PERFORMANCE ANALYSIS AND COMPARISON OF THE 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 since 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 compression ratio, quality, bit rates and error resilience. However unlike Dirac, it requires payment of patent fees.
The objective of this thesis is to analyze 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 analyzed at various constant ‘target’ bit rates ranging from 10KBps to 200KBps. The test results indicate that Dirac’s performance is comparable to that of H.264.
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Thesis outline</td>
<td>3</td>
</tr>
<tr>
<td>2. OVERVIEW OF DIRAC VIDEO CODEC</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Dirac architecture</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Dirac encoder</td>
<td>4</td>
</tr>
<tr>
<td>2.3 Dirac decoder</td>
<td>6</td>
</tr>
<tr>
<td>2.4 Features of Dirac</td>
<td>7</td>
</tr>
<tr>
<td>2.5 Advantages of Dirac</td>
<td>8</td>
</tr>
<tr>
<td>2.6 Applications of Dirac</td>
<td>10</td>
</tr>
<tr>
<td>2.7 Summary</td>
<td>12</td>
</tr>
</tbody>
</table>
3. OVERVIEW OF H.264 VIDEO CODEC ................................................................. 13
   3.1 H.264 / AVC - Introduction ........................................................................ 13
   3.2 H.264 / AVC profiles ................................................................................ 15
   3.3 H.264 encoder and decoder ....................................................................... 20
   3.4 Features of H.264 .................................................................................. 39
   3.5 Applications of H.264 / MPEG-4 Part10 .................................................. 40
   3.6 Summary .............................................................................................. 41
4. ENCODING AND DECODING IN DIRAC ................................................... 42
   4.1 Need for encoding .................................................................................. 42
   4.2 Wavelet transform .................................................................................. 42
   4.3 Scaling and quantization ........................................................................ 46
   4.4 Entropy coding ...................................................................................... 47
   4.5 Motion estimation ................................................................................... 48
   4.6 Motion compensation ............................................................................. 52
   4.7 Decoding ............................................................................................... 55
   4.8 Summary .............................................................................................. 55
5. RESULTS, CONCLUSIONS AND FUTURE RESEARCH ............................... 56
   5.1 Implementation and results .................................................................... 56
   5.2 Conclusions ........................................................................................... 84
   5.3 Future research ..................................................................................... 87
APPENDIX

A. STRUCTURAL SIMILARITY INDEX (SSIM) ................................................................. 88

B. DIRAC PRO ........................................................................................................ 95

C. QUALITY FACTOR IN DIRAC ........................................................................... 98

REFERENCES ............................................................................................................ 101

BIOGRAPHICAL INFORMATION .......................................................................... 106
# LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Dirac’s encoder architecture</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Dirac’s decoder architecture</td>
<td>6</td>
</tr>
<tr>
<td>3.1 Supporting picture format – 4:2:0 chroma sampling</td>
<td>14</td>
</tr>
<tr>
<td>3.2 Specific coding parts for H.264 profiles</td>
<td>19</td>
</tr>
<tr>
<td>3.3 Basic H.264 / AVC macro-block coding structure</td>
<td>20</td>
</tr>
<tr>
<td>3.4 Basic block diagram of H.264 encoder</td>
<td>21</td>
</tr>
<tr>
<td>3.5 Block diagram emphasizing transform</td>
<td>22</td>
</tr>
<tr>
<td>3.6 Assignment of indices of the DC (dark samples) to luma 4 x 4 block</td>
<td>23</td>
</tr>
<tr>
<td>3.7 Chroma DC coefficients for 4x4 IntDCT</td>
<td>25</td>
</tr>
<tr>
<td>3.8 Transform, scaling and quantization at H.264 encoder</td>
<td>26</td>
</tr>
<tr>
<td>3.9 Comparison of CAVLC and CABAC</td>
<td>27</td>
</tr>
<tr>
<td>3.10 H.264 scan orders to read residual data</td>
<td>28</td>
</tr>
<tr>
<td>3.11 De-blocking filter process</td>
<td>29</td>
</tr>
<tr>
<td>3.12 Intra-prediction block</td>
<td>31</td>
</tr>
<tr>
<td>3.13 Mode decisions for intra prediction</td>
<td>32</td>
</tr>
<tr>
<td>3.14 Inter prediction in H.264</td>
<td>33</td>
</tr>
<tr>
<td>3.15 Segmentations of the macro-block for motion compensation</td>
<td>34</td>
</tr>
<tr>
<td>3.16 Block diagram emphasizing sub-pel motion compensation</td>
<td>35</td>
</tr>
</tbody>
</table>
5.6 Bit rate achieved by Dirac for “miss-america” QCIF sequence

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

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

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

5.10 Original “stefan.yuv” CIF sequence

5.11 “stefan.yuv” (a) Y component, (b) Cb component, (c) Cr component

5.12 Dirac compressed “stefan.yuv” CIF sequence at QF = 0, QF = 10

5.13 Dirac compressed “stefan.yuv” CIF sequence (lossless encoding)

5.14 Compression achieved by Dirac for “stefan” CIF sequence

5.15 Bit rate achieved by Dirac for “stefan” CIF sequence

5.16 SSIM achieved by Dirac for “stefan” CIF sequence

5.17 PSNR achieved by Dirac for “stefan” CIF sequence

5.18 MSE achieved by Dirac for “stefan” CIF sequence

5.19 Original “susie.yuv” SDTV sequence

5.20 “susie.yuv” (a) Y component, (b) Cb component, (c) Cr component

5.21 Dirac compressed “susie.yuv” SDTV sequence at QF = 0, QF = 10

5.22 Dirac compressed “susie.yuv” SDTV sequence (lossless encoding)

5.23 Compression achieved by Dirac for “susie” SDTV sequence

5.24 Bit rate achieved by Dirac for “susie” SDTV sequence

5.25 SSIM achieved by Dirac for “susie” SDTV sequence

5.26 PSNR achieved by Dirac for “susie” SDTV sequence
5.27 MSE achieved by Dirac for “susie” SDTV sequence
5.28 Comparison of Dirac and H.264 at CBR = 10KBps, QCIF
5.29 Comparison of Dirac and H.264 at CBR = 100KBps, QCIF
5.30 Compression ratio comparison of Dirac and H.264 for “miss-america” QCIF sequence
5.31 SSIM comparison of Dirac and H.264 for “miss-america” QCIF sequence
5.32 PSNR comparison of Dirac and H.264 for “miss-america” QCIF sequence
5.33 MSE comparison of Dirac and H.264 for “miss-america” QCIF sequence
5.34 Comparison of Dirac and H.264 at CBR = 10KBps, CIF
5.35 Comparison of Dirac and H.264 at CBR = 100KBps, CIF
5.36 Compression ratio comparison of Dirac and H.264 for “stefan” CIF sequence
5.37 SSIM comparison of Dirac and H.264 for “stefan” CIF sequence
5.38 PSNR comparison of Dirac and H.264 for “stefan” CIF sequence
5.39 MSE comparison of Dirac and H.264 for “stefan” CIF sequence
5.40 Comparison of Dirac and H.264 at CBR = 10KBps, SDTV
5.41 Comparison of Dirac and H.264 at CBR = 100KBps, SDTV
5.42 Compression ratio comparison of Dirac and H.264 for “susie” SDTV sequence
5.43 SSIM comparison of Dirac and H.264 for “susie” SDTV sequence
5.44 PSNR comparison of Dirac and H.264 for “susie” SDTV sequence
5.45 MSE comparison of Dirac and H.264 for “susie” SDTV sequence
<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Algorithmic element comparison of Dirac with other codecs</td>
<td>11</td>
</tr>
<tr>
<td>3.1 H.264 profiles for particular applications</td>
<td>16</td>
</tr>
<tr>
<td>3.2 H.264 / MPEG-4 Part 10 profile specifications</td>
<td>17</td>
</tr>
<tr>
<td>5.1 Basic information for QCIF sequence “miss-america.yuv”</td>
<td>57</td>
</tr>
<tr>
<td>5.2 Performance analysis of Dirac at different QF for QCIF sequence</td>
<td>59</td>
</tr>
<tr>
<td>5.3 Basic information for CIF sequence “stefan.yuv”</td>
<td>62</td>
</tr>
<tr>
<td>5.4 Performance analysis of Dirac at different QF for CIF sequence</td>
<td>65</td>
</tr>
<tr>
<td>5.5 Basic information for SDTV sequence “susie.yuv”</td>
<td>68</td>
</tr>
<tr>
<td>5.6 Performance analysis of Dirac at different QF for SDTV sequence</td>
<td>71</td>
</tr>
<tr>
<td>5.7 Performance comparison of Dirac with H.264 at CBR for QCIF sequence</td>
<td>74</td>
</tr>
<tr>
<td>5.8 Performance comparison of Dirac with H.264 at CBR for CIF sequence</td>
<td>78</td>
</tr>
<tr>
<td>5.7 Performance comparison of Dirac with H.264 at CBR for SDTV sequence</td>
<td>81</td>
</tr>
</tbody>
</table>
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
PSNR: Peak signal to noise ratio
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

Video compression has become necessary in the widespread use of multimedia over the internet. Data compression is used to exploit limited storage and transmission capacity as efficiently as possible which is important for the internet and high definition media.

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,\cite{Dirac} and as such competes with existing formats such as H.264 \cite{H.264} - \cite{H.264_4} and WMV \cite{WMV} \cite{WMV_2}. Dirac was used internally by the BBC to transmit HDTV pictures at the Beijing Olympics in 2008.\cite{Beijing} \cite{Beijing_2}

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 \cite{MPEG-4} \cite{MPEG-4_2} and Microsoft's WMV 9 \cite{WMV} \cite{WMV_2}. 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.\cite{VC-1}

1
Dirac's implementers make the preliminary claim of "a two-fold reduction in bit rate over MPEG-2 for high definition video",\cite{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\cite{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.\cite{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\cite{4}\cite{28} is not intended simply to provide reference coding and decoding. It is a prototype implementation that can freely be modified and deployed. Dirac’s decoder in particular is designed to be fast and more agile than other conventional decoders. The resulting specification is simple and straightforward to implement and optimized for real-time performance.\cite{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.
In the above figure, $V_{in}$ is the input video frame; $P$ refers to the previous motion compensated frame. Motion compensated prediction error (MCPE) is obtained as the difference between $V_{in}$ and $P$; $e_{TQ}$ is the MCPE after wavelet transformation and quantization; $e'$ refers to MCPE after inverse transformation; $V_{local} = P + e'$
2.3 Dirac decoder

The Dirac decoder \cite{1} \cite{30} performs the inverse operations of the encoder as depicted in Fig. 2.2.

![Dirac decoder architecture](image)

**Fig. 2.2:** Dirac decoder architecture

In the above figure, $P$ represents the motion compensated previous frame; $e$ refers to the MCPE after inverse wavelet transformation and quantization; $V_{out}$ represents the output video frame.
2.4 Features of Dirac

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 up to 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:

- No 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 \textsuperscript{[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 \cite{35}

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

The above table shows the algorithmic element comparison of Dirac with other state-of-the-art codecs such as MPEG-2, MPEG-4 and WMV9. It is evident that Dirac is much more simple yet robust, ensuring good performance.
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. The various features, advantages and the numerous applications of Dirac were also discussed in detail. 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 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 (group of protocols responsible for encapsulating application data blocks into data units suitable for transfer through the network) 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]
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). 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

<table>
<thead>
<tr>
<th>Profile</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>Video Conferencing, Videophone</td>
</tr>
<tr>
<td>Main</td>
<td>Digital Storage Media, Television Broadcasting</td>
</tr>
<tr>
<td>Extended</td>
<td>Streaming Video</td>
</tr>
<tr>
<td>High</td>
<td>Content contribution, Content distribution, Studio editing, Post processing</td>
</tr>
</tbody>
</table>
The common coding parts for the profiles are listed below:\textsuperscript{[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](image)

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.

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]

![Block diagram emphasizing transform](image)

Fig. 3.5: Block diagram emphasizing transform[12]
Figure 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.

![Figure 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]

\[
Y = (C_f x C_f^T) \otimes E_f
\]

\[
Y = \begin{bmatrix}
1 & 1 & 1 & 1 \\
2 & 1 & -1 & -2 \\
1 & -1 & -1 & 1 \\
1 & -2 & 2 & -1
\end{bmatrix}
\begin{bmatrix}
x_00 \\
x_{01} \\
x_{02} \\
x_{03}
\end{bmatrix}
\begin{bmatrix}
1 & 2 & 1 & 1 \\
1 & 1 & -1 & -2 \\
1 & -1 & -1 & 2 \\
1 & -2 & 1 & -1
\end{bmatrix}
\begin{bmatrix}
a^2 \\
\frac{a b}{2} \\
\frac{a b}{4} \\
\frac{a^2}{2}
\end{bmatrix}
\begin{bmatrix}
\frac{a b}{2} \\
\frac{a b}{4} \\
\frac{a b}{2} \\
\frac{a b}{4}
\end{bmatrix}
\]

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

\[
a = \frac{1}{2}, \quad b = \frac{\sqrt{2}}{\sqrt{5}}, \quad d = \frac{1}{2}
\]
The inverse 4x4 DCT can be represented by the following equation:

\[
X' = C_i^T \left( Y \otimes E_i \right) C_i
\]  

(3.2)

where \( C_i = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \) and \( E_i = \begin{bmatrix} a^2 & \frac{ab}{2} & \frac{ab}{2} & \frac{ab}{2} \\ \frac{ab}{2} & b^2 & \frac{ab}{2} & b^2 \\ \frac{ab}{2} & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{ab}{2} & \frac{ab}{2} & b^2 \end{bmatrix} \)

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

\[
Y_D = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_{D00} & x_{D01} & x_{D02} & x_{D03} \\ x_{D10} & x_{D11} & x_{D12} & x_{D13} \\ x_{D20} & x_{D21} & x_{D22} & x_{D23} \\ x_{D30} & x_{D31} & x_{D32} & x_{D33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \]  

(3.3)

where // represents rounding to the nearest integer.

The Walsh – Hadamard transform can be represented as follows:

\[
Y_D = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} DC_{00} & DC_{01} \\ DC_{10} & DC_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad 4:2:0
\]  

(3.4)
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:  

\[
B_y = A_y \left\lfloor \frac{SF_{ij}}{Q_{step}} \right\rfloor
\]

(3.5)

where \(A\) is the quantizer input, \(B\) refers to the quantizer output, \(Q_{step}\) is the quantization parameter and \(SF\) is the scaling term.

Fig. 3.8 shows the transform, scaling and quantization blocks at the encoder part of H.264 / MPEG-4.
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. \cite{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. \cite{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. Typically CABAC provides 10-15 % reduction in bit rate compared to CAVLC, for the same PSNR as shown in Fig. 3.9.

![Comparison of CAVLC and CABAC](image)

**Fig. 3.9: Comparison of CAVLC and CABAC**

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 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 \([M\text{ Zeroes}] [1] [M\text{ bits of INFO}]\). 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 \(\text{INFO} = \text{code_num} + 1 - 2M\)). The length of each Exp-Golomb codeword is \((2M+1)\) bits where \(M = \text{Floor}(\log_2 (\text{code_num} + 1))\). \([12]\)

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. \([12]\)

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. Vertical edges are filtered first (left to right) followed by the horizontal edges (top to bottom) as shown in Fig. 3.11.

![De-blocking filter process](image)

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.\textsuperscript{[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.\textsuperscript{[50]} In High Profile, there is also adaptive directional intra prediction for 8x8 blocks. The intra prediction block is shown in Fig. 3.12.
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.

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 16x16 pixels. 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 ($\Delta$) 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. $^{[51]}$
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. An inverse transform is carried out on all the blocks in order to map them from transform domain to pixel domain. A 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]

\[ A'_y = B_y \cdot Q_{\text{step}} \cdot SF_y \]  

(3.6)

where \( B \): inverse quantizer input, \( A' \): inverse quantizer output, \( Q_{\text{step}} \): 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.
3.4 Features of H.264\textsuperscript{[45]}\textsuperscript{[50]}

H.264/AVC/MPEG-4 Part 10 \textsuperscript{[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
- 4x4 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

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 are illustrated and the encoding, 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. 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 speed of the standard 56K modem 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.

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 than block based transforms. 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.
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. 4.1: Perfect reconstruction analysis and synthesis filter pairs

Fig. 4.2: Dirac’s wavelet transform architecture \[^{[5]}\]
Daubechies wavelet filters \cite{39} \cite{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. \cite{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. \cite{41}

One filter available in Dirac is an approximation of the Daubechies (9, 7) low pass wavelet filter whose lifting stages are defined as follows: \cite{41}

\begin{align*}
    s^0_n &= x_{2n} \\
    d^0_n &= x_{2n+1} \\
    d^1_n &= d^0_n - \left( 6497 \cdot \left( s^0_n + s^0_{n+1} \right) \right) / 4096 \\
    s^1_n &= s^0_n - \left( 217 \cdot \left( d^1_n + d^1_{n-1} \right) \right) / 4096 \\
    d^2_n &= d^1_n + \left( 3616 \cdot \left( s^1_n + s^1_{n+1} \right) \right) / 4096 \\
    s^2_n &= s^1_n + \left( 1817 \cdot \left( d^2_n + d^2_{n-1} \right) \right) / 4096
\end{align*}

(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. \cite{41}
At the other extreme is the (5, 3) Daubechies high pass filter.\textsuperscript{[41]}

\[
\begin{align*}
   d_n^1 &= d_n^0 - (s_n^0 + s_{n+1}^0) / 2 \\
   s_n^2 &= s_n^0 + (d_n^1 + d_{n-2}^1) / 4
\end{align*}
\] (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.

![Stages of wavelet transform](Image)

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.\textsuperscript{[41]}
A Dirac-coded picture is free from block artifacts and is clearly superior to pictures coded by block-based transforms in the case of moving images.\cite{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 steps twice as wide. This allows Dirac to perform coarser quantization on smaller values compared to other codecs such as the MPEG-4.\cite{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.

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.

![Hierarchical motion estimation](image)

**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 since 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 the lowest level is diamond shaped [31] with the search range 5 and all other levels except the lowest level use a square shaped search pattern with a search range of 1. Fig. 4.8 shows both search patterns where there are altogether 61 search points in the diamond shape and 9 points in the square shape. [9]

![Search patterns in Dirac](image)
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 the zero motion vector and predicted motion vector respectively with a diamond search pattern. The motion vector is spatially predicted, and is the median vector of the left, the top left and the top blocks of the current block where motion estimation is carried out. The sum of the absolute difference (SAD) is used as the cost function. Given two blocks \(X, Y\) of samples, the SAD is given by:

\[
SAD(X, Y) = \sum_{i,j} |X_{i,j} - Y_{i,j}|
\]  

Initially the SAD calculation is carried out only for the center point of the 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 by 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, the 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 motion vector.

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, the vectors for blocks to the left and above the left are available for calculating the median vector.
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 group of pictures (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 \( p(x,y,t) \) 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](image)

Each macro-block may be split into prediction units consisting either of 16 individual blocks, or of an array of 4 mid-size blocks, termed sub-macro-blocks, 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]

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 the entropy decoding technique. Next, scaling and inverse quantization is performed. In the final stage, inverse transform is applied on the data to produce the decoded, uncompressed video output.

![Diagram of decoding process](image)

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 based on the 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 described in detail. The advantages of using wavelet transform, overlapped block based motion compensation and entropy coding techniques are also discussed.
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 an 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 bit rate, compression, SSIM, PSNR and MSE have been recorded.

To ensure the accuracy of the tests, each codec must be encoded using the same bit rate. [5] Since the latest version of Dirac does include a constant bit rate (CBR) mode, the comparison between Dirac and H.264’s performance was produced by encoding several test sequences at different bit rates. Utilizing the CBR mode within H.264, we can ensure that H.264 is being encoded at the same bit rate as that of Dirac.
Objective tests are separated into five sections, namely (i) Compression, (ii) Bit rate, (iii) Structural similarity index (SSIM), (iv) Peak signal to noise ratio (PSNR) and (v) Mean square error (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 values of the QF.

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

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

<table>
<thead>
<tr>
<th>Frame rate (fps)</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration (sec)</td>
<td>6</td>
</tr>
<tr>
<td>Width</td>
<td>176</td>
</tr>
<tr>
<td>Height</td>
<td>144</td>
</tr>
<tr>
<td>Bit rate (KB/sec)</td>
<td>950.4</td>
</tr>
<tr>
<td>Size (KB)</td>
<td>5569</td>
</tr>
<tr>
<td>Number of frames</td>
<td>150</td>
</tr>
</tbody>
</table>

Fig. 5.1: Original “miss-america.yuv” QCIF sequence
Fig. 5.2: “miss-americayuv” (a) Y component, (b) Cb component and (c) Cr component

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

<table>
<thead>
<tr>
<th>QF</th>
<th>Size(KB)</th>
<th>Bitrate(Kbps)</th>
<th>Compression ratio</th>
<th>PSNR-Y(dB)</th>
<th>PSNR-U(dB)</th>
<th>PSNR-V(dB)</th>
<th>SSIM</th>
<th>MSE-Y</th>
<th>MSE-U</th>
<th>MSE-V</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>30</td>
<td>4.963</td>
<td>192</td>
<td>30.627</td>
<td>34.100</td>
<td>31.187</td>
<td>0.882</td>
<td>25.280</td>
<td>49.478</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>5.255</td>
<td>181</td>
<td>31.883</td>
<td>34.890</td>
<td>32.406</td>
<td>0.892</td>
<td>22.444</td>
<td>37.362</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>33</td>
<td>5.686</td>
<td>170</td>
<td>32.977</td>
<td>35.006</td>
<td>33.160</td>
<td>0.917</td>
<td>20.533</td>
<td>31.414</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>36</td>
<td>6.020</td>
<td>166</td>
<td>33.664</td>
<td>35.299</td>
<td>33.640</td>
<td>0.930</td>
<td>19.960</td>
<td>29.122</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>41</td>
<td>6.095</td>
<td>136</td>
<td>35.269</td>
<td>35.452</td>
<td>34.652</td>
<td>0.944</td>
<td>18.329</td>
<td>28.122</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>49</td>
<td>8.211</td>
<td>116</td>
<td>37.035</td>
<td>35.340</td>
<td>35.959</td>
<td>0.957</td>
<td>16.707</td>
<td>20.489</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>61</td>
<td>10.361</td>
<td>92</td>
<td>38.140</td>
<td>37.970</td>
<td>37.679</td>
<td>0.969</td>
<td>15.765</td>
<td>19.587</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>81</td>
<td>13.702</td>
<td>68</td>
<td>41.341</td>
<td>37.975</td>
<td>38.768</td>
<td>0.976</td>
<td>14.775</td>
<td>18.873</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>115</td>
<td>19.601</td>
<td>48</td>
<td>44.480</td>
<td>38.008</td>
<td>41.666</td>
<td>0.982</td>
<td>13.183</td>
<td>16.018</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>186</td>
<td>31.686</td>
<td>30</td>
<td>44.224</td>
<td>40.296</td>
<td>43.162</td>
<td>0.985</td>
<td>2.458</td>
<td>8.160</td>
<td>3.147</td>
</tr>
<tr>
<td>10</td>
<td>308</td>
<td>52.651</td>
<td>18</td>
<td>46.643</td>
<td>41.244</td>
<td>44.476</td>
<td>0.987</td>
<td>1.773</td>
<td>4.883</td>
<td>2.320</td>
</tr>
<tr>
<td>Lossless</td>
<td>2927</td>
<td>482.3</td>
<td>2</td>
<td>100.000</td>
<td>100.000</td>
<td>100.000</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

* Indicates encoded file size including all 150 frames after compression

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

59
Fig. 5.6: Bit rate 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”

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame rate (fps)</td>
<td>6.4285</td>
</tr>
<tr>
<td>Duration (sec)</td>
<td>14</td>
</tr>
<tr>
<td>Width</td>
<td>352</td>
</tr>
<tr>
<td>Height</td>
<td>288</td>
</tr>
<tr>
<td>Bit rate (KB/sec)</td>
<td>977.55</td>
</tr>
<tr>
<td>Size (KB)</td>
<td>13365</td>
</tr>
<tr>
<td>Number of frames</td>
<td>90</td>
</tr>
</tbody>
</table>

Fig. 5.10: Original “stefan.yuv” CIF sequence
Fig. 5.11: “stefan.yuv” (a) Y component, (b) $C_b$ component and (c) $C_r$ component
Fig. 5.12: Dirac compressed "stefan.yuv" CIF sequence at QF = 0 (left), QF = 10 (right)

Dirac output sequence, QF = 0
PSNR: 21.327, Bitrate: 5.68 KBps

Dirac output sequence, QF = 10
PSNR: 42.667, Bitrate: 156.740 KBps

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

Dirac output sequence, lossless encoding
PSNR: 100.00, Bitrate: 821.337 KBps
Table 5.4: Performance analysis of Dirac at different QF for CIF sequence

<table>
<thead>
<tr>
<th>QF</th>
<th>Size(KB)*</th>
<th>Bitrate(Kbps)</th>
<th>Compression ratio</th>
<th>PSNR-Y(dB)</th>
<th>PSNR-U(dB)</th>
<th>PSNR-V(dB)</th>
<th>SSIM</th>
<th>MSE-Y</th>
<th>MSE-U</th>
<th>MSE-V</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>60</td>
<td>1,333</td>
<td>140</td>
<td>21.327</td>
<td>29.174</td>
<td>29.045</td>
<td>0.886</td>
<td>479.000</td>
<td>78.646</td>
<td>67.392</td>
</tr>
<tr>
<td>1</td>
<td>85</td>
<td>1,600</td>
<td>116</td>
<td>23.601</td>
<td>30.425</td>
<td>30.406</td>
<td>0.773</td>
<td>299.828</td>
<td>89.699</td>
<td>58.227</td>
</tr>
<tr>
<td>2</td>
<td>117</td>
<td>1,961</td>
<td>86</td>
<td>25.281</td>
<td>30.757</td>
<td>30.945</td>
<td>0.840</td>
<td>192.738</td>
<td>64.029</td>
<td>62.305</td>
</tr>
<tr>
<td>3</td>
<td>151</td>
<td>2,521</td>
<td>74</td>
<td>27.182</td>
<td>31.572</td>
<td>31.722</td>
<td>0.887</td>
<td>124.125</td>
<td>47.408</td>
<td>45.738</td>
</tr>
<tr>
<td>4</td>
<td>201</td>
<td>3,860</td>
<td>56</td>
<td>29.418</td>
<td>32.183</td>
<td>33.480</td>
<td>0.826</td>
<td>74.436</td>
<td>80.515</td>
<td>26.515</td>
</tr>
<tr>
<td>5</td>
<td>277</td>
<td>4,840</td>
<td>40</td>
<td>31.800</td>
<td>38.848</td>
<td>35.808</td>
<td>0.850</td>
<td>44.866</td>
<td>26.806</td>
<td>19.154</td>
</tr>
<tr>
<td>6</td>
<td>397</td>
<td>6,850</td>
<td>28</td>
<td>33.829</td>
<td>36.721</td>
<td>37.397</td>
<td>0.888</td>
<td>28.108</td>
<td>17.419</td>
<td>12.007</td>
</tr>
<tr>
<td>7</td>
<td>593</td>
<td>9,796</td>
<td>19</td>
<td>35.824</td>
<td>37.519</td>
<td>39.950</td>
<td>0.978</td>
<td>17.012</td>
<td>11.514</td>
<td>6.281</td>
</tr>
<tr>
<td>8</td>
<td>889</td>
<td>14,560</td>
<td>13</td>
<td>37.787</td>
<td>38.845</td>
<td>40.157</td>
<td>0.933</td>
<td>10.024</td>
<td>8.491</td>
<td>6.272</td>
</tr>
<tr>
<td>9</td>
<td>1270</td>
<td>21,316</td>
<td>9</td>
<td>40.031</td>
<td>40.231</td>
<td>41.977</td>
<td>0.937</td>
<td>3.457</td>
<td>6.185</td>
<td>4.221</td>
</tr>
<tr>
<td>10</td>
<td>1670</td>
<td>31,363</td>
<td>6</td>
<td>42.867</td>
<td>42.395</td>
<td>44.420</td>
<td>0.931</td>
<td>3.519</td>
<td>3.745</td>
<td>2.330</td>
</tr>
<tr>
<td>Loadless</td>
<td>11250</td>
<td>621,337</td>
<td>0</td>
<td>100.000</td>
<td>100.000</td>
<td>100.000</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

* Indicates encoded file size including all 90 frames after compression

Fig. 5.14: Compression achieved by Dirac for “stefan” CIF sequence
Fig. 5.15: Bit rate 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”

<table>
<thead>
<tr>
<th>Frame rate (fps)</th>
<th>1.389</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration(sec)</td>
<td>18</td>
</tr>
<tr>
<td>Width</td>
<td>720</td>
</tr>
<tr>
<td>Height</td>
<td>480</td>
</tr>
<tr>
<td>Bit rate(KB/sec)</td>
<td>960.077</td>
</tr>
<tr>
<td>Size (KB)</td>
<td>16875</td>
</tr>
<tr>
<td>Number of frames</td>
<td>25</td>
</tr>
</tbody>
</table>

Fig. 5.19: Original “susie.yuv” SDTV sequence
Fig. 5.20: “susie.yuv” (a) Y component, (b) $C_b$ component and (c) $C_r$ component
Fig. 5.21: Dirac compressed “susie.yuv" SDTV sequence at QF = 0 (left), QF = 10 (right)

Dirac output sequence, QF = 0
PSNR: 33.673, Bitrate: 3.133Kbps

Dirac output sequence, QF = 10
PSNR: 44.730, Bitrate: 5.416Kbps

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

Dirac output sequence, lossless encoding
PSNR: 100.00, Bitrate: 515.995Kbps
Table 5.6: Performance analysis of Dirac at different QF for SDTV sequence

<table>
<thead>
<tr>
<th>QF</th>
<th>Size(KB)</th>
<th>Bitrate(Kbps)</th>
<th>Compression ratio</th>
<th>PSNR-Y(dB)</th>
<th>PSNR-U(dB)</th>
<th>PSNR-V(dB)</th>
<th>SSIM</th>
<th>MSE-Y</th>
<th>MSE-U</th>
<th>MSE-V</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>64</td>
<td>3.613</td>
<td>352</td>
<td>33.679</td>
<td>33.779</td>
<td>33.759</td>
<td>0.892</td>
<td>27.870</td>
<td>27.360</td>
<td>27.371</td>
</tr>
<tr>
<td>1</td>
<td>75</td>
<td>4.266</td>
<td>299</td>
<td>34.870</td>
<td>35.018</td>
<td>34.884</td>
<td>0.902</td>
<td>20.707</td>
<td>20.460</td>
<td>21.217</td>
</tr>
<tr>
<td>2</td>
<td>92</td>
<td>5.210</td>
<td>243</td>
<td>35.729</td>
<td>35.886</td>
<td>35.402</td>
<td>0.910</td>
<td>17.306</td>
<td>16.630</td>
<td>16.893</td>
</tr>
<tr>
<td>3</td>
<td>104</td>
<td>6.096</td>
<td>218</td>
<td>36.351</td>
<td>36.886</td>
<td>36.185</td>
<td>0.915</td>
<td>15.086</td>
<td>15.102</td>
<td>16.728</td>
</tr>
<tr>
<td>4</td>
<td>127</td>
<td>7.221</td>
<td>178</td>
<td>37.615</td>
<td>37.281</td>
<td>36.900</td>
<td>0.925</td>
<td>11.281</td>
<td>12.218</td>
<td>13.004</td>
</tr>
<tr>
<td>5</td>
<td>158</td>
<td>8.990</td>
<td>141</td>
<td>38.470</td>
<td>37.975</td>
<td>37.689</td>
<td>0.932</td>
<td>9.229</td>
<td>10.365</td>
<td>11.121</td>
</tr>
<tr>
<td>6</td>
<td>212</td>
<td>12.031</td>
<td>100</td>
<td>39.751</td>
<td>39.041</td>
<td>38.742</td>
<td>0.943</td>
<td>8.888</td>
<td>8.110</td>
<td>8.888</td>
</tr>
<tr>
<td>7</td>
<td>303</td>
<td>17.186</td>
<td>74</td>
<td>41.181</td>
<td>40.200</td>
<td>39.893</td>
<td>0.955</td>
<td>4.978</td>
<td>3.208</td>
<td>6.865</td>
</tr>
<tr>
<td>8</td>
<td>448</td>
<td>25.503</td>
<td>50</td>
<td>42.540</td>
<td>41.411</td>
<td>42.082</td>
<td>0.985</td>
<td>3.023</td>
<td>4.699</td>
<td>5.187</td>
</tr>
<tr>
<td>9</td>
<td>677</td>
<td>39.465</td>
<td>33</td>
<td>43.540</td>
<td>42.913</td>
<td>42.123</td>
<td>0.972</td>
<td>2.812</td>
<td>3.562</td>
<td>3.889</td>
</tr>
<tr>
<td>10</td>
<td>1046</td>
<td>59.465</td>
<td>21</td>
<td>44.750</td>
<td>43.594</td>
<td>43.144</td>
<td>0.977</td>
<td>2.173</td>
<td>2.848</td>
<td>3.153</td>
</tr>
<tr>
<td>Lossless</td>
<td>9336</td>
<td>513.965</td>
<td>2</td>
<td>100.000</td>
<td>100.000</td>
<td>100.000</td>
<td>1.000</td>
<td>3.000</td>
<td>3.000</td>
<td>3.000</td>
</tr>
</tbody>
</table>

* 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: Bit rate 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-americayuv” QCIF sequence

![Fig. 5.28: Comparison of Dirac and H.264 at CBR = 10KBps, QCIF](image1)

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

<table>
<thead>
<tr>
<th>CBR (KB/s)</th>
<th>Dirac Size (KB)</th>
<th>Compression</th>
<th>PSNR-Y (dB)</th>
<th>MSE (Y)</th>
<th>SSIM</th>
<th>H.264 Size (KB)</th>
<th>Compression</th>
<th>PSNR-Y (dB)</th>
<th>MSE (Y)</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>59</td>
<td>95</td>
<td>38.913</td>
<td>8.353</td>
<td>0.866</td>
<td>63</td>
<td>90</td>
<td>44.162</td>
<td>2.404</td>
<td>0.983</td>
</tr>
<tr>
<td>20</td>
<td>120</td>
<td>46</td>
<td>42.811</td>
<td>3.826</td>
<td>0.981</td>
<td>123</td>
<td>45</td>
<td>46.729</td>
<td>1.730</td>
<td>0.987</td>
</tr>
<tr>
<td>40</td>
<td>247</td>
<td>23</td>
<td>44.649</td>
<td>2.230</td>
<td>0.966</td>
<td>243</td>
<td>23</td>
<td>47.257</td>
<td>1.223</td>
<td>0.993</td>
</tr>
<tr>
<td>80</td>
<td>477</td>
<td>12</td>
<td>46.180</td>
<td>1.567</td>
<td>0.988</td>
<td>481</td>
<td>12</td>
<td>49.054</td>
<td>0.800</td>
<td>0.982</td>
</tr>
<tr>
<td>100</td>
<td>594</td>
<td>9</td>
<td>48.640</td>
<td>1.410</td>
<td>0.989</td>
<td>501</td>
<td>9</td>
<td>49.928</td>
<td>0.677</td>
<td>0.993</td>
</tr>
<tr>
<td>180</td>
<td>948</td>
<td>6</td>
<td>47.717</td>
<td>1.100</td>
<td>0.981</td>
<td>911</td>
<td>6</td>
<td>52.073</td>
<td>0.403</td>
<td>0.995</td>
</tr>
<tr>
<td>200</td>
<td>1166</td>
<td>5</td>
<td>48.420</td>
<td>0.936</td>
<td>0.982</td>
<td>912</td>
<td>6</td>
<td>52.077</td>
<td>0.403</td>
<td>0.995</td>
</tr>
</tbody>
</table>

* indicates encoded file size including all 150 frames after compression

![Fig. 5.29: Comparison of Dirac and H.264 at CBR = 100KBps, QCIF](image2)
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

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

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

<table>
<thead>
<tr>
<th>CBR (Kbps)</th>
<th>Dirac Size (KB)</th>
<th>Compression</th>
<th>PSNR (dB)</th>
<th>MSE (Y)</th>
<th>SSIM</th>
<th>H.264 Size (KB)</th>
<th>Compression</th>
<th>PSNR (dB)</th>
<th>MSE (Y)</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>145</td>
<td>82</td>
<td>27.468</td>
<td>0.8306</td>
<td>0.956</td>
<td>142</td>
<td>94</td>
<td>31.617</td>
<td>0.855</td>
<td>0.965</td>
</tr>
<tr>
<td>20</td>
<td>286</td>
<td>47</td>
<td>31.613</td>
<td>0.951</td>
<td>0.951</td>
<td>262</td>
<td>48</td>
<td>34.645</td>
<td>0.974</td>
<td>0.974</td>
</tr>
<tr>
<td>40</td>
<td>553</td>
<td>24</td>
<td>35.296</td>
<td>0.975</td>
<td>0.959</td>
<td>559</td>
<td>24</td>
<td>38.055</td>
<td>1.018</td>
<td>0.964</td>
</tr>
<tr>
<td>80</td>
<td>1114</td>
<td>12</td>
<td>39.012</td>
<td>0.996</td>
<td>0.996</td>
<td>1112</td>
<td>12</td>
<td>42.103</td>
<td>1.007</td>
<td>0.991</td>
</tr>
<tr>
<td>100</td>
<td>1386</td>
<td>10</td>
<td>40.343</td>
<td>0.998</td>
<td>0.998</td>
<td>1389</td>
<td>10</td>
<td>43.134</td>
<td>1.016</td>
<td>0.992</td>
</tr>
<tr>
<td>180</td>
<td>2216</td>
<td>6</td>
<td>43.273</td>
<td>0.992</td>
<td>0.992</td>
<td>2199</td>
<td>6</td>
<td>46.840</td>
<td>1.046</td>
<td>0.995</td>
</tr>
<tr>
<td>200</td>
<td>2757</td>
<td>5</td>
<td>44.664</td>
<td>0.994</td>
<td>0.994</td>
<td>2731</td>
<td>5</td>
<td>48.729</td>
<td>0.871</td>
<td>0.997</td>
</tr>
</tbody>
</table>

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

Fig. 5.40: Comparison of Dirac and H.264 at CBR = 10KBps, SDTV
Table 5.9: Performance comparison of Dirac with H.264 at CBR for SDTV sequence

| CBR(KB/s) | Dirac Size(KB) | Compression | PSNR-Y(dB) | MSE(Y) | SSIM | H.264 Size(KB) | Compression | PSNR-Y(dB) | MSE(Y) | SSIM |
|-----------|----------------|-------------|------------|--------|------|----------------|-------------|------------|--------|------|------|
| 10        | 94             | 39.055      | 9.004      | 0.931  | 178  | 95             | 41.028      | 5.132      | 0.956  |
| 20        | 44             | 41.728      | 4.307      | 0.960  | 361  | 47             | 41.530      | 4.571      | 0.962  |
| 40        | 22             | 43.220      | 3.090      | 0.970  | 701  | 24             | 44.814      | 2.146      | 0.976  |
| 80        | 1470           | 44.276      | 2.423      | 0.878  | 1405 | 12             | 45.871      | 1.682      | 0.961  |
| 100       | 1822           | 44.876      | 2.215      | 0.878  | 1894 | 10             | 47.451      | 1.158      | 0.966  |
| 150       | 2849           | 45.589      | 1.790      | 0.883  | 2552 | 7              | 50.016      | 0.648      | 0.991  |
| 200       | 3639           | 46.988      | 1.639      | 0.885  | 2353 | 6              | 50.818      | 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]

Figures 5.5, 5.14 and 5.23 show Dirac’s compression performance at QF ranging from 0-10 and for lossless encoding. Compression is linked to bit rate i.e when there is high compression, there is a lower bit rate and vice versa.

Figures 5.6, 5.15 and 5.24 depict the variation of bit rate whilst increasing the QF of Dirac. The relationship between QF and bit rate is non-linear, with the bit rate 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 bit rate. [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]
Figures 5.7, 5.16 and 5.25 indicate SSIM evaluation of Dirac for “missamerica”, “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.

Figures 5.8, 5.17 and 5.26 show the PSNR analysis of Dirac and Figures 5.9, 5.18 and 5.27 show the MSE of the Dirac encoded sequences with respect to the original test sequences. From these tests, it is evident that PSNR improves with increasing QF while the reverse is true for MSE.

As with all video codec evaluations there is a need to maintain a compatible test bed; this would usually entail the two codecs to be tested under the same bit rates. [5] It is possible to set a “target rate” for both the codecs and this would prevail over quality i.e. QF in the 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. Figures 5.30, 5.36 and 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 bit rates than H.264 in case of QCIF sequences. At higher QCIF bit rates both Dirac and H.264 achieve similar compression. H.264 provides better compression at higher bit rates especially for CIF and SD, HD media.
Figures 5.31, 5.37 and 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) \cite{55} or the AAC \cite{34}. MP2 is royalty free, applicable to high quality audio and has performance similar to MP3 \cite{57} at higher bit rates. The Dirac research group at BBC also suggests Vorbis \cite{54} audio codec and FLAC (Free lossless audio codec)\cite{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 \cite{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.
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.\textsuperscript{[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.\textsuperscript{[38]} It follows that a measure of structural information change can provide a good approximation to perceived image distortion since it considers image degradations as perceived changes in structural information variation.\textsuperscript{[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 while considering 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. This means, 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 JPEG \cite{52} compression. The blurred images should therefore 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 to be 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. \cite{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 from the structural details. 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. \cite{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.\textsuperscript{[23]}

Fig. A.2: Structural similarity (SSIM) measurement system\textsuperscript{[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.\textsuperscript{[23]}
The SSIM metric is calculated on various windows of an image. The measure between two windows of size $N_x \times N_y$ is \[^{[38]}\]:

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

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

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 pixels. 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 bit rates 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:  

- 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. DiracPRO has a wide range of applications from lossless HD to < 50Mbit/s that includes (but not limited to) the following:  

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

QUALITY FACTOR IN DIRAC

This appendix explains the quality factor (QF) term that is extensively used in the performance analysis of the Dirac video codec.
In Dirac, the overall trade-off factor is derived from a value termed “QF”, meaning quality or quantization factor. QF is not a direct measure of quality. Coding with constant QF will ensure constant quality only on homogenous material where the trade-off between distortion and rate is constant. \[6\]

Picture lambda values are used for rate-distorion control of quantization and motion estimation: They are initially derived from the picture QF, which is either set on the command line and used for all pictures or determined by means of the Rate Control algorithm. However, a number of factors are used to modify the lambda values after motion estimation. \[6\]

The initial assignation of lambda values is as follows: \[6\]

\[
\begin{align*}
I_{\text{lambda}} &= \frac{1}{16} \times 10^{\frac{(10 - QF)}{2.5}} \\
L1_{\text{lambda}} &= 32 \times I_{\text{lambda}} \\
L2_{\text{lambda}} &= 256 \times I_{\text{lambda}}
\end{align*}
\]  
\text{(C.1)}

These lambda variables are used for quantizer selection in I, L1 and L2 pictures. From these, motion estimation lambdas are derived. The ideal trade-offs may change with different sequences, video resolutions, perceptual weightings, or block sizes. \[6\]

\[
\begin{align*}
L1_{\text{me.lambda}} &= 2.0 \times \sqrt{L1_{\text{lambda}}} \\
L2_{\text{me.lambda}} &= L1_{\text{me.lambda}}
\end{align*}
\]  
\text{(C.2)}
The guiding principles for I, L1 and L2 pictures are as follows: [6]

1. I pictures should be of higher quality than L1 pictures and L1 pictures should be of higher quality than L2 pictures.

2. Motion data and good motion rendition is more significant at lower bit rates (low QFs) than at higher ones (high QFs).

The first principle arises because the I pictures are used as references for the L1 and the L2 pictures; L1 pictures are used as references for the L2 pictures. If the quality were to go up from I to L1 or from L1 to L2, then the encoder would need to correct the quantization error introduced in the reference picture and “pushed forward” by motion compensation. This error is noise-like and expensive to code. Also, an error in a single coefficient in the reference picture can spread to several coefficients when that picture is shifted through motion compensation. As a result, the L1 and the L2 lambdas multiply.

The aim of the second principle is to stop the quality from falling off a cliff since when QF goes down, lambdas go up. The motion field is not over-smoothed at low bit rates. Even if the quality is lower, there are no poorly corrected areas. L2 pictures have less opportunity to correct motion estimation errors in residual coding. [6]
REFERENCES


[19] “And now, Dirac from the Olympics, a new free codec!”:
http://www.videsignline.com/210601739


[22] “Video encoding and bandwidth requirements”:
http://www.cybertechmedia.com/videoencoding.html

http://www.ece.uwaterloo.ca/~z70wang/


[26] MPEG-4 Part 2, ISO/IEC 14496-2, International Organization for Standardization,
http://www.iso.ch


[30] Dirac video codec - A programmer's guide:
http://dirac.sourceforge.net/documentation/code/programmers_guide/toc.htm


[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

94, pp. 60-76, Jan. 2006

Oct. 2004

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

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


[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


in Wavelet Domain”, Applications of Digital Image Processing XXXII conference,


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


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


[56] MPEG-TS / MPEG transport stream:


[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.