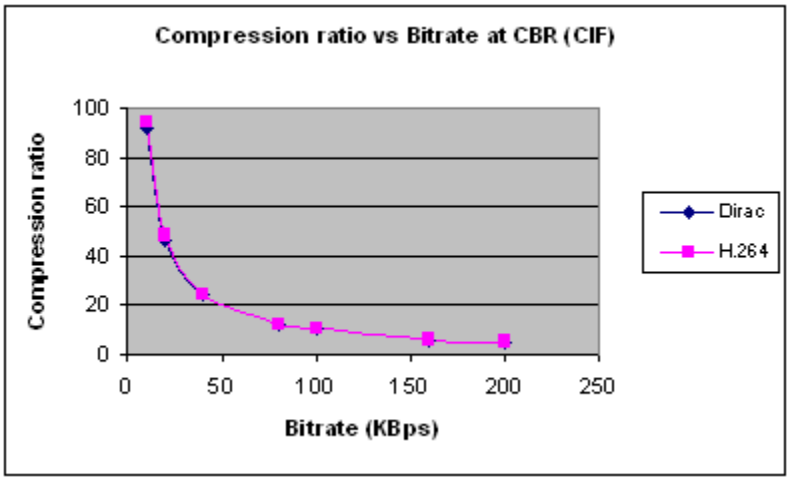


Fig. 5.36: Compression ratio comparison of Dirac and H.264 for “stefan” CIF sequence

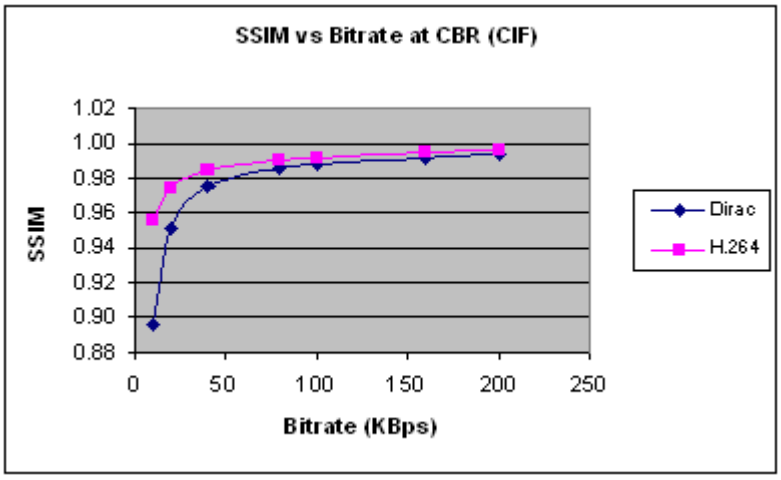


Fig. 5.37: SSIM comparison of Dirac and H.264 for “stefan” CIF sequence

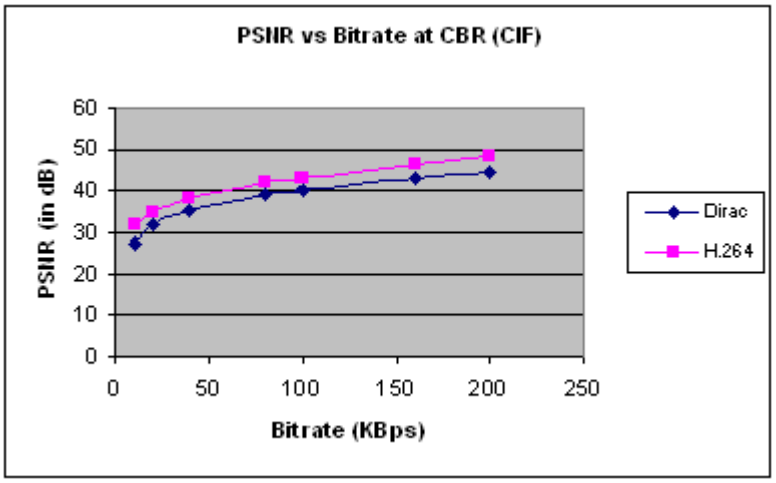


Fig. 5.38: PSNR comparison of Dirac and H.264 for “stefan” CIF sequence

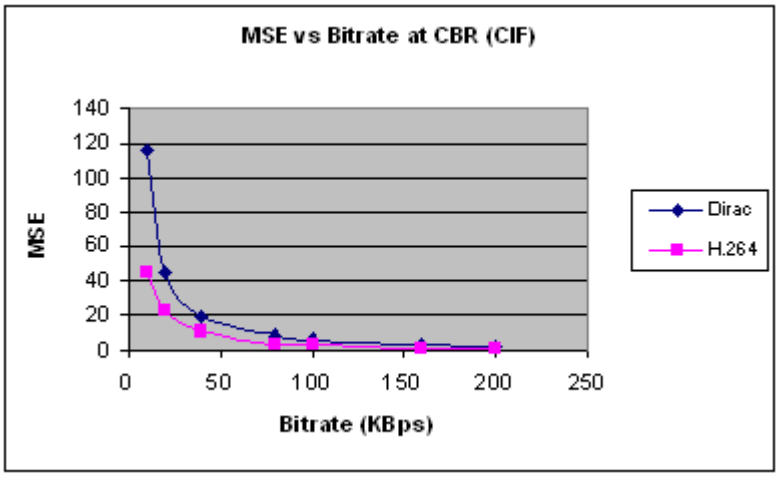


Fig. 5.39: MSE comparison of Dirac and H.264 for “stefan” CIF sequence

VI. Performance comparison of Dirac and H.264 for “susie.yuv” SDTV sequence

Dirac output sequence, CBR = 10KBps
PSNR: 10.214, Bitrate: 39.055KBps

H.264 Output sequence, CBR = 10KBps
PSNR: 41.028, Bitrate: 10.124KBps

Fig. 5.40: Comparison of Dirac and H.264 at CBR = 10KBps, SDTV

Dirac output sequence, CBR = 100KBps
PSNR: 44.676, Bitrate: 103.606KBps

H.264 Output sequence, CBR = 100KBps
PSNR: 47.491, Bitrate: 96.324KBps

Fig. 5.41: Comparison of Dirac and H.264 at CBR = 100KBps, SDTV

Table 5.9: Performance comparison of Dirac with H.264 at CBR for SDTV sequence

	Dirac					H.264				
CBR(KB/s)	Size(KB)*	Compression	PSNR-Y(dB)	MSE(Y)	SSIM	Size(KB)*	Compression	PSNR-Y(dB)	MSE(Y)	SSIM
10	180	94	39.055	8.084	0.937	178	95	41.028	5.132	0.958
20	388	44	41.729	4.367	0.960	361	47	41.530	4.571	0.962
40	751	22	43.220	3.098	0.970	701	24	44.814	2.146	0.976
80	1470	11	44.276	2.429	0.976	1405	12	45.871	1.682	0.981
100	1822	9	44.676	2.215	0.978	1694	10	47.491	1.159	0.986
160	2849	6	45.589	1.796	0.983	2562	7	50.016	0.648	0.991
200	3539	5	45.988	1.638	0.985	2953	6	50.819	0.539	0.993

** indicates encoded file size including all 25 frames after compression*

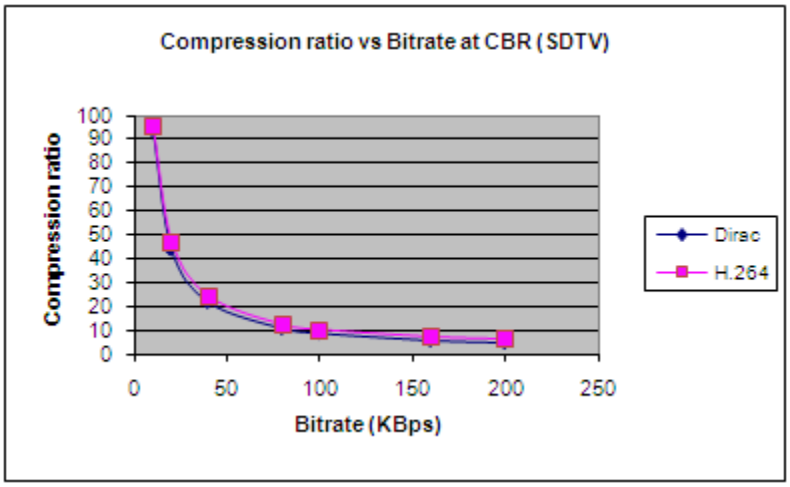


Fig. 5.42: Compression ratio comparison of Dirac and H.264 for “susie” SDTV sequence

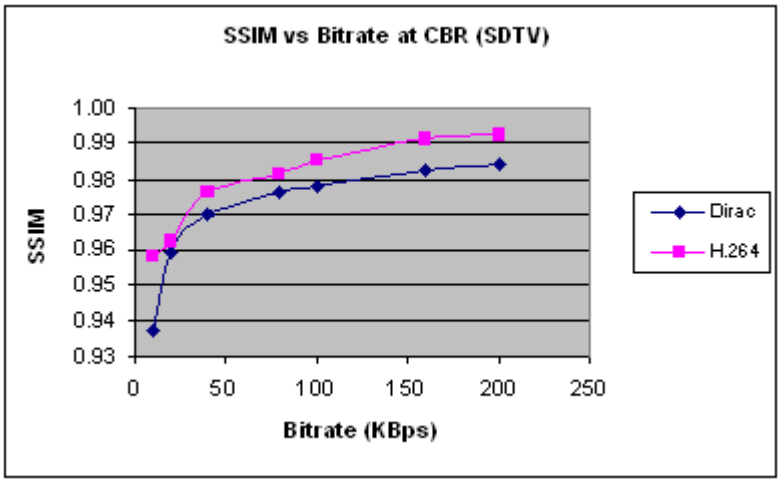


Fig. 5.43: SSIM comparison of Dirac and H.264 for “susie” SDTV sequence

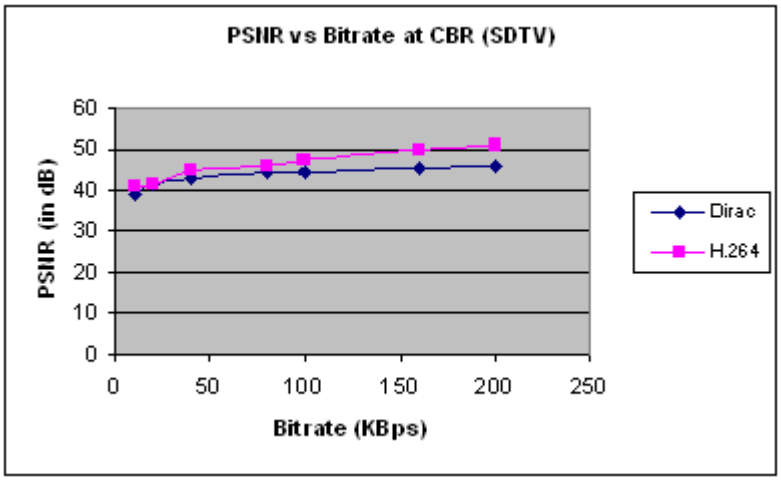


Fig. 5.44: PSNR comparison of Dirac and H.264 for “susie” SDTV sequence

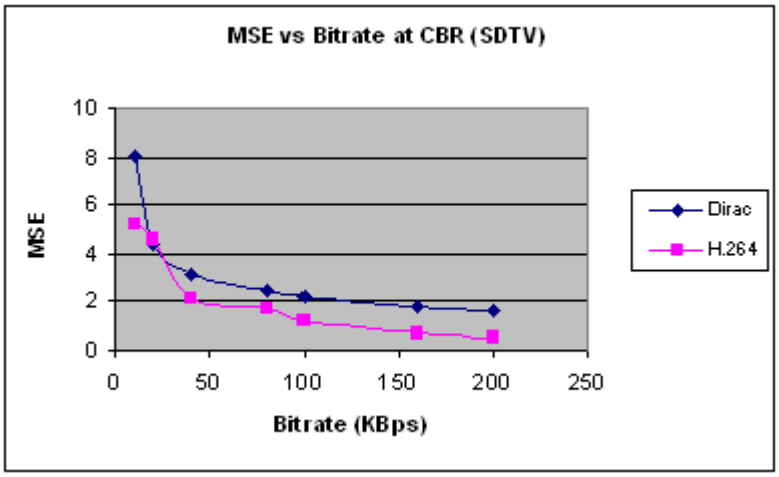


Fig. 5.45: MSE comparison of Dirac and H.264 for “susie” SDTV sequence

5.2 Conclusions

Overall Dirac is very promising. According to BBC R&D, Dirac was developed with a view to optimize its performance be it compression ratio and perceptual quality at the forefront. Its simplistic nature provides robustness which is a very beneficial feature, therefore to a large extent Dirac has succeeded in its aim. [5]

Fig.5.5, Fig.5.14 and Fig.5.23 show Dirac's compression performance at QF ranging from 0-10 and for lossless encoding. Compression is linked to bitrate i.e when there is high compression, there is less bitrate and vice versa.

Fig.5.6, Fig.5.15 and Fig.5.24 depict the variation of bitrate whilst increasing the QF of Dirac. The relationship between QF and bitrate is of a non-linear type, with the bitrate nearly exponentially increasing at QF > 8 for SD media. Dirac handles smaller video sizes such as QCIF better than bigger ones such as SD, HD media. Therefore Dirac should be using QF between 5-10 for encoding SD, HD data as encoding with higher QF will produce an intolerable increase in bitrate. [5]

Structural similarity (SSIM) operates by way of comparing local patterns of pixel intensities that have been normalized for luminance and contrast [23] (see Appendix A). This basically means that SSIM is calculated based on the combination of luminance similarity, contrast similarity and structural similarity encompassed into one value. The maximum possible value for SSIM is 1, which indicates the encoded sequence is an exact replica of the reference sequence. [5] SSIM is an alternative method of objectively evaluating video quality. [5]

Fig.5.7, Fig.5.16 and Fig.5.25 indicate SSIM evaluation of Dirac for “miss-america”, “stefan” and “susie” sequences. It can be seen that there is an initial large variation in SSIM for QF 0-3 for all video sequences and then it tends to saturate for QF > 8.

Fig.5.8, Fig.5.17 and Fig.5.26 show the PSNR analysis of Dirac and Fig.5.9, Fig.5.18 and Fig.5.27 show the MSE of the Dirac encoded sequences w.r.t to the original test sequences. From these tests, it is evident that PSNR improves with increasing QF while the reverse is true in case of MSE.

As with all video codec evaluations there is a need to maintain a compatible testbed, this would usually entail the two codecs to be tested under same bitrates. [5] It is possible to set a “target rate” for both the codecs and this would prevail over quality i.e. QF in case of Dirac. This would ensure that both codecs were being used under equal operating environments. In these tests QF has been replaced with the bit rate metric (KBps).

By evaluating the magnitude of the *.drc and *.264 files, compression ratio results in comparison to the file size of the original sequence were produced from Dirac and H.264 respectively. Fig.5.30, Fig.5.36 and Fig.5.42 show a comparison of how Dirac and H.264 perform in compression for QCIF, CIF and SDTV sequences respectively. Dirac achieves slightly higher compression ratios for lower bitrates than H.264 in case of QCIF sequences. At higher QCIF bitrates both Dirac and H.264 achieve similar compression. H.264 provides better compression at higher bitrates especially for CIF and SD, HD media.

Fig.5.31, Fig.5.37 and Fig.5.43 show the SSIM comparison of Dirac and H.264 for “miss-america” QCIF, “stefan” CIF and “susie” SD sequences. From these figures it appears that H.264 has slightly better SSIM than Dirac for all the test sequences.

Similarly, H.264 achieves better PSNR and lower MSE than Dirac as seen in Fig.5.32, Fig.5.38 and Fig.5.44; Fig.5.33, Fig.5.39 and Fig.5.45. Since Dirac has not been designed to maximize PSNR, measurements of PSNR performance against other codecs are not meaningful. [1]

Expert viewing suggests that, despite its simple toolset, Dirac is very comparable to other state-of-the-art codecs such as H.264. [1] SSIM indicates that H.264 has slightly greater improvement. However the question remains whether the enormous cost in royalty fees justifies the additional increase in quality. [5] Both Dirac and H.264 maintain a near constant quality at low bit rates, which is beneficial for applications such as video streaming which work under these environments.

In conclusion, Dirac is an extremely simple yet robust codec, [5] nearly equaling H.264 in performance, sans the cost of royalty fees.

5.3 Future research

This implementation of the Dirac codec is directed towards high-quality video compression from web video up to ultra HD. However, the standard just defines a video codec and has no mention of any audio compression. It is necessary to associate an audio stream along with the video in order to have a meaningful delivery of the video to the end user.

The Dirac video codec can be further improved by integrating it with an audio codec such as or MPEG Layer 2 (MP2) [55] or the AAC [34]. MP2 is royalty free, applicable to high quality audio and has performance similar to MP3 [57] at higher bit rates. The Dirac research group at BBC also suggests Vorbis [54] audio codec and FLAC (Free lossless audio codec)[53] developed by Xiph.Org Foundation as high quality audio formats available under royalty free terms that could be used with Dirac video codec.

According to Dr. T. Borer, Dirac team leader, the Dirac codec has to be wrapped in some container format so that it can be used with audio and then put into transport streams with any audio format supported by MPEG-TS [56]. These include uncompressed audio, MPEG layer 2, MPEG layer 3 and AAC.

Hence it is possible to transcode Dirac by multiplexing the video and audio coded bit streams to create a single bit stream for transmission and de-multiplexing the streams at the receiving end. This can be followed by synchronization of the audio and video during playback so that it can be suitable for various applications.

APPENDIX A

STRUCTURAL SIMILARITY INDEX (SSIM)

This appendix introduces a new image quality measure called “Structural similarity index” and explains how it can be computed.

Natural image signals are highly structured. Their pixels exhibit strong dependencies, especially when they are spatially proximate, and these dependencies carry important information about the structure of the objects in the visual scene. Objective methods for assessing perceptual image quality traditionally attempted to quantify the visibility of errors (differences) between a distorted image and a reference image using a variety of known properties of the human visual system. Although most quality measures based on error sensitivity decompose image signals using linear transformations, these do not remove the strong dependencies. Under the assumption that human visual perception is highly adapted for extracting structural information from a scene, Structural Similarity (SSIM), an alternative complementary framework for quality assessment based on the degradation of structural information was introduced. [23]

The Structural Similarity (SSIM) index is a method for measuring the similarity between two images. It compares local patterns of pixel intensities that have been normalized for luminance and contrast. The SSIM index is a full reference metric, in other words, the measurement of image quality based on an initial uncompressed or distortion-free image as reference. SSIM is designed to improve on traditional metrics such as PSNR and MSE, which have proved to be inconsistent with human eye perception. [38] It follows that a measure of structural information change can provide a good approximation to perceived image distortion as it considers image degradations as perceived changes in structural information variation. [23]

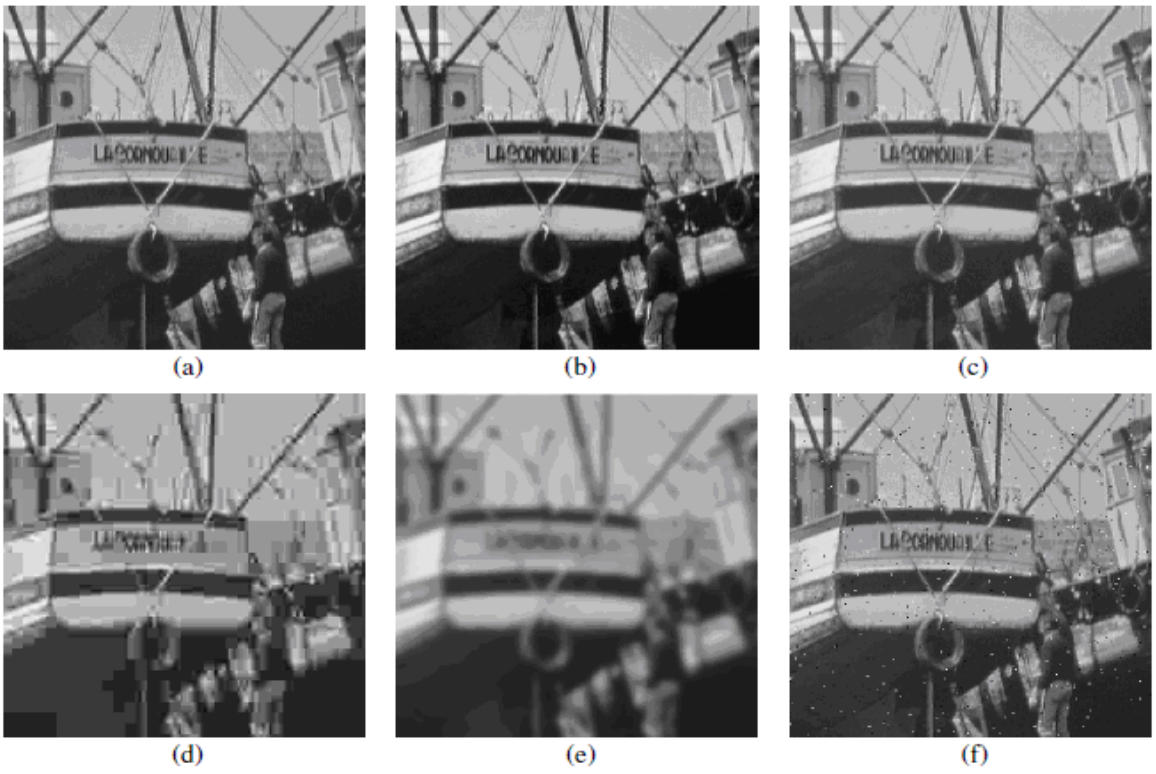


Fig. Comparison of "Boat" images with different types of distortions, all with MSE = 210. (a) Original image (8 bits/pixel; cropped from 512 × 512 to 256 × 256 for visibility). (b) Contrast-stretched image, MSSIM = 0.9168. (c) Mean-shifted image, MSSIM = 0.9900. (d) JPEG compressed image, MSSIM = 0.6949. (e) Blurred image, MSSIM = 0.7052. (f) Salt-pepper impulsive noise contaminated image, MSSIM = 0.7748.

Fig. A.1: Comparison of "Boat" images with different types of distortions [23]

In the example shown in Fig. A.1, the original "Boat" image is altered with different distortions, each adjusted to yield nearly identical MSE relative to the original image. Despite this, the images can be seen to have drastically different perceptual qualities. With the error sensitivity method, it is difficult to explain why the contrast-stretched image has very high quality in consideration of the fact that its visual difference from the reference image is easily discerned. But it is easily understood with the SSIM technique since nearly all the structural information of the reference image is preserved, in the sense that the original information can be nearly fully recovered via a simple point wise inverse linear luminance transform (except perhaps for the very bright and dark regions where saturation occurs). [23]

On the other hand, some structural information from the original image is permanently lost in the JPEG [52] compressed and the blurred images, and therefore they should be given lower quality scores than the contrast-stretched and mean-shifted images. The SSIM method is a top-down approach, mimicking the hypothesized functionality of the overall human visual system (HVS). It reduces the cognitive interaction problem to a certain extent because probing the structures of the objects being observed is thought of as the purpose of the entire process of visual observation, including high level and interactive processes. Also, the problems of natural image complexity and de-correlation are also avoided to some extent because this method does not attempt to predict image quality by accumulating the errors associated with psychophysically understood simple patterns. Instead, it proposes to evaluate the structural changes between two complex-structured signals directly. [23]

The luminance of the surface of an object being observed is the product of the illumination and the reflectance, but the structures of the objects in the scene are independent of the illumination. Consequently, to explore the structural information in an image, the influence of the illumination needs to be separated. The structural information in an image is defined as those attributes that represent the structure of objects in the scene, independent of the average luminance and contrast. Since luminance and contrast can vary across a scene, the local luminance and contrast is used to define SSIM. [23] The block diagram of the SSIM quality assessment system is shown in Fig. A.2.

Supposing there are two non negative image signals, which have been aligned with each other. If one of the signals is considered to have perfect quality, then the similarity measure can serve as a quantitative measurement of the quality of the second signal. [23]

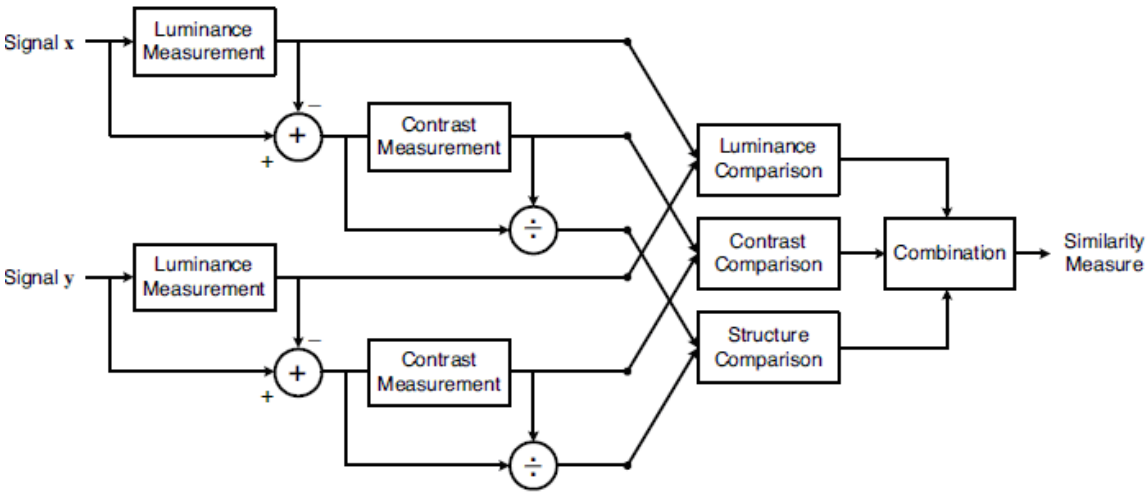


Fig. A.2: Structural similarity (SSIM) measurement system [23]

The system separates the task of similarity measurement into three comparisons: luminance, contrast and structure. First, the luminance of each signal is compared and the mean intensity is estimated. Second, the mean intensity is removed from the signal. Third, the signal is normalized (divided) by its own standard deviation. Finally, the three components are combined to yield an overall similarity measure. An important point is that the three components are relatively independent. The change of luminance and/or contrast will not affect the structures of images. The similarity measure satisfies the following conditions: symmetry, bounded-ness and unique maximum. [23]

The SSIM metric is calculated on various windows of an image. The measure between two windows of size N x N *x* and *y* is [38]:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2cov_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad \text{(A.1)}$$

Notations: μ_x: average of *x*, μ_y: average of *y*, σ_x^2: variance of *x*, σ_y^2: variance of *y*, cov_{xy}: covariance of *y*, $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ are two variables to stabilize the division with weak denominator, L : dynamic range of the pixel-values (typically this is $2^{\#bits\ per\ pixel} - 1$), $k_1 = 0.01$, $k_2 = 0.03$ [38]

In order to evaluate the image quality this formula is applied only on the luminance. The resultant SSIM index is a decimal value between -1 and 1, and value 1 is only reachable in the case of two identical sets of data. Typically it is calculated on window sizes of 8x8. The window can be displaced pixel-by-pixel on the image but the authors propose to use only a subgroup of the possible windows to reduce the complexity of the calculation. [38]

For image quality assessment, it is useful to apply the SSIM index locally rather than globally. First, image statistical features are usually highly spatially non-stationary. Second, image distortions, which may or may not depend on the local image statistics, may also be space-variant. Third, at typical viewing distances, only a local area in the image can be perceived with high resolution by the human observer at one time instance.

And finally, localized quality measurement can provide a spatially varying quality map of the image, which delivers more information about the quality degradation of the image and may be useful in some applications. [23]

The application scope of the SSIM index may not be restricted to image processing. In fact, because it is a symmetric measure, it can be thought of as a similarity measure for comparing any two signals. The signals can be either discrete or continuous, and can live in a space of arbitrary dimensionality. [23]

APPENDIX B

DIRAC PRO

This appendix introduces DiracPRO [58], a version of the Dirac family of video compression tools, optimized for professional production and archiving applications with emphasis on quality and low latency.

DiracPRO is a non proprietary, royalty free, open technology designed for simplicity, efficiency and speed, and intended for high quality applications with lower compression ratios. It can work on all major operating systems and can be easily imported onto a wide range of hardware - from specialist signal processors to application-specific LSI circuits. [58]

DiracPRO (a.k.a SMPTE VC-2) is the intra frame version of Dirac, currently being standardized by the SMPTE to become SMPTE S2042. The main difference between Dirac and DiracPRO is in the final process of compression i.e. arithmetic coding. Arithmetic coding is hardware intensive and introduces undesirable delay at high bitrates and therefore is omitted in DiracPRO. [58] DiracPRO instead uses a form of context adaptive binary arithmetic coding (CABAC), similar to the H.264/AVC. There are substantial differences between the AVC implementation and DiracPRO but the result for both is improved compression. [59]

Another point of DiracPRO's significance lies in transcoding with other codecs such as the H.264. According to Dr. T. Borer, Dirac team leader, transcoding between H.264 and Dirac has limited relevance because the application domains do not overlap much. On the other hand, transcoding long GOP from H.264 to DiracPRO is more significant. There will be requirements to transcode from H.264 to VC-2 when video content moves from a low bandwidth link (e.g. over a satellite) to a higher bandwidth medium.

DiracPRO supports the following features: [58]

- Intra-frame only (forward and backward prediction modes are also available)
- Supports 10 bit chroma format - 4:2:2
- No sub-sampling
- Lossless or visually lossless compression
- Low latency on encoding / decoding
- Robust over multiple passes
- Ease of transport (can use a range of transport standards)
- Low complexity for decoding
- Open specification
- Multiple vendor
- Support for multiple HD image formats and frame rates.

DiracPRO is aimed at professional applications not end user distribution. Typically it would be used for compression ratios upto 20:1. [59] DiracPRO has a wide range of applications from lossless HD to < 50Mbit/s that includes (but not limited to) the following: [58]

- Lossless or visually lossless compression for archives.
- Mezzanine compression for re-use of existing equipment.
- Low latency compression for live video links.

REFERENCES

[1] T. Borer, and T. Davies, “Dirac video compression using open technology”, BBC EBU Technical Review, July 2005

[2] BBC Research on Dirac: http://www.bbc.co.uk/rd/projects/dirac/index.shtml

[3] H. Eeckhaut, et al, “Speeding up Dirac’s entropy coder”, Proc. 5th WSEAS Int. Conf. on Multimedia, Internet and Video Technologies, pp. 120-125, Greece, Aug. 2005.

[4] The Dirac web page: http://dirac.sourceforge.net

[5] K. Onthriar, K. K. Loo and Z. Xue, “Performance comparison of emerging Dirac video codec with H.264/AVC”, IEEE International Conference on Digital Telecommunications, 2006, ICDT apos; Vol. 06, Page: 22, Issue: 29-31, Aug. 2006.

[6] T. Davies, “The Dirac Algorithm”: http://dirac.sourceforge.net/documentation/algorithm/, 2005.

[7] M. Tun and W. A. C. Fernando, “An error-resilient algorithm based on partitioning of the wavelet transform coefficients for a DIRAC video codec” , Tenth International Conference on Information Visualization, 2006, IV, Vol. 5-7 , pp.: 615 – 620, Issue : July 2006.

[8] T. Davies, “A modified rate-distortion optimization strategy for hybrid wavelet video coding”, IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006, Vol.: 2, pp.: II, Publication Date: 14-19 May 2006.

[9] M. Tun, K. K. Loo and J. Cosmas, “Semi-hierarchical motion estimation for the Dirac video codec”, 2008 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, pp.: 1-6, Publication Date: March 31 2008-April 2 2008.

[10] /CMPT 365 Course Slides/, School of Computing Science, Simon Fraser University,fig3: http://www.cs.sfu.ca/CourseCentral/365/li/material/notes/Chap4/Chap4.3/Chap4.3.html

[11] Dirac developer support: Overlapped block-based motion compensation: http://dirac.sourceforge.net/documentation/algorithm/algorithm/toc.htm

[12] S.K.Kwon, A.Tamhankar and K.R.Rao, “Overview of H.264 / MPEG-4 Part 10” J. Visual Communication and Image Representation, Vol 17, pp.186-216, April 2006.

[13] T.Wiegand, et al “Overview of the H.264/AVC video coding standard”, IEEE Trans. CSVT, Vol.13, pp 560-576, July 2003.

[14] T. Wiegand and G. J. Sullivan, “The H.264 video coding standard”, IEEE Signal Processing Magazine, vol. 24, pp. 148-153, March 2007.

[15] D. Marpe, T. Wiegand and G. J. Sullivan, “The H.264/MPEG-4 AVC standard and its applications”, IEEE Communications Magazine, vol. 44, pp. 134-143, Aug. 2006.

[16] I. Richardson, “The H.264 AVC standard”, to be published in late 2009.

[17] C. Gargour et al, “A short introduction to wavelets and their applications”, IEEE Circuits and Systems Magazine, vol. 9, pp. 57-68, II Quarter, 2009.

[18] “Dirac Pro to bolster BBC HD links”: http://www.broadcastnow.co.uk/news/multi-platform/news/dirac-pro-to-bolster-bbc-hd-links/1732462.article

[19] “And now, Dirac from the Olympics, a new free codec!”:
http://www.videsignline.com/210601739
[20] ‘Dirac codec’: http://en.wikipedia.org/wiki/Dirac_(codec)
[21] ‘Video encoding’: http://www.webopedia.com/TERM/V/video_encoding.html
[22] “Video encoding and bandwidth requirements”:
http://www.cybertechmedia.com/videoencoding.html
[23] Z. Wang, et al “Image quality assessment: From error visibility to structural similarity”, IEEE Trans. on Image Processing, vol. 13, pp. 600-612, Apr. 2004.
http://www.ece.uwaterloo.ca/~z70wang/
[24] http://en.wikipedia.org/wiki/Windows_Media_Video
[25] Microsoft Windows Media: http://www.microsoft.com/windows/windowsmedia
[26] MPEG-4 Part 2, ISO/IEC 14496-2, International Organization for Standardization, http://www.iso.ch
[27] http://en.wikipedia.org/wiki/MPEG-4_Part_2
[28] Dirac software and source code: http://diracvideo.org/download/dirac-research/
[29] http://en.wikipedia.org/wiki/VC-1
[30] Dirac video codec - A programmer's guide:
http://dirac.sourceforge.net/documentation/code/programmers_guide/toc.htm
[31] H. Jia and L. Zhang, “Directional diamond search pattern for fast block motion estimation”, IEE Electronics Letters, Vol. 39, No. 22, pp. 1581-1583, 30th Oct. 2003.
[32] Video test sequences (YUV 4:2:0): http://trace.eas.asu.edu/yuv/index.html

[33] Video test sequences ITU601:

http://www.cipr.rpi.edu/resource/sequences/itu601.html

[34] MPEG–2 advanced audio coding, AAC. International Standard IS 13818–7, ISO/IEC JTC1/SC29 WG11, 1997

[35] G. A. Davidson, et al “ATSC video and audio coding”, Proceedings of IEEE, vol. 94, pp. 60-76, Jan. 2006

[36] A. Puri, X. Chen and A. Luthra, “Video coding using the H.264/MPEG-4 AVC compression standard”, Signal Processing: Image Communication, vol. 19, pp. 793-849, Oct. 2004

[37] H.264 AVC JM software: http://iphome.hhi.de/suehring/tml/

[38] SSIM: http://en.wikipedia.org/wiki/SSIM

[39] Daubechies wavelet: http://en.wikipedia.org/wiki/Daubechies_wavelet

[40] Daubechies wavelet filter design: http://cnx.org/content/m11159/latest/

[41] Dirac developer support: Wavelet transform:

http://dirac.sourceforge.net/documentation/algorithm/algorithm/wlt_transform.xht

[42] Dirac developer support: RDO motion estimation metric:

http://dirac.sourceforge.net/documentation/algorithm/algorithm/rdo_mot_est.xht

[43] Digital down converter: http://en.wikipedia.org/wiki/Digital_down_converter

[44] T. Jacob, K. R. Rao and D. N. Kim, “Image Mirroring, Rotation and Interpolation in Wavelet Domain”, Applications of Digital Image Processing XXXII conference, SPIE symposium on Optical Engineering + Applications, 2-6 Aug. 2009.

[45] H.264/MPEG-4 AVC: http://en.wikipedia.org/wiki/H.264

[46] M.Fieldler, “Implementation of basic H.264/AVC Decoder”, seminar paper at Chemnitz University of Technology, June 2004

[47] H.264 encoder and decoder: http://www.adalta.it/Pages/407/266881_266881.jpg

[48] “H.264 video compression standard”, White paper, Axis communications.

[49] MPEG-4: ISO/IEC JTC1/SC29 14496-10: Information technology – Coding of audio-visual objects - Part 10: Advanced Video Coding, ISO/IEC, 2005.

[50] D. Kumar, P. Shastry and A. Basu, “Overview of the H.264 / AVC”, 8th Texas Instruments Developer Conference India, 30 Nov – 1 Dec 2005, Bangalore.

[51] R. Schäfer, T. Wiegand and H. Schwarz, “The emerging H.264/AVC standard”, EBU Technical Review, Jan. 2003.

[52] Joint Photographic Experts Group, JPEG: http://www.jpeg.org/

[53] FLAC - Free Lossless Audio Codec: http://flac.sourceforge.net/

[54] Vorbis: http://www.vorbis.com/

[55] MPEG Layer II: http://en.wikipedia.org/wiki/MPEG-1_Audio_Layer_II

[56] MPEG-TS / MPEG transport stream: http://en.wikipedia.org/wiki/MPEG_transport_stream

[57] MP3 / MPEG Layer III: http://en.wikipedia.org/wiki/MP3

[58] Dirac Pro: http://www.bbc.co.uk/rd/projects/dirac/diracpro.shtml

[59] T. Borer, “Dirac coding: Tutorial & Implementation”, EBU Networked Media Exchange seminar, June 2009.

[60] DiracPRO software: http://dirac.kw.bbc.co.uk/download/

BIOGRAPHICAL INFORMATION

Aruna Ravi has received her Bachelor of Engineering degree in Electrical and Electronics Engineering from Anna University, India, in June 2007. She is currently pursuing her Master's degree in Electrical Engineering at The University of Texas at Arlington. She is a member of the multimedia processing research group, guided by Dr. K. R. Rao. Her research interests include video processing, digital signal processing and wireless telecommunications.