Research and Developments of Dirac Video Codec

A thesis submitted for the degree of Doctor of Philosophy

By

Myo Tun

School of Engineering and Design, Brunel University

September 2008

This work is dedicated to my parents, brother and sister, and especially to my wife who had endlessly supported and encouraged me during this period of my three years study.

Acknowledgement

First of all, I would like to express my sincere gratitude to Dr. Jonathan Loo who was my advisor and principal investigator of the Dirac development project and Prof. John Cosmas who was my second advisor for their constant guidance, invaluable advice and suggestions throughout my study in Brunel University.

Secondly, I would like to gratefully acknowledge the financial support from the BBC R&D and Brunel University, UK. I would also like to thank Dr. Thomas Davies and Dr. Tim Borer from the BBC R&D for their insightful and constructive comments on this work.

And finally, I would like to thank Prof. Mohammed Ghanbari for kindly accepting as my Ph.D examiner. The book written by him, which is "Video Coding, an introduction to standard codecs" was my first hand book when I started learning the basic technology behind the video coding and had been very helpful for me. Without this book, it would have been really hard for a person like me to start building basics knowledge in video compression research area.

Abstract

In digital video compression, apart from storage, successful transmission of the compressed video data over the bandwidth limited erroneous channels is another important issue. To enable a video codec for broadcasting application, it is required to implement the corresponding coding tools (e.g. error-resilient coding, rate control etc.). They are normally non-normative parts of a video codec and hence their specifications are not defined in the standard. In Dirac as well, the original codec is optimized for storage purpose only and so, several non-normative part of the encoding tools are still required in order to be able to use in other types of application.

Being the "Research and Developments of the Dirac Video Codec" as the research title, phase 1 of the project is mainly focused on the error-resilient transmission over a noisy channel. The error-resilient coding method used here is a simple and low complex coding scheme which provides the error-resilient transmission of the compressed video bitstream of Dirac video encoder over the packet erasure wired network. The scheme combines source and channel coding approach where error-resilient source coding is achieved by data partitioning in the wavelet transformed domain and channel coding is achieved through the application of either Rate-Compatible Punctured Convolutional (RCPC) Code or Turbo Code (TC) using un-equal error protection between header plus MV and data. The scheme is designed mainly for the packet-erasure channel, i.e. targeted for the Internet broadcasting application.

But, for a bandwidth limited channel, it is still required to limit the amount of bits generated from the encoder depending on the available bandwidth in addition to the error-resilient coding. So, in the 2nd phase of the project, a rate control algorithm is presented. The algorithm is based upon the Quality Factor (QF) optimization method where QF of the encoded video is adaptively changing in order to achieve average bitrate which is constant over each Group of Picture (GOP). A relation between the bitrate, R and the QF, which is called Rate-QF (R-QF) model is derived in order to estimate the optimum QF of the current encoding frame for a given target bitrate, R.

In some applications like video conferencing, real-time encoding and decoding with minimum delay is crucial, but, the ability to do real-time encoding/decoding is largely determined by the complexity of the encoder/decoder. As we all know that motion estimation process inside the encoder is the most time consuming stage. So, reducing the complexity of the motion estimation stage will certainly give one step closer to the real-time application. So, as a partial contribution toward real-time application, in the final phase of the research, a fast Motion Estimation (ME) strategy is designed and implemented. It is the combination of modified adaptive search plus semi-hierarchical way of motion estimation. The same strategy was implemented in both Dirac and H.264 in order to investigate its performance on different codecs. Together with this fast ME strategy, a method which is called partial cost function calculation in order to further reduce down the computational load of the cost function calculation was presented. The calculation is based upon the pre-defined set of patterns

which were chosen in such a way that they have as much maximum coverage as possible over the whole block.

In summary, this research work has contributed to the error-resilient transmission of compressed bitstreams of Driac video encoder over a bandwidth limited error prone channel. In addition to this, the final phase of the research has partially contributed toward the real-time application of the Dirac video codec by implementing a fast motion estimation strategy together with partial cost function calculation idea.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AIP	Adaptive Irregular Pattern
AIPS	Adaptive Irregular Pattern Search
APDS	Adjustable Partial Distortion Search
ARP	Adaptive Rood Pattern
ARPS	Adaptive Rood Pattern Search
AVC	Advance Video Coding
BBC	British Broadcasting Corporation
BDM	Block Distortion Matrix
BMAs	Block Matching Algorithms
BMC	Block Motion Compensation
CAVLC	Context-Adaptive Variable-Length Coding
CBR	Constant Bitrate
CGI	Control Grid Interpolation
CIF	Common Intermediate Format
CPB	Coded Picture Buffer
CRC	Cyclic Redundancy Check
DCT	Discrete Cosine Transform
D-Q	Distortion-Quantization
DS	Diamond Search
DWT	Discrete Wavelet Transform
EHS	Extended Hexagon-Based Search
EPSZ	Enhanced Predictive Zonal Search
ExtEPZS	Extended EPZS
EZW	Embedded Zerotrees Wavelet
FEC	Forward Error Correction
FMO	Flexible Macroblock Ordering
FourSS	Four-Step Search
FS	Full Search
FS-BMA	Full Search Block Matching Algorithm
GOP	Group of Picture
HBMA	Hierarchical Block Matching Algorithm
HD	HiDefinition
HDTV	HiDefinition Television
HEXBS	Hexagon-Based Search
HRD	Hypothetical Reference Decoder
HVS	Human Visual System
IDR	Instantaneous Decoding Refresh
IDWT	Inverse Discrete Wavelet Transform
IPTV	Internet Protocol Television
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
JVT	Joint Video Team
LDSP	Large Diamond Search Pattern

LFP	Last Frame Prediction
LRFP	Last Reference Frame Prediction
MAD	Mean Absolute Difference
MAP	Maximum a Posteriori
MB	Macro Block
MBD	Minimum Block Distortion
MDSC	Multiple Description Scalable Coding
MDSQ-FGS	Multiple Description Scalar Quantization for Fine Granularity Scalability
ME	Motion Estimation
ML	Maximum-Likehood
MMX	Multimedia Extension
MP	Median Prediction
MPEG	Moving Pictures Expert Group
MRME	Multi-Resolution Motion Estimation
MSE	Mean Square Error
MV	Motion Vector
NPDS	Normalized Partial Distortion Search
NTSS	New Three-Step Search
OBMC	Overlapped Block-based Motion Compensation
PCCC	Parallel Concatenated Convolutional Codes
PSNR	Peak Signal to Noise Ratio
QCIF	Quarter Common Intermediate Format
QF	Quality Factor
QMEE	Quadrilateral Matching Motion Estimation
QP	Quantization Parameter
RCPC	Rate-Compatible Punctured Convolutional
R-D	Rate-Distortion
RDO	Rate-Distortion Optimization
ROI	Region Of Interest
R-Q	Rate-Quantization
R-QF	Rate-Quality Factor
RS	Reed-Solomon
RSC	Recursive Systematic Convolutional
SAD	Sum of the Absolute Difference
SD	Standard Definition
SDSP	Small Diamond Search Pattern
SMPTE	Society of Motion Picture and Television Engineers
SPIHT	Set Partitioning in Hierarchical Trees
SS	Spiral Search
SSIM	Structural SIMilarity
TC	Turbo Code
TD	Temporal Distance
TM5	Test Model Version 5
TMN8	Test Model Near-term Version 8
TSS	Three-Step Search
ULP	UpLayer Prediction

UMHexagonS	Hybrid Unsymmetrical-Cross Multi-Hexagon-Grid Search
UP	Un-Partitioned
VA	Viterbi Algorithm
VCEG	Video Coding Experts Group
VHS	Video Home System
VLC	Variable Length Coding
VM8	Verification Model Version 8
VQEG	Video Quality Experts Group
VQM	Video Quality Metrics
VZT	Virtual Zero-Tree
WME	Warping Motion Estimation
ZTE	Zerotree Entropy
λ	Lagrangian Parameter/Multiplier
Δ	Quantization Factor

List of Publication

Journals

[1] M. Tun, K. K. Loo and J. Cosmas, "Error-Resilient Performance of Dirac Video Codec over Packet-Erasure Channel," *IEEE Transactions on Broadcasting*, Vol. 53, Issue 3, pp. 649-659, Sept. 2007.

[2] M. Tun, K. K. Loo and J. Cosmas, "Semi-Hierarchical Based Motion Estimation Algorithm for the Dirac Video Encoder," *WSEAS Transactions on Signal Processing*, Vol. 4, Issue. 5, pp. 261-270, May 2008.

[3] M. Tun, K. K. Loo and J. Cosmas, "Enabling Error-Resilient Internet Broadcasting using Motion Compensated Spatial Partitioning and Packet FEC for the Dirac Video Codec," *Journal of Multimedia (JMM)*, Vol. 3, No. 2, pp. 1-11, June 2008.

[4] M. Tun, K. K. Loo and J. Cosmas, "Rate Control Algorithm Based on Quality Factor Optimization for DIRAC Video Codec," *Elsevier Journal of Signal Processing: Image Communication*, Vol. 23, Issue 9, pp. 649-664, Oct. 2008.

[5] M. Tun, K. K. Loo and J. Cosmas, "An Efficient Rate Control Algorithm for Wavelet Video Codec," *International Journal of Wavelets, Multiresolution and Information Processing (IJWMIP)*, Accepted for future publication.

Conferences

[1] Myo Tun, K. K. Loo and W. A. C. Fernando, "An Error-Resilient Algorithm Based on Partitioning of the Wavelet Transform Coefficients for a DIRAC Video Codec," *International Conference on Wireless Personal Multimedia Communications, WPMC'06*, 17-20 Sept. 2006, San Diego, CA, USA.

[2] M. Tun, K. K. Loo and J. Cosmas, "Preliminary Results on the Application of Error-Resilient Scheme on Dirac Video Codec," *The 3rd European Conference on Visual Media Production (CVMP 2006)*, pp 48–54, 29-30 Nov. 2006, London, UK.

[3] M. Tun, K. K. Loo and J. Cosmas, "A Novel Rate Control Algorithm for the Dirac Video Codec Based upon the Quality Factor Optimization," *IEEE Conference Proceedings on Geometric Modeling and Imaging, GMAI '07*, pp. 14-18, 4-6 July 2007, Zurich, Switzerland.

[4] M. Tun, K. K. Loo and J. Cosmas, "Semi-Hierarchical Motion Estimation for Dirac Video Codec," *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, 31 March - 2 April 2008, Las Vegas, Nevada, USA.

[5] M. Tun, K. K. Loo and J. Cosmas, "Motion Estimation using Partial Cost Function Calculation," *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, 31 March - 2 April 2008, Las Vegas, Nevada, USA.

[6] M. Tun, K. K. Loo and J. Cosmas, "Fast Motion Estimation using Semi-Hierarchical Approach for the Dirac Video Codec," 8th *WSEAS Int. Conf. on Multimedia Systems and Signal Processing (MUSP '08)*, pp 273-279, April 6-8, 2008, Hangzhou, China.

Chapter 1

1 Introduction

Now a day, analogue video recording is a mature technology and has almost reached its limits. Investments in enhancing the technology provide increasingly small returns. On the other hand, digital video technology has the potential to achieve much higher levels of quality and the technology is being improved at an increasing rate. It has a number of unique properties that make possible applications that could not be realised using analogue video. Firstly, digital video can be manipulated more easily than analogue video. In addition to this, digital video can be stored on random access media, whereas analogue video is generally stored sequentially on magnetic tape. This random access allows for interactivity, since individual video frames are addressable and can be accessed quickly. Digital video can be duplicated without loss of quality which is important for editing applications.

The ability to easily store and transmit is by far its most important property. Video in digital form can be transmitted across channels where transmission of analogue video was almost impossible. Because compressed digital video can be transmitted using less bandwidth than analogue television, it is possible to provide many channels where before there were only a few or none. By exploiting the digital technology, cable TV systems can have enough capacity to provide hundreds of channels of digital video. Video-on-demand is currently available on trial basis. The video was delivered to the consumers homes via their copper telephone wire and normal telephone service was not disrupted. In future, video-on-demand services might eventually replace the trip to the video store.

In addition to the applications mentioned above, modern digital video applications also include storage on different media such as Video-CD, DVD, Blu-Ray Disc, broadcasting over wireless mobile channel, streaming over the internet, satellite and terrestrial digital TV, video-conferencing and video-telephony and many more. Widely development of the digital video applications has led the generation of the international standards for different types of applications under the auspices of the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) and the International Organization for Standardization / International Electrotechnical Commission (ISO/IEC). The ITU-T's H.261 video coding standard is originally designed for transmission over ISDN lines on which data rates are multiples of 64 Kbit/s and H.263 is designed as a low-bitrate compressed format for video conferencing. On the other hand, the Moving Pictures Expert Group (MPEG), established under the ISO/IEC, standardized MPEG-1 to compress Video Home System (VHS) quality raw digital video. Under the same group, another standard called MPEG-2 is widely used as the format of digital television signals that are broadcast by terrestrial (over-the-air), cable, and direct broadcast satellite TV systems. It also specifies the format of movies and other programs that are distributed on DVD. MPEG-4 is for compression of Audio Video data for web (streaming media) and CD distribution, voice (telephone, videophone) and broadcast television applications. It

provides improved coding efficiency, ability to encode mixed media data (video, audio, data) and error-resilience to enable robust transmission.

The latest standard, H.264 which is also called MPEG-4 Part 10 was developed by the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG) as the product of a partnership effort known as the Joint Video Team (JVT). It is aimed to elaborate an open standard that is not application-specific and that perform significantly better than the existing standards in terms of compression, network adaptation and error robustness.

1.1 Background and Motivation

In the near future, H.264 will gain wide acceptance on many applications especially on Internet broadcasting. However, the usage of H.264 incurs royalty fees which may not be cost effective for non-profit and public content owners such as public service broadcasters, archive institutes, etc., for deployment of Internet-based services. Whilst these costs are manageable initially, these could become prohibitive if the services scaling up to millions of users, or if new services are deployed which were not envisaged in the original license agreements.

As an alternative, a royalty-free general-purpose video codec called Dirac [1] is designed, which is aimed at a wide range of applications from storage of video content to streaming video in view to address the above demands. Being "open technology", Dirac is an attractive option as it allows content owners to distribute contents without royalty-fees in anyway. However, the Dirac's current release is still in alpha development stage and has been optimized for storage purposes only. Currently, there is no error-resilient encoding mechanism and rate control facility, which are the essential tools for broadcasting over the bandwidth limited error prone channels. Being able to perform real-time encoding/decoding is another important feature for a video codec in order to be able to use in video conferencing application where encoding/decoding the live data in real-time with minimum delay is required. So, it is required to develop the several features mentioned above so that the codec can be used in real-time broadcasting where faster encoding solution, error-resiliency and controlled output bitrate are normally required.

1.2 Research Challenges

Although many of the concepts are similar to existing standardized video compression systems, Dirac works differently in that it uses wavelet instead of Discrete Cosine Transform (DCT), which is capable of offering coding at resolutions scalable from QCIF to HD if properly exploited. Even though naturally scalable characteristic of the wavelet transform is good for scalability coding, this feature is not helpful for other type of mechanisms especially in error-resilient coding. Unlike block based transform in DCT approach, wavelet transform is applied on the whole frame basis resulting series of subbands. The visual quality of impairments that Dirac will experience will be very different from those experienced by block based codecs since the whole wavelet subband may be received in

error, so artefacts may not be localized. This will seriously affect the error concealment capability since the localized error is normally desirable in most of the error concealment coding.

Another factor is that Dirac's intention is to stay free of any patented algorithm, thus remaining a royalty free codec. So, in designing the required functional module, it is really important to avoid any patented algorithms. Moreover, the philosophy behind the Dirac codec is 'keep it simple' even though the codec is to be competitive with the other state-of-the-art standard video codecs. Sophisticated algorithms and complex mathematical calculation are to be avoided. The main concept is to design simple, efficient and patent free algorithm which can be applied on a wavelet based hybrid video encoder and hence it is going to be a great challenge for a researcher to design an algorithm which fulfils all these requirements.

1.3 Aim and Objectives

The main goal of this research is to develop the Dirac video encoder which is currently optimized for storage purpose only, in order to be able to use in real-time broadcasting over bandwidth limited erroneous channels. A series of projects had been conducted to achieve this aim. To enable the encoder to be used in broadcasting, error-resilient coding scheme and rate control algorithm have been developed. These are the essential tools for the robust transmission of the sensitive compressed video data over the error prone channel and controlling of generated output bitrate over the bandwidth limited channel. The latter is especially useful in mobile wireless channel where radio frequency spectrum is normally scarce resources.

The real-time transmission is another issue, where both encoder and decoder have to be synchronized in order to be able to perform in real-time encoding and decoding without having any noticeable delay. One of the intentions of Dirac is to achieve real time decoding at standard definition (e.g. 720 ×576 pixels at 25 frames/s). At present, decoder approaches real time for CIF pictures (352 × 288) at 12 frames/s. The development team is still working hard in order to achieve their goal of real-time decoding. But, there is still a big issue as far as the encoding side is concerned. Application of multi level fully hierarchical motion estimation at the encoder together with 61 points diamond and 9 points square search patterns makes the encoder hard to use in real-time application. It was found that up to 80% of the total encoding time is spent in motion estimation stage only. So, another objective of this research is to implement fast and accurate motion estimation strategy which can really reduce down the number of search in the pixel accuracy stage of the motion estimation in Dirac. But, there are still a number of algorithms to be optimized in order to achieve real-time encoding including Rate-Distortion Optimization (RDO) mode decision and quantizer selection for each subbands. Unfortunately, these issues cannot be addressed in this research because of the limited time frame which is only three years for all of these projects. But, the implementation of the fast motion estimation strategy can help the encoder one way or another to be able to use in real-time application in the future optimized version of Dirac.

1.4 Main Contributions

In the beginning phase of the research, a simple and low complexity error-resilient coding scheme which provides the error-resilient transmission of the compressed video bitstream of Dirac video encoder over the packet erasure wired network, was presented. The scheme is based on the combinations of source and channel coding approach and designed mainly for the packet-erasure channel, i.e. targeted for the Internet broadcasting application. But it is also possible to extend the scheme in order to suit in other type of channels (e.g. wireless mobile channels).

In the middle phase of the research, the rate control algorithm which is efficient, simple and easy to integrate to the Dirac encoder was presented. The algorithm is based upon the *QF* optimization method where *QF* of the encoded video is adaptively changing in order to achieve average bitrate which is constant over each GOP. A relation between the bitrate, *R* and the *QF*, which is called R-QF model is derived in order to estimate the optimum *QF* of the current encoding frame for a given target bitrate, *R*.

In the final phase of the research, a fast ME strategy which is a combination of modified adaptive search plus semi-hierarchical way of motion estimation was presented. It was implemented in both Dirac and H.264 in order to investigate its performance on both codecs. Together with this fast ME strategy, a method which is called partial cost function calculation in order to reduce down the computational load of the cost function calculation was proposed. The calculation is based upon the pre-defined set of patterns which were chosen in such a way that they have as much maximum coverage as possible over the whole block.

1.5 Organization of Thesis

The thesis is composed of six chapters where the first chapter is introduction followed by the detail description of Dirac video encoder's architecture in chapter 2. The main contributions of the research which are error-resilient coding scheme, rate control algorithm and fast motion estimation strategy are in chapter 3, 4 and 5, respectively followed by conclusion and future recommendation in chapter 6. Reference list is presented in the last part of the thesis.

Chapter 1: Highlights some important applications of the digital video and the role of the compression techniques together with the development of the international standards for different type of applications. In addition to this, background and motivation, research challenges, aims and objectives, main contribution of the research were clearly presented. Finally, the organization of the thesis is stated where the overall description of each chapter was included.

Chapter 2: Details the architecture of the Dirac encoder starting from development timeline. After that, over all encoding structure is discussed followed by the detail description of motion estimation, Overlapped Block-based Motion Compensation (OBMC), Discrete Wavelet Transform (DWT), RDO quantization and entropy coding. Finally, block diagram of Dirac's bitstream syntax is presented in appendix A.

Chapter 3: Presents the error-resilient encoding scheme where the chapter begins with the introduction followed by related works and objective of the research. After that the presented scheme is discussed in details followed by result and discussion and chapter summary. In the last part of the chapter, the block diagram of the modified bitstream syntax of the Dirac with error-resilient coding and different spatial partitioning formats are presented in appendix C and D, respectively.

Chapter 4: Presents the rate control algorithm where, as in chapter 3, the chapter begins with the introduction followed by related works and objective of the research. After that the current rate control algorithm of H.264 is presented. And then, the rate control algorithm for Dirac is discussed in detail followed by result and discussion and chapter summary.

Chapter 5: Presents a fast motion estimation strategy where like before, the chapter begins with the introduction followed by related works and objective of the research. Some of the most famous fast Block Matching Algorithms (BMAs) are presented followed by the detail discussion of proposed fast motion estimation strategy. And then, existing motion estimation strategies of H.264 are presented. In addition to the motion estimation strategy, partial cost function calculation is also discussed in detail followed by result and discussion and chapter summary.

Chapter 6: is the conclusion of the thesis where overall achievements of the research and further recommendations of each phase of the research are presented.

Chapter 2

2 Dirac Video Codec

2.1 Dirac Development Timeline

Over the past few years, BBC has developed an advanced video compression system, called "Dirac", [1] which is comparable with the latest standards, H.264/MPEG-4 AVC and VC-1. Potential uses of this codec (coder/decoder) include Internet distribution such as web clips, video on demand and IPTV. With industry plans for "On Demand" TV and streaming over the Internet, open platforms technologies like Dirac have become more significant. The way Dirac has been developed allows it to be used on any platform and without the payment of royalties.

Dirac has continued to mature with improved performance both in terms of compression and implementation. One of the intentions is to achieve real time decoding at standard definition (e.g. 720 ×576 pixels at 25 frames/s). At present, decoder approaches real time for CIF pictures (352 × 288) at 12 frames/s. Table 2-1 shows the development timeline of Dirac for some significant modification.

Date	Development
11 March 2004	Initial Release of Dirac (Dirac 0.1.0)
10 May 2004	Release of Dirac 0.2.0
12 May 2004	Added support for frame padding so that arbitrary block sizes and frame dimensions can be supported.
18 May 2004	Added support for I-frame only coding by setting *num_L1* equal 0; *num_L1* negative gives a single initial I-frame ('infinitely' many L_1 frames)
19 May 2004	Replaced zero-padding with edge-padding to eliminate colour fringing at low bitrates.
27 May 2004	Release of Dirac 0.3.0
08 June 2004	Release of Dirac 0.3.1
06 July 2004	Added *QualityMonitor* class to do constant-quality encoding. Class looks at difference between locally decoded and original frames, and adjusts Lagrangian parameters or λ appropriately.
11 Aug. 2004	Added support for cut-detection and intra frame insertion.
18 Aug. 2004	Release of Dirac 0.4.0
25 Aug. 2004	Release of Dirac 0.4.1
09 Sept. 2004	Release of Dirac 0.4.2
17 Sept. 2004	Changed quality metric from PSNR to one based on 4^{th} powers of errors, to give bigger weighting to large errors.
17 Sept. 2004	Changed structure to use a map for the different λ called λ map, which is encapsulated in the *MEData* structure. Limited size of MV costs to allow encoder to cope with motion transitions better.
20 Sept. 2004	Release of Dirac 0.4.3

04 Nov. 2005	Including arrays to hold global motion data in *MvData* and *MEData.*
01 Dec. 2004	Release of Dirac 0.5.0
02 Feb. 2005	Modified wavelet coefficient coding to code blocks of coefficients with skip flags if all coefficients are zero.
02 Feb. 2005	The arithmetic coding and decoding engines have been modified to improve speed and code organization.
02 Feb. 2005	Added classes to handle selecting of quantizers. These allow a single quantizer per subband to be chosen, or alternatively one for each code block.
17 Feb. 2005	Release of Dirac 0.5.1
17 March 2005	Initial implementation of Global Motion. Allows for Global Motion Only switch for each frame. If this is set, only the Global Motion Parameters are coded - i.e. no block MVs are coded.
06 April 2005	Speed-up to arithmetic coding and decoding, using probability range re-normalization to avoid unnecessary divides.
04 May 2005	Added three new filter types with lifting implementations, all much faster than Daubechies (9,7).
19 May 2005	Added support for multiple levels of MV precision - pixel, half pixel (1/2 pixel), quarter pixel (1/4 pixel) and eighth pixel (1/8 pixel).
25 May 2005	Release of Dirac 0.5.2
10 June 2005	Changed default block height and width for motion compensation to 24 by 24 pixels from 20 by 20. Separation remains 16 by 16. The result is reduced blockiness in areas of poor motion prediction. Downside is slower encoder, but the block parameters are merely scaled-up versions of the SD parameters.
26 July 2005	Reduced chroma weighting factors in order to increase chroma fidelity - chroma bit rates should increase from about 5% of total for 420 chroma format to 7.5% of total.
10 Aug. 2005	Changed default wavelet filters to fast filters, APPROX97 and FIVETHREE.
10 Aug. 2005	Removed constant quality encoding control mechanism, because it worked too poorly for varied sequences. Instead, *QualityMonitor* just monitors quality and encoder control reverts to constant Lagrangian parameters, λ.
10 Aug. 2005	Optimized 5_3 wavelet synthesis function, using MMX instructions resulting in 6 percent speed improvement in decoding SD.
23 Aug. 2005	Release of Dirac 0.5.3
31 Aug. 2005	Added support for lossless I-frame and long-GOP coding.
27 Sept. 2005	Inverse wavelet transform is now done for L_2 frames only if local decoding is required. This is because L_2 frames are (by definition) not used for reference and can be left in a partially reconstructed state if local decoding is not required.
27 Sept. 2005	Re-organized classes for doing SAD calculations at pixel and sub-pixel accuracy. These are present also in 'bailout' form so that the algorithm can leave the calculation as soon as it knows that the current SAD is not going to beat the best match so far. These modifications support speeding up motion estimation, and hence the encoder generally.
27 Sept. 2005	Search ranges restricted to improve motion estimation speed.
08 Nov. 2005	Modified motion compensation so that it takes MB splitting mode into consideration when compensating a row of blocks. Depending on the MB split mode, 1, 2 or 4 blocks of reference data are used to calculate the motion compensated data. This speed up the default quarter pixel block motion compensation routine

	by an average of 20-23% for a 2Mbps Dirac bitstream.
08 Nov. 2005	Optimized *WaveletTransform::VHFilter5_3::Split* using MMX instructions. Minor modification to *WaveletTransform::VHFilter5_3::Synth* function improve speed slightly.
05 Dec. 2005	Release of Dirac 0.5.4
10 Feb. 2006	Modified calculation of quantization, inverse quantization and offset values so as to conform to specification. (This modifies the bitstream.)
15 Feb. 2006	Currently a frame is padded so that it has an integer number of whole MBs and is also a multiple of $2^{(\text{wavelet transform depth})}$. This has changed in the Dirac specification where a frame is padded so that its dimensions are a multiple of $2^{(\text{wavelet transform depth})}$ only. Also the luma and the chroma components can be padded differently. e.g. the luma dimensions may be a perfect multiple of $2^{(\text{wavelet transform depth})}$ so the luma component is not padded but the chroma component may need to be padded depending on the chroma format of the input. To take care of this, *FrameParams* class has been modified to accept both padded chroma and padded luma dimensions. (This modifies the bitstream.)
15 Feb. 2006	A linear function is used to calculate the OBMC weights instead of a raised cosine function to make it compliant with spec. Also motion compensation is performed only on true picture dimensions and not padded picture dimensions. (This modifies the bitstream.)
21 Feb 2006	Modified arithmetic coding engine for speed and spec conformance. 1. Changed context statistics so that maximum weight is 256 2. Changed look-up table so that inverse of weight is calculated to 16 instead of 31 bits. 3. Changed scaled count of zero *m_prob0* so that it's 16 bits instead of 10
18 April 2006	Changing binarisation for arithmetic coding, and associated contexts. Binarisation for the magnitude values of wavelet coefficients, MV prediction residues and DC values has been changed from unary to interleaved exp-Golomb so that the number of symbols to be decoded in order to reconstruct a coefficient is reduced. Exp-Golomb binarisation takes a number $N \geq 0$ and codes it via the binary representation of $N+1$, $1bbbbbb$. If there are K bits following the leading 1, the representation is K zeroes followed by the binary representation: 00...01*bbbbb* Interleaved exp-Golomb is the same, except that the K bits are interleaved with the zeroes: 0*b*0b 0*b*1 so that a single decoding loop can be used. The zeroes here act as "follow bits" indicating that another bit is to be sent, with 1 as the terminator.
15 May 2006	Changed quantization to give 2 bits of accuracy to quantization factors. This improves performance in high-quality applications, and reduces large steps in quality.
16 May 2006	Max number of quantizers increased to 97 (0..96).
16 May 2006	Computes PSNR instead of weighted 4^{th} power metric; chroma PSNRs are also computed
05 June 2006	Add support for Digital Cinema video formats.
13 June 2006	Release of Dirac 0.6.0
29 June 2006	Defined new Wavelet Filter class *VHFilterHaar* to include HAAR filter support in Dirac.
29 Sept. 2006	Non-overlapped blocks are now allowed and supported. The raised cosine macro has now been removed and only linear weights are supported.
09 Nov. 2006	- Changed Layer 1 frames with *P* frames and Layer 2 with *B* frames, rather than with Inter Ref and Inter Non-ref respectively. This is more efficient with the new GOP structure - Modified RDO framework to provide correction where there has been ME failure, i.e. lots of Intra blocks - Slightly increased ME search areas

	- Corrected the frame type parameter for the final *B* frame in a sequence
13 Nov. 2006	Changed quantizer offsets to be different for Intra and Inter frames, as per the latest draft of the spec. Having an offset of 0.5×quantizer for intra frames improves performance at high rate, especially iterated coding with Dirac Pro application.
22 Nov. 2006	Inserted intra frames are now given lower quality than other intra frames so as to match *P* and *B* frames more closely.
11 Jan. 2007	Changed spec. so that the size of the reference picture buffer is determined from the level and profile information rather than fixed as 5 pictures.
21 March 2007	Added support for Constant Bitrate (CBR) encoding. This equalises bitrate over a GOP (*I* frame to *I* frame). It does not enforce bit rate, nor does it operate according to a buffer model. To apply CBR coding, add -targetrate *N*, where *N* is the target bit rate in Kb/s. Precedence: 1) if a *QF* is also set with *-qf Q*, then the value of *Q* is used as the initial value for the system, but CBR is still applied. 2) if *-lossless* is also set, then lossless coding is applied and CBR constraints are ignored.
27 March 2007	Major re-factor of rate control algorithm. The algorithm has been revised so that a target buffer occupancy is aimed for, with the target bit rate set as equal to the mean bit rate plus an adjustment factor to steer back to target. A buffer size of 4×bit rate has been selected.
04 April 2007	Updated quantisation factors so that they represent an integer approximation of $2^{((q/4)+2)}$ up to q=128.
11 April 2007	MV data is now split into a number of different parts which are coded as separate units: - Superblock/MB splitting mode - Block prediction mode - MVs, reference 1, horizontal component - MVs, reference 1, vertical component - MVs, reference 2, horizontal component - MVs, reference 2, vertical component - DC values, Y - DC values, U - DC values, V Each is a separate, byte-aligned element with a prefixing length code. This allows parallel encoding and decoding of these elements, especially beneficial in hardware.
09 May 2007	Release of Dirac 0.7.0
15 May 2007	Modified constant bit-rate operation, so that when a cut is detected, the long-term *QF* is used, instead of the current one in the rate model. This has the effect of reducing quality crashes and rate explosions.
20 June 2007	Added supports for full-search block matching. This is controlled by using the flag *-full_search* [*xr*] [*yr*] to do an initial pixel-accurate search in the range [*-xr*, *xr*]×[*-yr*, *yr*]. Sub-pixel refinement is unaffected.
26 Sept. 2007	Added support for interlaced coding. Changes include refactoring of *PicIO* classes, *SequenceCompressor* class to handle interlaced coding. Modified GOP structure for interlaced coding to code interlaced material more efficiently.
02 Oct. 2007	Release of Dirac 0.8.0
16 Nov. 2007	Added support for using VLC codes for entropy coding of coefficient data.

23 Nov. 2007	Changed up conversion filter (used in subpixel refined ME) to 8 taps with 6 signed bits per tap by approximating the original 10 tap, 9 bit filter. The intermediate calculations will now fit into 16 bits even for 10 bit input data. The new filter causes compression to be up to 0.1dB worse, in return for much speedier performance when optimized.
21 Dec. 2007	VC2 is now submitted to SMPTE (Dirac Pro)
23 Jan. 2008	Release of Dirac 0.9.0
26 Jan. 2008	Release of Dirac 0.9.1
22 Feb. 2008	Release of Schroedinger 1.0.0 [2] Schroedinger has a much more flexible architecture and should be able to get much better compression in the long-term than with the Dirac codebase, which should eventually die.
30 April 2008	Added support for various forms of pre-filtering. Pre-filtering using rectangular, diagonal or centre-weighted median filtering is now available as a command-line option.
04 June 2008	Release of Dirac 0.10.0. All the Dirac old releases can be downloaded from here [3].

Table 2-1 Dirac Development Timeline

As part of the commitment to standardization, BBC has initiated proceedings with the Society of Motion Picture and Television Engineers (SMPTE) to formulate Dirac Pro's specification under the name VC-2. Dirac Pro is an extension to the Dirac family of video compression tools optimized for professional production and archiving applications. It is designed for simplicity, efficiency, speeds and intended for high quality applications with lower compression ratios including transportation of signals between studios and production areas. In addition, it can be utilized for desktop production over IP networks, file storage and video editing. It provides wide range of bit rates from below 100Kbps to more than 1 Gbps but it is most suitable above 100 Mbps. In terms of architecture, Dirac Pro employs only I-Frame coding which means that each frame is coded independently, making editing and production easier and uses exp-Golomb coding which makes hardware and software implementation simple, efficient and low cost. Initial application of Dirac Pro (defined as Dirac Pro 1.5) was for use in transporting full HDTV (1080P50/60) giving 2:1 compression ratio with almost no loss in quality allowing full HDTV video to be carried on the same cables and infra structure used for conventional HDTV. A second application of Dirac Pro (defined as Dirac Pro 270) is to allow the transmission of HDTV signals using the cables and infrastructure formerly used for standard definition television. This requires more compression of HDTV signals than Dirac Pro 1.5 but Dirac Pro is sufficiently flexible to allow this to be achieved with little loss in quality. Detail information of the Dirac encoder's architecture and its corresponding functional blocks will be explained in the following sections. Most of the materials for these sections are taken from [4].

2.2 Overall Encoder Architecture

Dirac is a general-purpose video codec. It is aimed at a wide range of applications, offering efficient coding at resolutions from QCIF (176×144) to HDTV (1920×1080). It is based on wavelet

technology which is different from that used in the main standard video compression systems including H.264. It aims to be competitive with the other state-of-the-art standard video codecs and its performance is very much better than MPEG-2 and slightly less than H.264 even in the Alpha development stage. However, the performance was not the only factor driving its design. The philosophy behind the Dirac codec is 'keep it simple'. This is an ambitious aim since video codecs, particularly those with state-of-the-art performance, tend to be fearsomely complex. Dirac offers facilities for both interlaced and progressive scan sources and common chroma formats (luma only, 4:4:4, 4:2:2, 4:2:0) by means of frame padding. The frame padding allows variable Macro Block (MB) size to be used for motion estimation and ensures that the wavelet transform to be made on irregular frame dimensions.

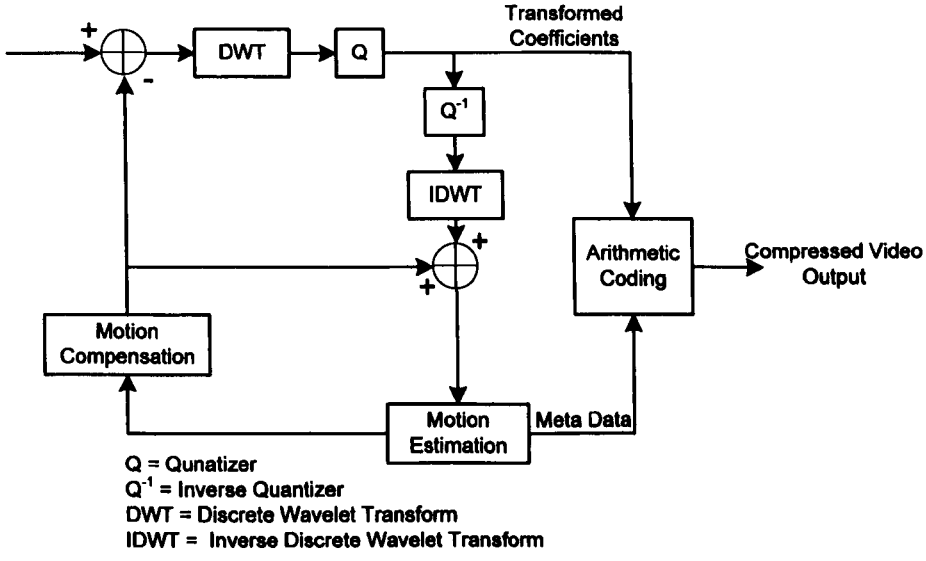


Fig. 2-1 Overall Hybrid Encoder Architecture [4]

Fig. 2-1 shows the structure of Dirac encoding architecture. The Dirac's design is that of a conventional hybrid motion-compensated architecture which based around fundamental coding algorithms. It uses hierarchical motion estimation and OBMC to avoid block-edge artifacts. First the motion compensated residual frames are wavelet-transformed using separable wavelet filters and divided into subbands. Then, they are quantized using RDO quantizers. Finally, the quantized data is entropy coded using an Arithmetic encoder. The following sections will give the detail explanation of each functional block in Fig. 2-1.

Dirac defines three frame types. Intra frames (*I* frames) are coded independently without reference to other frames in the sequence. Level 1 frames (L_1 frames) and Level 2 frames (L_2 frames) are both inter frames, which are coded with reference to other previously (and/or future) coded frames. The definition of the L_1 and L_2 frames are the same with *P* and *B* frames in H.264. The encoder operates with standard GOP modes whereby the number of L_1 frames between *I* frames, and the separation between L_1 frames, can be specified depending on the application. It can also be operated on *I* frame-only mode where all the frames are intra coded like in motion JPEG2000. A prediction method for

frame coding using a standard GOP structure is shown in Fig. 2-2. In this figure, the number of L_1 frames between *I* frames is 2 and the L_1 frame separation is 3.

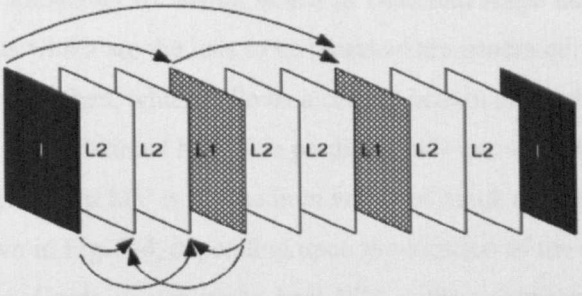


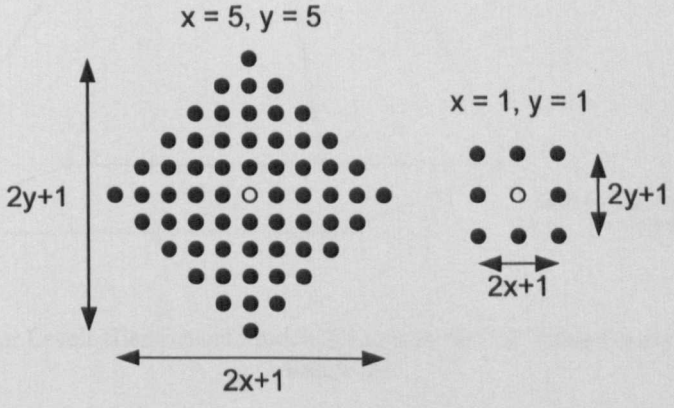
Fig. 2-2 Prediction of L_1 and L_2 frame [4]

2.3 Motion Estimation

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. The number of down conversion levels depends upon the frame format (i.e. *width* and *height* or dimension of the frame) and can be calculated using equation (2.1) as follow.

$$level = \left\lfloor \min\left(\log_2\left(\frac{width}{12}\right), \log_2\left(\frac{height}{12}\right)\right)\right\rfloor \tag{2.1}$$

According to equation (2.1), the number of down conversion levels can be 4 or 6 depending upon the video formats whether it is CIF or HD (1920×1080). In the down conversion process, the dimension (both height and width) of the frames are reduced by the factor of two in each level. 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.



(a) Diamond Search Pattern, Search Range = 5

(b) Square Search Pattern, Search Range = 1

Fig. 2-3 Search Patterns of Dirac [3]

The search pattern used in the lowest level is Diamond shape with the search range 5 and all other levels except lowest use square shape search pattern with search range 1. Fig. 2-3 shows both search patterns where there are altogether 61 search points in Diamond shape and 9 points in square shape. First of all, candidate lists which are the lists to be searched are generated. A candidate list consists of a number of points to be searched, which follows a certain pattern either diamond or square as shown in Fig. 2-3 and centered at a predicted MV. The predicted MV can be either zero, spatially predicted or guide MV. Spatially predicted MV is the medium vector of block number 1, 2 and 3 or mean vector of block 1 and 2 as shown in Fig. 2-4, depending upon the location of the current block where motion estimation is carried out. Guide vector is the best MV at the corresponding block location of the adjacent lower hierarchical level and it is not available for the lowest level.

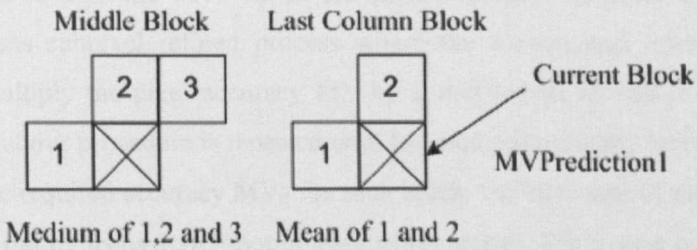


Fig. 2-4 Spatially Predicted MV of Dirac, the Current Block is the Block where ME is being Performed [3]

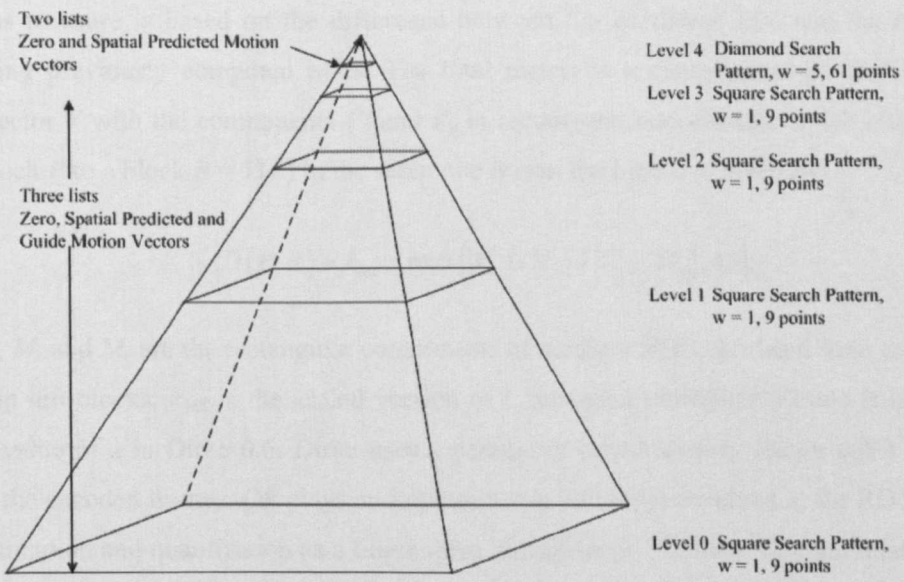


Fig. 2-5 Dirac's Four Levels Hierarchical Motion Estimation for CIF Video Format, where *w* is Search Range [3]

In Fig. 2-5, for lowest level search, two candidate lists are generated centered at zero MV and spatially predicted MV, respectively with the Diamond Search (DS) pattern. Sum of the Absolute Difference (SAD) is used here as the cost function. At the initial search step, the SAD calculation is carried out only for the center point of diamond pattern in each list and finds the list which gives the minimum cost. The candidate lists to be searched are chosen by multiplying the minimum cost with

1.5 and choose all the lists which give the cost less than 1.5 times minimum costs. So, there can be at most two candidate lists and 122 search points can be involved in lowest level search if there is no overlapping between the two lists. In the refined step, SAD calculation is carried out for all chosen candidate lists on their corresponding search points and the coordinate of the point which gives the minimum cost, is recorded as the best MV.

The search procedure is basically the same for all other levels except the addition of one more candidate list which is centered at the guide vector. So, there are three candidate lists in these levels with the square search pattern as shown in Fig. 2-3 and the maximum number of search points can be at most 27 in each level if there is no overlapping between the lists.

After going through all these levels, the pixel accuracy MVs for each block are obtained. Dirac provides the option to find the MVs up to 1/8 pixel accuracy. In order to achieve this, motion estimation undergoes subpixel refined process where the current and references pictures are up converted by 2, multiply the pixel accuracy MV by 2 and search around in order to get 1/2 pixel accuracy MV. The above procedure is repeated until the required accuracy level is reached.

After getting the required accuracy MVs for each block, the last stage of motion estimation, mode decision is carried out by using RDO motion estimation metric. The metric consists of a basic block matching metric which is SAD plus some constant times a measure of the local MV smoothness. The smoothness measure is based on the difference between the candidate MV and the median of the neighbouring previously computed MVs. The total metric is a combination of these two metrics. Given a vector V with the components V_x and V_y in rectangular coordinates, which maps the current frame's block P to a block $R = V(P)$ in the reference frame, the metric is given by:

$$SAD(P,R) + \lambda_{ME} \times \max\left(|V_x - M_x| + |V_y - M_y|, 48\right). \tag{2.2}$$

Where, M_x and M_y are the rectangular components of medium MV calculated from the MVs of left, top and top left blocks. λ_{ME} is the scaled version of Lagrangian multiplier (λ) and it is equal to two times the value of λ in Dirac 0.6. Dirac uses a parameter called Quality Factor (QF) to control the quality of the encoded frames. QF plays an important role since it is involved in the RDO processes of motion estimation and quantization as a Lagrangian multiplier (λ). In Dirac 0.6, the relation between λ and QF is defined as shown in equation (2.3).

$$\lambda = \left(10^{(10-QF)/2.5}\right)/16 \tag{2.3}$$

In mode decision, Dirac encoder considers the total of 12 modes which includes the combination of 3 MB splitting levels as shown in Fig. 2-6 and 4 prediction modes. A MB consists of a 4×4 array of blocks, and there are three possible ways of splitting a MB:

- Splitting level 0: no split, a single MV per reference frame for the MB;

- Splitting level 1: split into four sub-macroblocks (sub-MBs), each a 2×2 array of blocks, one MV per reference frame per sub-MB;
- Splitting level 2: split into the 16 constituent blocks.

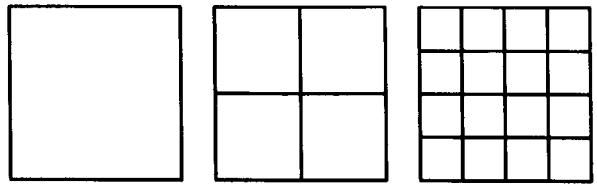


Fig. 2-6 MB Splitting Levels [4]

At the same time, the best prediction mode for each prediction unit (block, sub-MB or MB) is chosen. There are four prediction modes available:

- INTRA: intra coded, predicted by DC value;
- REF1_ONLY: only predict from the first reference;
- REF2_ONLY: only predict from the second reference (if one exists);
- REF1AND2: bi-directional prediction.

So, depending upon the MB splitting level and the prediction mode, there can be altogether 12 combination of mode which is decided by using the equation (2.2). Basically, mode decision process calculates the total cost using equation (2.2) for every combination of MB splitting level and prediction mode. And then, the best combination which yields the minimum cost is chosen as the best mode.

2.4 Overlapped Based-based Motion Compensation (OBMC)

A large weakness in traditional Block Motion Compensation (BMC) used in many MPEG architectures is unwanted block-edge artifacts located around the block boundaries. For this reason, Dirac uses OBMC in order to avoid the block-edge artifacts which are sensitive to wavelet transforms and expensive to be coded. Dirac encoder can deal with any degree of overlapping with variable block sizes and this is configurable at the encoder. One issue is that there should be an exact number of MB horizontally and vertically; otherwise it can also be achieved by padding the data.

Dirac's OBMC scheme is based on a separable Raised-Cosine 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 if it lies at the corner of a block as shown in Fig. 2-7. The figure shows the overlapped block structure of a MB which includes 16 blocks with 12 pixel block length and 8 pixel block separation.

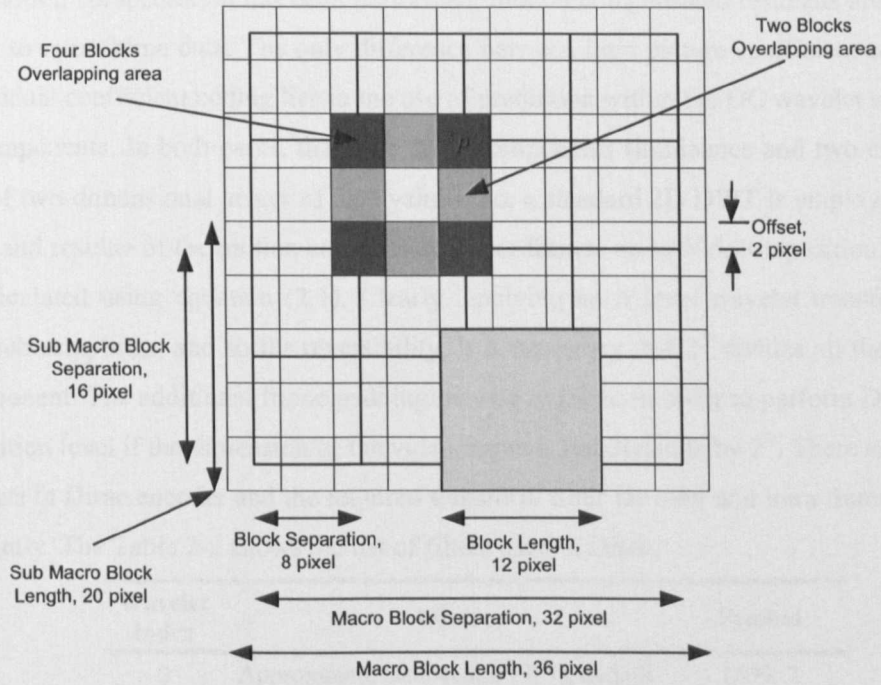


Fig. 2-7 Overlapped Block Structure of Dirac [3]

The predictor $\tilde{p}$ for p is the weight sum of all the corresponding pixels in the predicting blocks in reference frame, t'. The Raised-Cosine mask has the necessary property that the sum of the weights will always be 1. The predictor $\tilde{p}$ for MV set (V_i, W_i) for the frame t can be calculated as follow.

$$\tilde{p}(x,y,t) = \sum_{i=1}^{k} w_i p(x - V_i, y - W_i, t') \tag{2.4}$$

where, $\sum_i w_i = 1$, k is either 1, 2 or 4.

The value of k depends upon the physical location of the predicting pixel in the frame either within a single block, two overlapping blocks or four overlapping blocks.

2.5 Discrete Wavelet Transform (DWT)

As DCT is used in H.264, the DWT is used in Dirac in order to de-correlate the data in a roughly frequency-sensitive way, while having the advantage of preserving fine details better [4]. One of the weaknesses of DCT based encoder is that the picture goes all blocky when the encoder is applied under higher compression ratio because of the block based transformed method used in DCT. In contrast to DCT, DWT has been proven to be a more efficient technique where its transform can be applied to analyze the entire image without requiring block based transformation improving picture quality. This gives a wider coding spectrum and minimizes the block-edge effects. As a result, coding errors are spread out and artifacts found tend to be blended into the overall image. DWT is also used in the current JPEG2000 standard for image compression.

Once motion compensation has been performed, motion compensated residuals are treated almost identically to intra frame data. The only difference between intra picture coefficient coding and inter picture residual coefficient coding lies in the use of prediction within the DC wavelet subband of intra picture components. In both cases, there are three components (luminance and two chrominance) in the form of two-dimensional arrays of data values. So, a standard 2D DWT is employed to transform both intra and residue of the motion compensated inter frames up to N decomposition levels where N can be calculated using equation (2.1). Clearly, applying an N-level wavelet transform requires N levels of sub-samplings, and so for reversibility, it is necessary that 2^N divides all the dimensions of each component. The additional frame padding may be required in order to perform DWT for a given decomposition level if the dimension of the video frame is not divisible by 2^N. There are eight wavelet filter presets in Dirac encoder and the required transform filter for inter and intra frame can be chosen independently. The Table 2-2 shows the list of filters used in Dirac.

Wavelet Index	Filter	Symbol
0	Approximate Daubechies (9, 7), default	DD9_7
1	LeGall (5, 3)	LEGALL5_3
2	Approximate Daubechies (13, 7)	DD13_7
3	Haar with no shift	HAAR0
4	Haar with single shift per level	HAAR1
5	Haar with double shift per level	HAAR2
6	Fidelity filter	FIDELITY
7	Daubechies (9, 7) integer approximation	DAUB9_7

Table 2-2 Wavelet Filter Presets of Driac

In wavelet coding, the band splitting is done by passing the image data through a bank of bandpass analysis filters. Since the bandwidth of each filtered version of the image is reduced, they can now in theory be down-sampled at a lower rate, according to the Nyquist criteria, giving a series of reduced size sub-images. At the receiver, they are restored to their original sizes by passing through a bank of synthesis filters, where they are interpolated and added to reconstruct the image. In the absence of quantization error, it is required that the reconstructed picture should be an exact replica of the input picture.

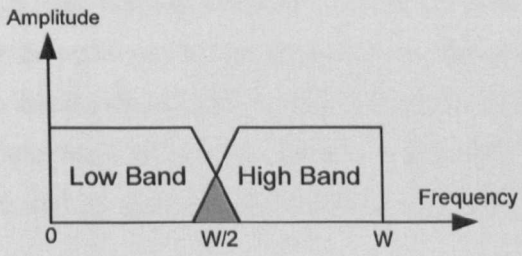


Fig. 2-8 A Two-Band Analysis Filter

This can only be achieved if the spatial frequency response of the analysis filter tiles the spectrum without overlapping, which requires infinitely sharp transition regions and cannot be realized practically. Instead, the analysis filter responses have finite transition regions and do overlap as shown in Fig. 2-8, which means that the down-sampling/up-sampling process introduces aliasing distortion into the reconstructed picture [5].

In order to eliminate the aliasing distortion, the synthesis and analysis filters have to have certain relationships such that the aliased components in the transition regions cancel out each other [5]. The corresponding one dimensional, two-band wavelet transform encoder/decoder is shown in Fig. 2-9. In this figure, $H_0(f)$ and $H_1(f)$ represent the frequency domain transfer functions of the respective low pass and high pass analysis filters. Filters $G_0(f)$ and $G_1(f)$ are the corresponding synthesis filters. The down-sampling and up-sampling factors are 2.

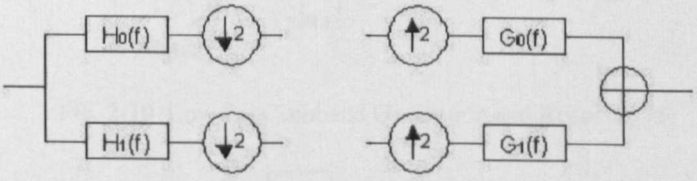


Fig. 2-9 A Two-Band Wavelet Transform Encoder and Decoder [4]

At the encoder, down-sampling by 2 is carried out by discarding alternate samples, the remainder being compressed into half the distance occupied by the original sequence. This is equivalent to compressing the source image by a factor of 2, which doubles all the frequency components present. The frequency domain effect of this down-sampling is thus to double the width of all components in the sampled spectrum causing aliasing.

At the decoder, the up-sampling is a complementary procedure. It is achieved by inserting a zero-valued sample between each input sample, and is equivalent to a spatial expansion of the input sequence. In the frequency domain, the effect is as usual the reverse and all components are compressed towards zero frequency. Fig. 2-10 shows this problem clearly and it is because of the impossibility of constructing ideal sharp-cut analysis filters.

Multi-dimensional and multi-band wavelet coding can be developed from the one-dimensional, two band low pass and high pass analysis/synthesis filter structure of Fig. 2-9. Wavelet coding of a two-dimensional image can be performed by carrying out a one-dimensional decomposition along the lines of the image and then down each column. A seven band wavelet transform coding of this type is illustrated in Fig. 2-11, where band splitting is carried out alternately in the horizontal and vertical direction. In this figure, L and H represent the low pass and high pass filters with a 2:1 down-sampling.

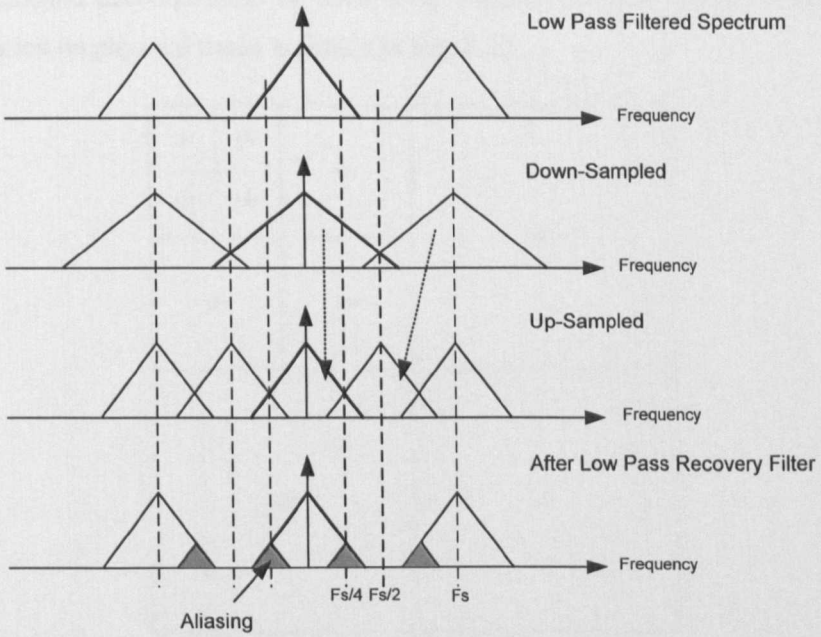


Fig. 2-10 Low Pass Subband Generation and Recovery [5]

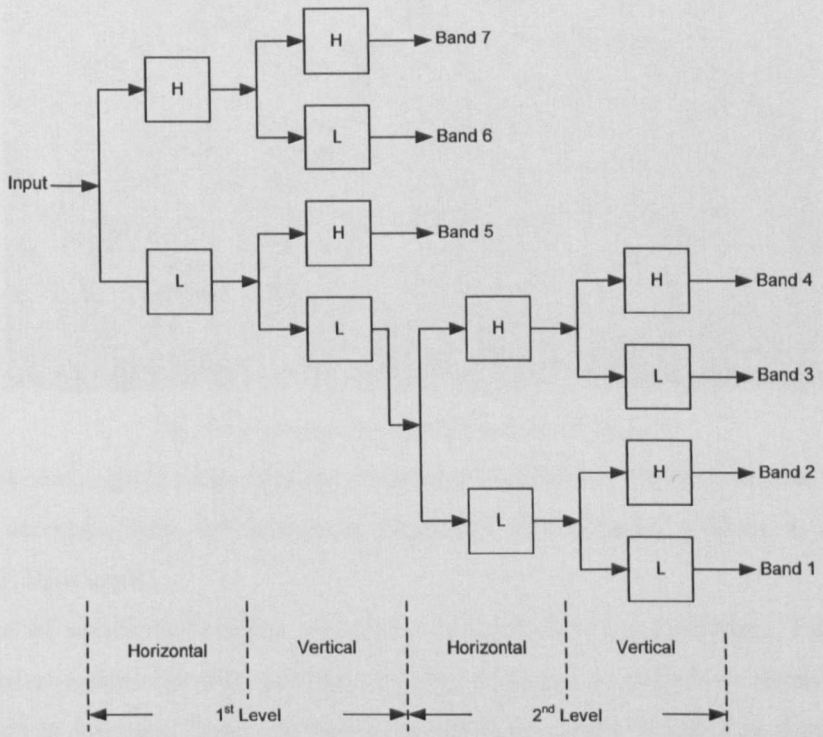


Fig. 2-11 Two-Dimensional, Multi-Band Wavelet Transform Coding using Repeated Two-Band Splits [5]

As shown in Fig. 2-11, a choice of wavelet filters is applied to each image component in both vertical and horizontal directions to produce four subbands termed Low-Low (LL), Low-High (LH), High-Low (HL) and High-High (HH) for each level. Only the LL band is iteratively decomposed to obtain the required decomposition level (in Fig. 2-11, there is only two level) yielding a series of sub

bands. The subband decomposition of three level wavelet transform showing their corresponding subbands location on physical frame is shown in Fig. 2-12.

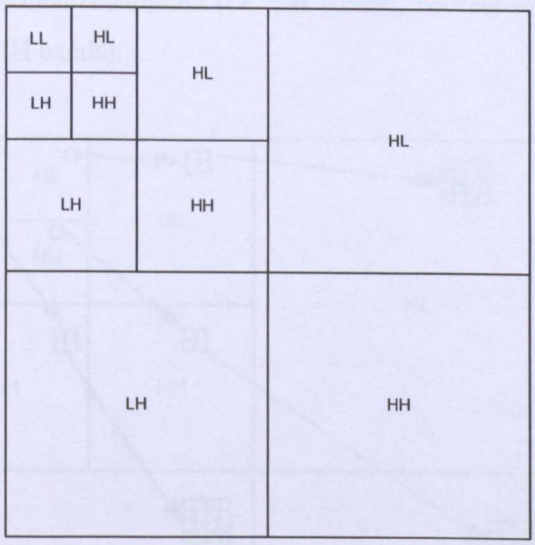


Fig. 2-12 Two Dimensional Wavelet Transform Frequency Decomposition (3 Levels) [4]

Fig. 2-13 3-Level Wavelet Transform of Lena [4]

Fig. 2-13 shows original image and the resulting 10 subbands (3 levels) wavelet transform of Lena image. After decomposition, the number of samples in each resulting subband is one quarter of the samples of the input signal.

The choice of wavelet filters has an impact on compression performance. Filters need to have compact impulse response in order to reduce ringing artefacts. It also has an impact on encoding and decoding speed in software. There are numerous filters supported by Dirac as shown in Table 2-2 to allow a trade-off between complexity and performance. These are configurable in the reference software.

Since each subband represents a filtered and subsampled version of the frame component, coefficients within each subband correspond to specific areas of the underlying picture and hence those that relate to the same area can be related. There is also stronger relation between the coefficients that have the same orientation (in terms of combination of high and low pass filters). The

relationship is illustrated Fig. 2-14, showing the situation for HL bands i.e. those that have been high pass filtered horizontally and low pass filtered vertically, LH bands and HH bands. Their relation can also be called as the horizontal orientation (i.e. LH bands), vertical orientation (i.e. HL bands) and diagonal orientation (i.e. HH bands).

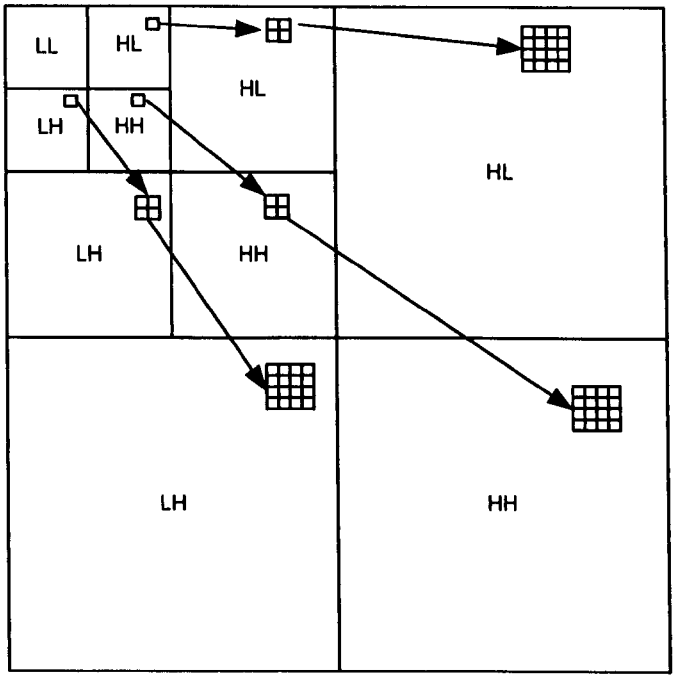


Fig. 2-14 Parent and Child Relationship between Subband Coefficients [4]

In the diagram it is easy to see that a coefficient (the parent) in the lowest HL band corresponds spatially to a 2×2 block of coefficients (the children) in the next HL band, each coefficient of which itself has a 2×2 block of child coefficients in the next band, and so on. This relationship relates closely to spectral harmonics: when coding image features (edges, especially) significant coefficients are found distributed across subbands, in positions related by the parent and child structure. In particular, a coefficient is more likely to be significant if its parent is non zero and on the other hand, if a parent is zero, it is likely that its children in the higher bands are zero.

When entropy coding the wavelet transformed coefficients, these factors will be helpful to take the parents into account in predicting how likely their children to be say, a zero. By coding from low frequency subbands to high frequency ones, and hence by coding parent before child, parent and child dependencies can be exploited in these ways without additional signalling to the decoder.

After transforming a frame into multiple subbands, each wavelet subband's coefficients are coded in turn using RDO quantization and entropy coding. Fig. 2-15 shows the architecture of wavelet coefficient coding in Dirac. The following sections will discuss these two functional blocks in details.

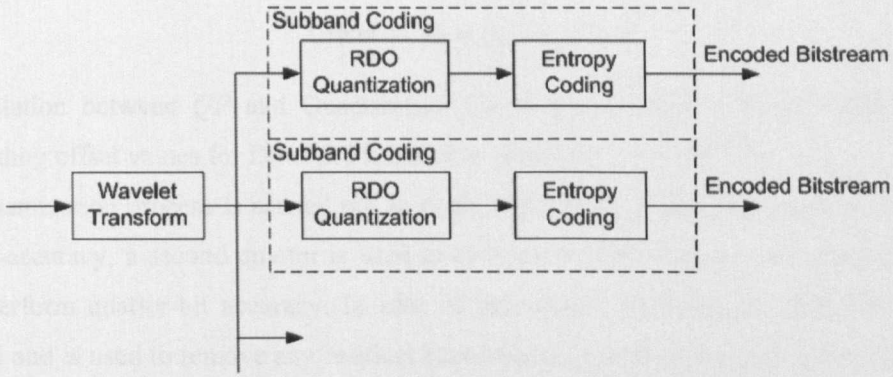
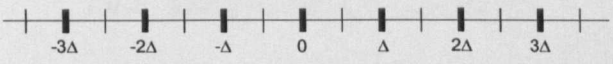


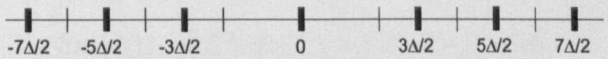
Fig. 2-15 The Architecture of Wavelet Coefficient Coding [4]

2.6 Rate Distortion Optimization Quantization

After having the image components treated by DWT, the resulting transform coefficients are quantized starting from the lowest frequency to the highest frequency subbands. There are 96 set of quantizers and the optimum Quantization Parameter (*QP*) index is chosen by RDO Quantization process. Dirac uses dead-zone quantizer in which the first region of the quantization steps is about twice wider than the uniform quantizer and so it applies more severe quantization of the smallest values, which acts as a simple but effective de-noising operation.



(a) Uniform Quantizer with Quantization Factor, Δ



(b) Uniform Dead-Zone Quantizer with Quantization Factor, Δ

Fig. 2-16 Uniform and Dead-Zone Quantizers [4]

Fig. 2-16 (a) and (b) show uniform quantizer and uniform dead zone quantizer, where the quantization factor is Δ with mid point reconstruction values shown in bold marking. But in Dirac, the reconstruction value is calculated by offset value which is 0.375 from the margin instead of taking mid point where offset is 0.5. It is because the values of transformed coefficients in a wavelet subband have a distribution with mean very near zero and which decays pretty rapidly and uniformly for larger values. Values are therefore more likely to occur in the first half of the interval than in the second half and the smaller value of 0.375 reflects this bias and gives better performance in practice.

$$\Delta = \left\lfloor 2^{2+\frac{QP}{4}} + 0.5 \right\rfloor, \quad 0 \le QP \le 96. \tag{2.5}$$

$$\text{Offset} = (\Delta \times 0.375) + 0.5 \tag{2.6}$$

The relation between *QP* and Quantization Factor (Δ) together with the calculation of their corresponding offset values for Dirac 0.6 is stated in equation (2.5) and (2.6).

The quantization process is carried out in three steps: firstly a quarter of coefficients are used to obtain bit-accuracy, a second quarter is used to estimate half-bit accuracy and the remaining half is used to perform quarter-bit accuracy. In case of intra frame quantization, coefficient prediction is performed and is used to remove any residual interdependencies between coefficients in the subbands allowing for effective entropy encoding. A coefficient is predicted by the mean value of the surrounding pixels of the current pixel and the difference is quantized and sent.

The current Dirac encoder uses an RDO technique to pick a quantizer by minimizing a Lagrangian combination of rate and distortion. Essentially, lots of quantizers are tried and the best is picked. Rate is estimated via an adaptively-corrected zeroth order entropy measure, *Ent*(Δ) of the quantized symbols resulting from applying the quantization factor (Δ), calculated as a value of bits/pixel. Total entropy for estimating the rate is the combination of *Ent*(Δ) and sign entropy, *SignEnt*(Δ) and can be calculated as follow.

$$Total\_Ent(\Delta) = Ent(\Delta) + SignEnt(\Delta) \tag{2.7}$$

Where, *Ent*(Δ) is measured in bit/pixel and can be found using equation (2.8) and (2.9) as follow.

$$Ent(\Delta) = -\left(P_0 \log_2(P_0) + P_1 \log_2(P_1)\right) \text{ bits} \tag{2.8}$$

$$Ent(\Delta) = \frac{Ent(\Delta) \times (num\_zero + num\_one)}{num\_coefficients} \text{ bits/pixel} \tag{2.9}$$

Where, in equation (2.8), P_0 and P_1 are the probabilities of zeros and ones. The same idea applies for the calculation of *SignEnt*(Δ) where the calculation is based upon the probabilities of positive and negative coefficients.

Distortion, *D*(Δ) is measured in terms of the perceptually-weighted fourth-power error, resulting from the difference between the original and the quantized coefficients. For a coefficient, P_{ij} and its corresponding quantized value, Q_{ij}, where *i* and *j* are row and column indices of a subband, the distortion can be found using equation (2.10) as follow.

$$D(\Delta) = \sqrt{\frac{\sum_{ij} |P_{ij} - Q_{ij}|^4}{num\_coeff \times w^2}} \tag{2.10}$$

Where, w is the perceptual weight associated with the subband where higher frequencies will have a larger weighting factor. Finally, the optimization is carried out by minimizing the combination of total entropy measure, $Total\_Ent(\Delta)$ as the rate and distortion, $D(\Delta)$ as follow.

$$D(\Delta) + \lambda.C.Total\_Ent(\Delta) \tag{2.11}$$

Where, C is the entropy correction factor which compensates any discrepancy between the measure of entropy and the actual cost in terms of bits, based on the actual bit rate produced by the corresponding elements of previous frames. It is calculated by dividing the actual coded bits to the estimated bits from the previous frame and applied to the current frame. It is necessary because the entropy measure does not take into account dependencies between coefficients that are taken into account in the actual coefficient entropy coding. In equation (2.11), λ is the Lagrangian multiplier and can be derived from the QF as shown in equation (2.3).

2.7 Entropy Coding

2.7.1 Wavelet Coefficient Coding

Dirac codes subbands from low-frequency to high frequency in a zig-zag order. Within each subband, the coefficients are further partitioned spatially divided into code blocks. Coefficients are scanned by coding the code blocks in normal raster order and coding the coefficients within each code block in raster order within that block. However, each code block can be skipped so a skip flag is included in the stream of symbols to be coded. The skip flag is interpreted in the decoder as setting all coefficients within the block to be zero.

The number of code blocks in a subband depends on the frame type and on the subband position. Intra frames have a single code block for the lowest-frequency subbands (i.e. the bottom 2 levels of the 4 levels wavelet transform) and a 4×3 array of code blocks for the remaining high frequencies bands. Predicted frames have more code blocks, as coefficients are more likely to be zero: only 1 block for the 4 lowest-level subbands; 8×6 for the next 3 subbands; 12×8 for the remaining subbands. There are two possible quantization modes made possible by the code block structure, which are a single quantizer for the whole subband or different quantizers for different blocks in a subband. The idea of multiple qunatizers for a subband is to allow the Region Of Interest (ROI) coding.

The entropy coding used by Dirac in wavelet subband coefficient coding is based on three stages: binarization, context modeling and adaptive arithmetic coding as shown in Fig. 2-17.

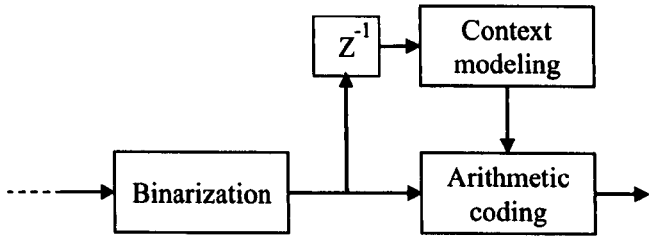


Fig. 2-17 Entropy Coding Structure [4]

Binarization is the process of transforming the multi-valued coefficient symbols into bits. The resulting bitstream can then be arithmetic coded. The original symbol stream could have been coded directly, using a multi-symbol arithmetic coder, but this process is more likely to cause context dilution since most symbols occur very rarely and so only sparse statistics can be gathered, which reduces coding efficiency. To avoid this problem, binarization is applied to transform the multi-valued coefficient symbols into bitstream with easily analyzable statistic that can be coded by arithmetic coding. Binarization can be done by using one of the Variable Length Coding (VLC) formats. In Dirac, VLC are used in three ways, (1) for direct encoding of the header values into bit stream, (2) for entropy coding of motion data and coefficients, where arithmetic decoding is not in use and (3) binarisation in the arithmetic encoding/decoding process. Unary code was used as VLC in which every non-negative number N is mapped to N zeros followed by a 1 and sign bit.

But, starting from Dirac 0.6.0 release, binarization for the magnitude values of wavelet coefficients, MV prediction residues and DC values has been changed from unary to interleaved exp-Golomb in order to reduce the number of symbols to be decoded in reconstructing a coefficient. Exp-Golomb binarisation takes a number $N \geq 0$ and codes it via the binary representation of N+1, 1bbbbbb. If there are K bits following the leading 1, the representation is K zeroes followed by the binary representation, i.e. 00...01bbbbb. As explained, conventional exp-Golomb coding places all follow bits at the beginning as a prefix. This is easier to read, but requires a count of the prefix length and can only be decoded in two loops, the prefix followed by the data bits. Interleaved exp-Golomb is the same, except that the K bits are interleaved with the zeroes, i.e. 0b0b...0b1 so that a single decoding loop can be used without the need for a length count. The zeroes here act as "follow bits" indicating that another bit is to be sent, with 1 as the terminator. Contexts for coding these symbols are selected for the follow bits and other bits ("information bits") separately. Compression performance is hardly affected, nor is speed performance in software, but hardware performance is greatly facilitated [6]. Dirac actually uses unsigned interleaved exp-Golomb and singed interleaved exp-Golomb codes separately where the former one is for unsigned integer coding and the latter consists of unsigned interleaved exp-Golomb code for the magnitude, followed by a sign bit for non-zero values.

In order to use arithmetic coding to compress data, a statistical model for the data is needed. The model needs to be able to accurately predict the frequency/probability of symbols in the input data stream and at the same time, need to deviate from a uniform distribution.

The need to accurately predict the probability of symbols in the input data is inherent in the nature of arithmetic coding. The principal of this type of coding is to reduce the number of bits needed to encode a character as its probability of appearance increases. So if the letter "E" represents 25% of the input data, it would only take 2 bits to code. If the letter "Z" represents only .1% of the input data, it might take 10 bits to code. If the model is not generating probabilities accurately, it might take 10 bits to represent "e" and 2 bits to represent "Z", causing data expansion instead of compression.

The second condition is that the model needs to make predictions that deviate from a uniform distribution. The better the model is at making predictions that deviate from uniform, the better the compression ratios will be. For example, a model could be created that assigned all 256 possible symbols a uniform probability of 1/256. This model would create an output file that was exactly the same size as the input file, since every symbol would take exactly 8 bits to encode. The number of bits can be reduced only by correctly finding probabilities that deviate from a normal distribution, leading to compression.

The idea of context modelling in Dirac does exactly the job as explained above in getting the statistical information of the input symbols. According to the nature of wavelet transform, the value of transformed coefficient (either zero, small or large) can be predicted well by its neighbours and parents. The context modelling in Dirac is based upon this idea. The reason for doing this approach is that whereas the wavelet transform largely removes correlation between a coefficient and its neighbours, they may not be statistically independent even if they are uncorrelated. Small and especially zero coefficients in wavelet subbands tend to clump together, located at points corresponding to smooth areas in the image, and are grouped together across subbands in the parent-child relationship.

The value of "small" depends upon the subband since the wavelet transform implemented in Dirac has a gain of 2 for each level of decomposition. So a threshold is set individually based on the subband type.

For predicting through neighbours, neighbourhood sum is calculated at each point (x, y) of each subband. It is the sum of the two previously coded quantized neighbouring coefficients and can be calculated as follow.

$$nhood\_sum(x,y) = |c(x-1,y)| + |c(x,y-1)| \tag{2.12}$$

And then, determine whether the parent coefficient is zero or not though parent and child relationship in the wavelet transform subbands. There are altogether 23 contexts used in transformed coefficients coding, which can be seen in appendix B.

After binarization, a context is selected and the probabilities for 0 and 1 that are maintained in the appropriate context will be fed to the arithmetic coding function along with the value itself to be coded. Contexts must be initialized with a count for both 0 and 1, which is used for encoding the first

symbol in that context. An additional source of redundancy lies in the local nature of the statistics. If the contexts are not refreshed periodically then later data has less influence in shaping the statistics than earlier data, resulting in bias, and local statistics are not exploited. Dirac adopts a simple way of refreshing the contexts by halving the counts of 0 and 1 for that context at regular intervals. The effect is to maintain the probabilities to a reasonable level of accuracy, but to keep the influence of all coefficients roughly constant [4] [7].

2.7.2 Motion Vector Data Coding

MV data coding is important to the performance of video coding, especially for codecs with a high level of MV accuracy (1/4 or 1/8 pixel). For this reason, MV coding and decoding is quite complicated, since significant gains in efficiency can be made by choosing a good prediction and entropy coding structure. The basic format of the MV coding module is similar to the coding of coefficient data: it consists of prediction, followed by binarization, context modelling and adaptive arithmetic coding as shown in Fig. 2-18.

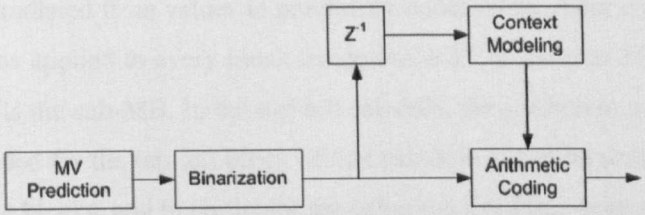


Fig. 2-18 MV Entropy Encoding Architecture [4]

All the MV data is predicted from previously encoded data from adjacent blocks (i.e. left, top left and top blocks). In predicting the data, a number of conventions are observed. The first convention is that all the block data (prediction modes, MVs and/or any DC values) is actually associated with the top-left block of the prediction unit to which they refer. This allows for a consistent prediction and coding structure to be adopted.

Example. If splitting level=1 and then the prediction units in a MB are sub-MBs (refers to Fig. 2-6). In this case, the prediction mode and any MVs are associated with the top-left block of each sub-MB and it is not required to encode other blocks' data in each sub-MB except top left.

The second convention is that all MB data is scanned in raster order for encoding purposes. All data is scanned first by MB in raster order, and then in raster order within each MB. That is, taking each MB in raster order and then each block value which needs to be coded within that MB is coded again in raster order as shown in Fig. 2-19.

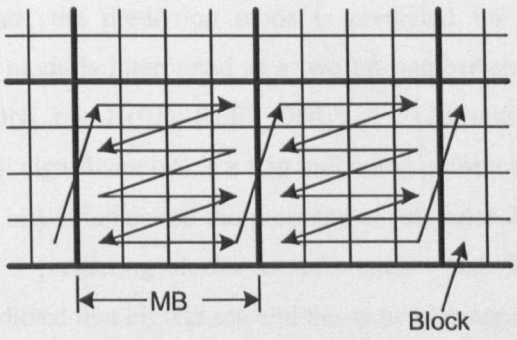


Fig. 2-19 Block Data Scanning Order [4]

The third convention concerns the availability of values for prediction purposes. Since prediction will be based on neighbouring values, it is necessary to propagate values for the purposes of prediction when the MV data has combined to ensure that values are not required for every block.

Example. Fig. 2-20 shows the requirement of propagation of values. Suppose that only REF1_x is being encoded. In the first MB, splitting level = 0 and so at most only the top-left block requires a value, which can be predicted from values in previously coded MBs. After encoding this value (i.e. *v*) it is then required to be applied to every block inside this MB. In the next MB, splitting level = 1, so the unit of prediction is the sub-MB. In the top-left sub-MB, the prediction mode is, say, REF1AND2 and so a value *x* is coded for the top-left block of that sub-MB. It can be predicted from any available values in neighbouring blocks, and in particular the value *v* is available from the adjacent block.

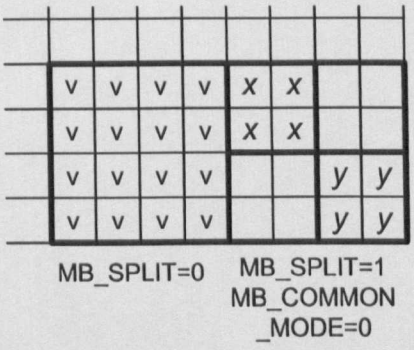


Fig. 2-20 Propagation of Data within MBs or sub-MBs [4]

The prediction used depends on the MV data being coded, but in all cases the aperture for the predictor is shown in Fig. 2-21. This aperture is interpreted as blocks where block data is concerned and MBs where MB data is concerned. The splitting level is predicted as the mean of the levels of the three MBs in the aperture.

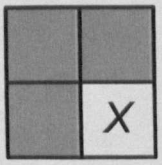


Fig. 2-21 Aperture for MV prediction [4]

From among block data, the prediction mode is predicted for reference 1 and reference 2 separately. The prediction mode is interpreted as a two bit number encoding with four possibilities: INTRA = 0 (00), REF1_ONLY = 1 (01), REF2_ONLY = 2 (10) and REF1AND2 = 3 (11). In this way the first bit (bit 0, least significant bit) is a flag indicating presence of reference 1 and the second bit (bit 1, most significant bit) is indication the presence of reference 2. Bit 0 and Bit 1 are predicted separately: if most of the 3 predicting blocks use reference 1 (i.e. their mode is REF1_ONLY or REF1AND2) then it is predicted that bit 0 is set, and the same idea applies for bit 1 prediction.

The DC values are predicted by the average of the three values in the aperture. The MVs themselves are predicted by predicting the horizontal and vertical components separately. The predictor for MV components is the median of the three corresponding values in the prediction aperture for the block.

In many cases MV or DC values are not available from all blocks in the aperture, for example if the prediction mode is different. In this case, these blocks are excluded from consideration. Where only two values are available, the median MV predictor becomes a mean. Where only one value is available, it will be the predicted value. Where no value is available, no prediction is made, except for the DC values, where 0 is used by default.

Chapter 3

3 Error-Resilient Coding Scheme

3.1 Introduction

In typical video communication system, the video is first compressed by a video encoder to reduce the data rate and the compressed bitstream is then segmented into fixed or variable length packets and usually undergoes a channel encoding stage, typically using Forward Error Correcting (FEC) to protect them from transmission errors.

Error control in video communication is very challenging for several reasons. First, compressed video streams are very sensitive to transmission errors because of the use of predictive coding and VLC. Due to the use of spatio-temporal prediction, a single erroneously recovered sample can lead to errors in the following samples in the same and following frames. Likewise, because of the use of VLC, a single bit error can cause the decoder to lose synchronization so that even correctly received following bits become useless. To make the compressed bitstream resilient to transmission errors, one must add redundancy into the stream, so that it is possible to detect and correct errors. Such redundancy can be added in either the source or channel coder.

3.2 Related Work of the Scheme

In the current application of real-time multimedia data streaming over the internet, routers drop packets randomly when output buffers are full without considering the relative importance of the packets. This may cause increased overall quality degradation in video communication because the important of the data in each packet varies. If the network is unable to transport all the data to the destination, one should guarantee that the most important part of the data is received to increase the reconstructed picture quality.

The most common way to protect data from packet losses is to request retransmission of any lost packets to the sender. However, unlike other application such as file transfer, real-time video applications may not benefit from retransmission-based error recovery because of the additional round-trip delay involved. Moreover, since packet losses often result from buffer overflow at the times of high network load, retransmitted packets make the network even more congested. Therefore, either source coding, channel coding (i.e. FEC) or combination of both (combined source and channel coding) is a more appropriate error control method for real-time Internet video applications.

Numerous sophisticated techniques have been developed over the last several decades to make the video transmission over a noisy channel resilient to errors. One approach is to transmit the video sequence into several bitstreams, called descriptions. In this method, a video sequence will be encoded into two or more bitstreams or descriptions and transmitted over multiple independent channels. The descriptions can be generated independently or correlatively. Any single description

should provide a basic level of quality, and more descriptions together will provide improved quality. The probability of failures of all channels is greatly decreased, so decoder will have a large probability to receive correct data from at least one channel at one time. When all of the descriptions are correctly received, the decoder can reconstruct the video with the best quality. If any of the description is lost during transmission, the decoder can still reconstruct the video with a lower, but acceptable quality. In [8], a new generation scheme using the side information was introduced which is based upon the multiple reference frames. A novel technique of Multiple Description Scalar Quantization for Fine Granularity Scalability (MDSQ-FGS) is presented in [9] to control the drifting error (error propagation) which is caused when reference data are lost in decoding. In Multiple Description Scalable Coding (MDSC), the advantages of scalable coding was combined with Multiple Description in order to adjust the amount of redundancy and the bitrate allocated to each description at transmission time [10]. Multiple description coding intentionally adds some redundancy among the descriptions causing lower compression efficiency (in fact, it is a trade off between compression efficiency and error resilience) and not suitable for the channel where the probability of losing all the descriptions is high (e.g. single carrier channel).

Kang [11] proposed an error resilient coding scheme for an H.264 video transmission. In his approach, for an H.264 I frame, the important data i.e. the edge direction within an MB is extracted and embedded into the next frame by the proposed MB-interleaving slice-based data embedding scheme. For P frame, two types (type 1 and type 2) of important data for each MB are extracted and embedded into the next frame. Type 1 data for an MB contains the coding mode, the reference frame(s) and the MV(s) for the MB, whereas the type 2 data includes the best Error Concealment scheme among 15 evaluated schemes for the MB. According to the simulation results, their proposed scheme can recover high-quality video frames from the corresponding corrupted video frames up to a video packet loss rate of 20%. But the important data extraction and embedding process for each and every MB could lead to a long encoding delay in real time application. The usage of multiple Error Concealment schemes could introduce the complexity to the encoder and as well as to the decoder, which are not the desirable features in real time video transmission.

In [12], Kim presents a new bit-plane-wise un-equal error protection algorithm for progressive bitstreams transmitted over lossy networks. The proposed algorithm protects a compressed bitstream generated by a 3-D Set Partitioning in Hierarchical Trees (SPIHT) algorithm by using systematic Reed-Solomon (RS) codes with an un-equal amount of redundancy to each bit-plane. The algorithm is fast and simple but error correction capability is limited to a certain extent especially in the hostile network conditions because of the usage of RS code.

Another approach, called coefficient partitioning makes image transmission resilient to channel errors by partitioning the wavelet coefficients into groups and independently processing each group. Thus, a bit error in one group does not affect the others, allowing more uncorrupted information to reach the decoder. This method was first reported by Creusere [13] for use with the EZW algorithm

and it is considered only for the image transmission. Block based coefficient partitioning method is presented in [14] where each subband data is partitioned into an equal number of coefficient blocks. Each coefficient block in a subband carries information about some localized region in the original frames. The components are then formed by grouping from each subband, equal number of coefficient blocks that correspond to different spatial regions of the source.

Some consider protecting the transmitted bitstreams against packet losses by applying an un-equal amount of FEC to different data fragments according to the importance of the data [15][16]. However, this technique has the disadvantage of still being vulnerable to packet erasures or channel errors that occur early in the transmission, either of which can cause a total collapse of the decoding process. To overcome this problem, combined source and channel coding has been considered in most cases where one of the coefficient partitioning methods is used as source coding and combined together with FEC to achieve double level of protection from transmission error [17].

In Pearlman's work [18], the wavelet transform coefficients is first broken into a number of spatio-temporal tree blocks according to [13], and the 3-D SPIHT algorithm is modified to work independently with these blocks. And then applies Kim's method [19][20] of RCPC channel coding as the FEC to every packet to protect the data.

It is interesting to note that the scheme proposed in [21] could be used in line with any error-resilient coding method mentioned above to alleviate the effect of error propagation by adding some periodic MBs in every fifth inter-frames.

3.3 The Objective of the Research

Widely development of the digital video applications has led the development of the next generation video codec called H.264/AVC [22] under the joint development of ITU-T's VCEG and ISO/IEC' MPEG is aimed to elaborate an open standard that is not application-specific and performs significantly better than the existing standards in terms of compression, network adaptation and error robustness. In the near future, H.264 will gain wide acceptance on many applications especially on Internet broadcasting. However, the usage of H.264 incurs royalty fees [23] which may not be cost effective for non-profit and public content owners such as public service broadcasters, archive institutes, etc., for deployment of Internet-based services. Whilst these costs are manageable initially, these could become prohibitive if the services scaling up to millions of users, or if new services are deployed which were not envisaged in the original license agreements.

As an alternative, a royalty-free general-purpose video codec called Dirac [1] is designed, which is aimed at a wide range of applications from storage of video content to streaming video in view to address the above demands. Being "open technology", Dirac is an attractive option as it allows content owners to distribute contents without royalty-fees in anyway.

However, current alpha releases of Dirac have only been optimized for storage purposes and still there is no error-resilient encoding mechanism. As mentioned in section 3.2, most of the error-

resilient coding schemes in the literature were designed to work with DCT based encoders and some of the combined source and channel coding strategies were indented for either image transmission or 3D wavelet transform based video encoder. So, it is still required to propose an error-resilient scheme which is suitable for two dimensional wavelet transform based Dirac video encoder. The main objective of this research is to propose a simple and low complexity error-resilient coding scheme and to investigate its performance. The proposed scheme is based on the combined source and channel coding approach and designed mainly for the packet-erasure channel, i.e. targeted for the Internet broadcasting application. But it is also possible to extend the scheme in order to suit in other type of channels (e.g. wireless).

3.4 The Combined Source and Channel Coding Scheme

In the existing Dirac's encoding architecture [1], after DWT stage, the resulting coefficients of each subband are scanned in raster order from lowest to highest frequency in order to undergo quantization and entropy coding processes. Even though this method is ideal for storage purpose, there could be serious problem if the encoded bitstream is transmitted into the erroneous channel. Because of the nature of entropy encoding, any single bit error in the middle of the bitstream could create the remaining part of the bitstream which belongs to the entire frame to become useless to the decoder.

In order to prevent this, it is required to divide the main bitstream into a number of several smaller independent bitstreams so that they are completely isolated to each other and the decoder can decode them independently. So, the bit error in one bitstream does not affect the decoding of the others, resulting more uncorrupted information at the decoder. Dividing the bitstreams in order to limit the affect of channel error or resilient to the channel error is called error-resilient coding. It is based on the source coding which is modification to the encoder side and one of the error-resilient coding methods available in the literature. The method of bitstream division or coefficient partitioning method for wavelet transformed based image coding was reported in [13]. Even though it is designed for image coding, it is possible to extend the idea to work as an error-resilient source coding for video encoder. But source coding only is not sufficient enough to protect the channel error in some case especially in the noisy channel where all the bitstreams are likely to be affected by the channel noise. So, it is required to add additional protection to the encoded bitstream in the form of channel coding. Combination of source and channel coding should protect the transmitted bitstream sufficiently from any kind of channel noises which normally occurs during the wireless or wired transmission. There are many types of FEC or channel coding methods in the literature with their various levels of complexities and performances. Among them, RCPC Code [24] and TC [25] are the most suitable methods because of its simplicity in the former and better performance in the latter one. A bitwise interleaver has been placed at the output of the channel encoder in order to distribute the series of information bits over the interleaving length so that a packet loss in the packet erasure channel or deep fading in the wireless channel does not reflect the loss in a large chunk of data. Instead, it is

equivalent to the loss of smaller portion of a whole packet, where the channel decoder at the receiving end can normally correct the error in most of the time. Cyclic Redundancy Check (CRC) has been used to detect the bit errors inside each packet so that the entropy decoder at the receiver can skip the erroneous packet and jump directly to the beginning of the another correctly received bitstream, preventing the possibility of decoder malfunctioning and saving decoding time.

In this work, transform coefficients coding algorithms such as Embedded Zerotrees Wavelet (EZW)[26], Set Partitioning in Hierarchical Trees (SPIHT) [27],Virtual Zero-Tree (VZT) [28], Zerotree Entropy (ZTE) [29] coidng, etc., are not considered since all of these are heavily patented and open source architecture of Dirac [1] prevents the use of these patented algorithms. Moreover, these algorithms do not perform very well in applying to the motion-compensated residual frames since most of the coefficients in these frames have already been transformed to zeros. Again, the application of these algorithms causes the perceptual weighting procedure to be more difficult. The perceptual weighting ensures the RDO quantization process to generate a larger weighting factor for the higher subband frequencies and vice versa. Finally, as far as the error resilient coding is concerned, the usage of these algorithms could introduce additional problems at the decoder if the received bitstream has the errors.

The detailed description of each mechanism which were used in error-resilient encoding of Dirac video encoder is given in the following sections.

3.4.1 Coefficient Partitioning, Source Coding

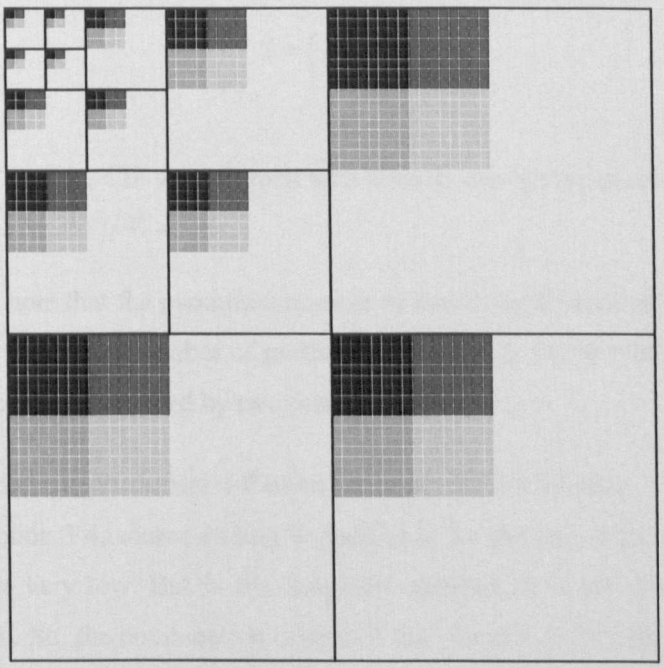


Fig. 3-1 Wavelet Coefficient Partitioning for $S = 4$ with Four Levels Wavelet Transform

The basic idea of the wavelet coefficient partitioning is to divide the wavelet coefficients at the output of the DWT process of the Dirac encoder into S groups and then quantize and code each of them

independently so that S different bitstreams are generated [13]. Quantization and coding in this stage is carried out by using Dirac's existing RDO quantizer and Arithmetic encoder[1].

By coding the wavelet coefficients with multiple independent bitstreams, any single bit error is truncated only in one of the S bitstreams while the others are still correctly received. Therefore, the wavelet coefficients represented by a corrupted bitstream are reconstructed at reduced accuracy, while those represented by the error-free streams are reconstructed at the full encoder accuracy. The partitioning method used here is the extension of [13], in which the idea in [13] is applied to the *motion-compensated residual frames,* instead of the intra coded frames for the image transmission in [13] and 3D wavelet transformed frames in [18]. In this way, the quality of the reconstructed frames particularly at high packet loss rate becomes much better than the schemes in [13] and [18] especially when the MV data and reference frames are correctly received. It is because the corrupted data can still be replaced with the exact replica pointed by the MV in the reference frame. The quality of reconstructed frame at the corrupted area mainly depends upon the accuracy of the motion estimation at this particular location and the quality of the reference frame. Since the motion compensated residual data is completely lost, decoder has to rely only on the data from the reference frame and MV data in order to reconstruct the corrupted area. But the error-resilient scheme presented in [18] uses 3D wavelet transformed frames resulting total loss at the corrupted area since there is no way to compensate the loss data.

If the image is of size $X \times Y$ and L levels of wavelet decomposition are used, then the maximum number of independent bitstreams allowed can be calculated as shown in equation (4.33).

$$S = (X \times Y)/4^L \tag{3.1}$$

For example,

$X = 352$, $Y = 288$, $L = 4$, i.e CIF video format with 4 levels wavelet transform

$S = (X \times Y)/4^L = (352 \times 288)/4^4 = 396$

It is important to note that the maximum number of partitions depends also on the chroma format. For YUV 4:2:0, the maximum number of partitions is limited to 396/4 = 99 since the dimensions of the U and V components are reduced by two times.

3.4.2 Rate-Compatible Punctured Convolutional (RCPC) Code

As mentioned in section 3.4, source coding is good only for the case where possibility of losing the transmitted packet is very low. But in the congested network, it is not good enough to protect the transmitted bitstream. So, the combination of source and channel coding becomes very popular since the transmitted bitstream will have double protection from the combined protective coding. But there is trade-off between the error protection using channel coding and bandwidth requirement since all of the channel encoders add the parity bits in the transmitted bitstream at the encoder so that the decoder

can correct or at least detect the error if there is error in the received packet. Basically, using higher number of parity bits normally offers better error correction performance with the requirement of higher level of bandwidth. The number of added parity bits can be adjusted dynamically in some channel coder so that less parity bit can be assigned while the network condition is good and increase the added number of parity bits gradually when the network becomes more and more congested. In this way, the level of bandwidth requirement can be controlled to a certain extent. Among from many types of channel encoders, RCPC code is the most suitable one because it offers flexibility in added amount of parity bits or encoding rate, with the optimum level of complexity and performance. Where, encoding rate is the ratio of uncoded information bits and the coded bits which is the combination of information and parity.

The concept of the RCPC code [24] is the extension of the traditional convolutional codes by puncturing a low rate $1/N$ code periodically with period P to obtain a family of codes with rate $P/(P + l)$ where l can be varied between 1 to $(N - 1)P$.

As for the channel coder in the proposed method, the value of N is chosen to be 4 and $P = 8$, i.e. one input and four outputs convolutional encoder, with mother code rate, $R = 1/N = 1/4$. By choosing the value of N to be 4 and $P = 8$, the code rates, R can be ($R = 8/(8 + l)$ where l can be varied between 1 to 24) varied between the lowest rate, 1/4 (i.e. $l = 24$) to highest rate, 8/9 (i.e. $l = 1$) giving the enough range of encoding rate required.

The following expression in equation (3.2) shows the puncturing table, which can be described by the $N \times P$ matrix to generate the rate, $R = 2/3$ convolutional encoder. A zero in the puncturing table means that the code symbol is not to be transmitted. In the puncturing table in equation (3.2), the first row corresponds to the first output of the convolutional encoder and the second row corresponds to second output and so on. Since all the values in the first row of the puncturing table are 1, this means, the bits from the first output of the encoder are not punctured. But there is alternative puncturing from the second output of the encoder for every 8 input information bits since the puncturing period, $P = 8$ and puncturing all the bits from the third and fourth output (i.e. totally discard the encoded bits from these outputs) to get rate, $R = 2/3$ convolutional encoder.

$$a(l) = a(4) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.2}$$

Where, $R = P/(P+l) = 8/(8+4) = 2/3$. The number of "one" in the matrix of equation (3.2) is 12 and this means that there will be 12 encoded bits at the output of the convolutional encoder for every block of 8 information bits at the input. So, $(P+l)$ or the number of encoded bits for every block of

P=8 information bits becomes 12 and from there the value of l can be calculated as $l=12-P=12-8=4$.

The codec provides the different coding rates which are 2/3, 1/2, 1/3, 1/4 and their corresponding puncturing tables are as follows.

$$a(l)=a(8)=\begin{pmatrix}1&1&1&1&1&1&1&1\\1&1&1&1&1&1&1&1\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\end{pmatrix} \tag{3.3}$$

Where, $R=P/(P+l)=8/(8+8)=1/2$.

$$a(l)=a(16)=\begin{pmatrix}1&1&1&1&1&1&1&1\\1&1&1&1&1&1&1&1\\1&1&1&1&1&1&1&1\\0&0&0&0&0&0&0&0\end{pmatrix} \tag{3.4}$$

Where, $R=P/(P+l)=8/(8+16)=1/3$.

$$a(l)=a(24)=\begin{pmatrix}1&1&1&1&1&1&1&1\\1&1&1&1&1&1&1&1\\1&1&1&1&1&1&1&1\\1&1&1&1&1&1&1&1\end{pmatrix} \tag{3.5}$$

Where, $R=P/(P+l)=8/(8+24)=1/4$.

As shown above, the puncturing tables in equations (3.2)(3.3)(3.4)(3.5) give the encoder rate 2/3, 1/2, 1/3 and 1/4, providing flexible coding rates required for the channel coding in the proposed scheme. The generator matrix of the mother code which has rate, R = 1/N = 1/4, can be expressed by the $N\times(M+1)$ matrix where M is the amount of memory inside the encoder. The generator matrix used for the convolutional encoder is shown in equation (3.6) where the amount of memory, M is 4. It is chosen according to the optimality criterion of Viterbi decoding algorithm [30], which are large free distance, small number of paths and small information error weigh.

$$G=\begin{pmatrix}1&0&0&1&1\\1&1&1&0&1\\1&0&1&1&1\\1&1&0&1&1\end{pmatrix} \tag{3.6}$$

The corresponding encoder structure of the generator matrix in equation (3.6) is shown in Fig. 3-2. The structure of the encoder is based on the simple feed forward design instead of using feedback structure. Even though feedback structure normally yields slightly better performance compared with feed forward approach, encoder using feedback structure requires longer encoder tail or termination bits in order to flush the memory inside the encoder or reset the encoder to the original all zeros stage. The effect of adding longer termination bits is usually negligible in wireless transmission where packetization is not necessary. But in internet broadcasting, adding longer termination bits to each and every packet would certainly reduce the overall throughput. For that reason, feed forward structure is chosen here, where the number of bits to terminate the trellis is usually not more than the amount of memory, M inside the encoder.

On the receiving side, it is assumed that the decoder using the Viterbi Algorithm (VA) [30] knows the current puncturing rule. Because of the use of puncturing, at the RCPC decoder, an $N \times P$ ambiguity has to be resolved in the incoming data stream and so the decision depths of the punctured codes are generally longer than un-punctured one. The decision depth of the trellis at the decoder is considered to be 100 bits so that the decoder gets the sufficient depth in finding the smallest accumulated error metric, i.e. surviving path.

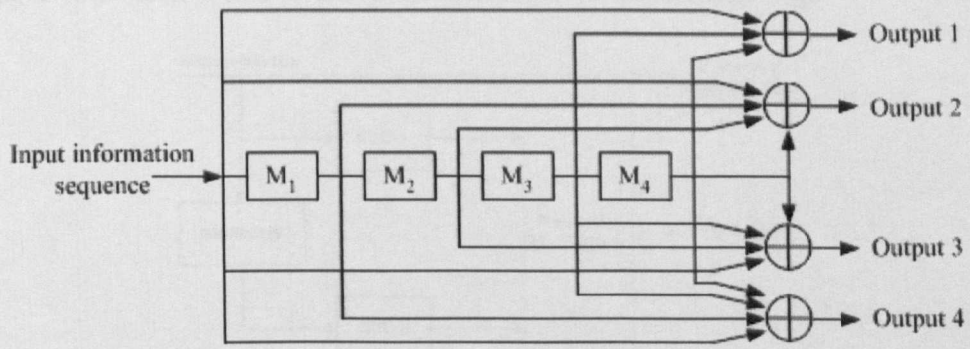


Fig. 3-2 Structure of the Rate 1/4 Convolutional Encoder

3.4.3 Turbo Coding (TC)

While using the combination of wavelet coefficient partitioning and RCPC coding as the combined source and channel coding can protect the serious channel noise to a certain level, it is still required to ensure that the transmission of header and MV information bits generated by the video encoder to be completely free from error. Safe delivery of this information is crucial in video transmission since it carries the vital information of the compressed video sequences including encoder parameters, sequence parameters and MVs data. So, it is required to protect these information by 100 % otherwise the decoder will simply fail to decode and the whole transmitted bitstream will become totally useless at the receiving side. Perfect protection of these data require more powerful channel coder since RCPC coder is able to protect perfectly up to a few packet loss rate. At this stage, complexity issue is second priority and generally the complexity at the decoder is negligible since the amount of header

and MV bits is very much lower than data in the coded bitstream. When complexity is not in crucial role, the most suitable type of channel coder for this task is Turbo Code because of its performance.

Turbo code [25], first presented to the coding community in 1993, represents the most important breakthrough in coding since Ungerboeck [31] introduced trellis codes in 1982. The original Turbo Code offers near Shannon's channel capacity performance for deep space and satellite channels. The invention of TC involves two Parallel Concatenated Convolutional Codes (PCCCs) in which the information bits are first encoded by a Recursive Systematic Convolutional (RSC) code and then, after passing through an interleaver, are encoded again by a second RSC encoder as shown in Fig. 3-3. The code sequences are formed by the information bits, followed by the parity check bits generated by both encoders. Puncturing at the puncturing mechanism in Fig. 3-3 is performed by taking the odd and even parity bits alternatively from the upper and lower RSC encoder outputs giving overall rate 1/2 TC encoder. Clearly, without puncturing, it is the rate 1/3 encoder, mapping a block of B input bits to $3B$ code bits. The interleaver inside the TC encoder is a pseudorandom interleaver with the interleaving length set to the length of the input block, B bits which is the length of the information bits plus CRC and encoder tails bits. This means that input binary information sequence is divided into L information bits each, add CRC plus encoder tail bits and undergoes Turbo encoding to each block of data (B bits) before sending to the packetizing stage.

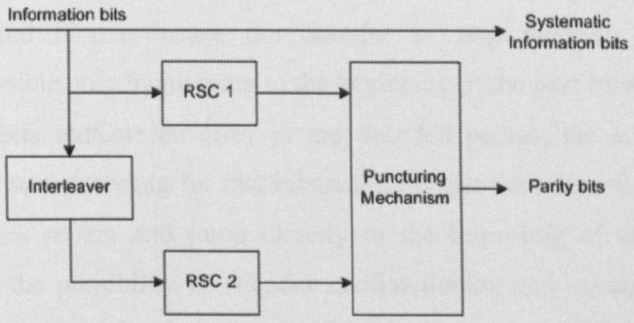


Fig. 3-3 Diagram of a Standard Turbo Encoder with two identical RSC Encoders

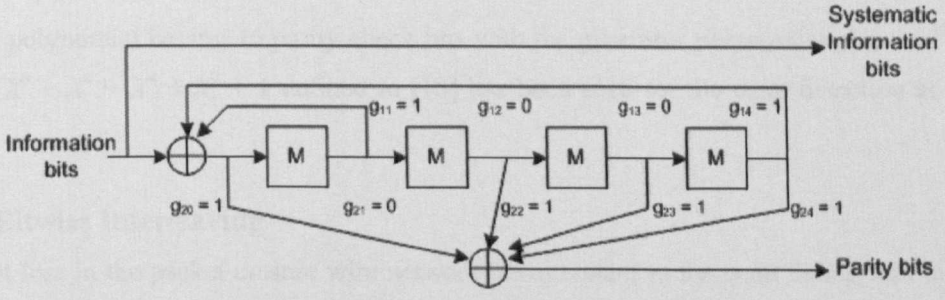


Fig. 3-4 The Structure of RSC Encoder for Code Generator $(g_1, g_2) = (31, 27)_{Octal}$

The generator polynomials of the two RSC encoders used here are chosen to be $G(D) = [g_1(D) \quad g_2(D)]$, where $g_1(D) = 1 + D + D^4$ and $g_2(D) = 1 + D^2 + D^3 + D^4$ with the number of memory, $M = 4$. The generator in octal form becomes, $g_1 = 31_{Octal}$ and $g_2 = 27_{Octal}$. Fig. 3-4 shows the complete structure of Recursive Systematic Convolutional encoder.

In the decoder, first, consider the Maximum-Likehood (ML) sequence decoder for a rate 1/2 convolutional code with block of L information bits. A ML decoder would have to compare 2^L code sequences to the noisy received sequence, choosing in favour of the codeword with the best correlation metric. Clearly, the complexity of such an algorithm would be unacceptable and un-applicable for real time decoding. Fortunately, such a brute force approach is simplified greatly by Viterbi's algorithm which permits a systematic elimination of candidate code sequences. Unfortunately, it is not applicable to Turbo Code because of the presence of the permuter or interleaver which immensely complicates the structure of a turbo code's trellis. Just prior to the discovery of turbo code, there was much interest in the coding community in suboptimal decoding strategies for concatenated codes. Among them, the method which received much attention was the symbol-by-symbol *Maximum a Posteriori* (MAP) algorithm [32]. Afterward, this algorithm was utilized in the iterative decoding of turbo code in [25] as BCJR algorithm. The algorithm used here is the modified version of BCJR called BCJR-MAP expressed in [33] with 6 decoding iterations.

3.4.4 Cyclic Redundancy Check, (CRC)

CRC is used to detect the bit errors inside each packet. It is required in order not to lose the synchronization at the arithmetic decoder with the received bit stream. When there is error in the received bitstream, because of the nature of VLC, arithmetic decoder loses synchronization with the received bitstream and it may cause the decoder to stop working or malfunctioning. Re-synchronization is possible only by jumping to the beginning of the next bitstream.

When the check bits indicate an error in the decoded packet, the error signal is sent to the arithmetic decoder to stop decoding for that bitstream. So that the arithmetic decoder at the receiver can skip the erroneous packet and jump directly to the beginning of another correctly received bitstream, preventing the possibility of decoder malfunctioning and saving the decoding time. The decoding procedure continues until either the final packet has arrived or a decoding failure has occurred in all sub-bitstreams.

CRC polynomial having 16 parity check bits with the generator polynomial $g(x) = X^{16} + X^{14} + X^{12} + X^{11} + X^8 + X^5 + X^4 + X^2 + 1$ defined in [18] has been used for the error detection at the received packets.

3.4.5 Bitwise Interleaving

A packet loss in the packet erasure wire network is equivalent to the deep fading loss in the wireless channel, where quite significant amount of information bits in serial or burst data is lost. The burst error correction capability of the channel coder is limited up to a few bits in series and it is absolutely impossible to correct or recover the data when a large chunk of data is affected by the channel errors. To overcome this, in most of the wireless transmission setup, a pair of channel interleaver and deinterleaver is used in order to break the correlation between the adjacent information bits or the

formation of burst error. The same idea can be applied in the packet erasure wired channel, where the coded bits are interleaved before packetization so that a packet no longer contains serial information bits preventing the formation of burst error when the packet is lost.

In order to achieve this, a bit wise interleaver is placed at the output of the channel encoder since the information to be transmitted is binary. The interleaver length is set to 100 times the length of the packet. That means that a packet loss in the packet erasure channel doesn't mean that the whole packet is lost instead the loss is only 1/100 of a packet. At the receiver, the channel decoder either RCPC or Turbo decoder can effectively correct those errors in most of the time since the interleaver has already eliminated the possibility of error burst formation.

3.5 Detail Encoding and Decoding Procedure of the Scheme

In this section, the overall error-resilient encoding and decoding procedure will be explained in detail. There are altogether 3 major stages which are

1. encoding the raw video material into error-resilient format (Source Encoding)
2. separating the compressed video files into two layers, header plus MV and data layer, and undergoes
   - 2.1. channel coding which includes adding CRC and tail bits
   - 2.2. bitwise interleaving
   - 2.3. channel simulation
   - 2.4. bitwise de-interleaving
   - 2.5. channel decoding which includes removing CRC and tail bits and
   - 2.6. marking with error code and multiplexing
3. decoding the multiplexed file from the previous stage (Source Decoding).

3.5.1 The Error-Resilient Source Encoding Procedure

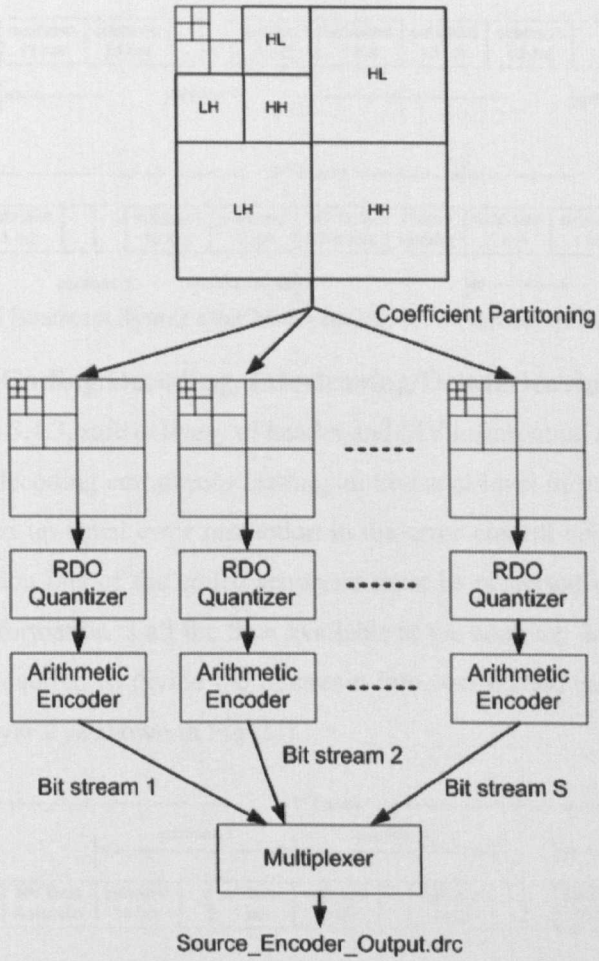


Fig. 3-5 Block Diagram of Error-Resilient Source Encoding Procedure using Wavelet Coefficient Partitioning Method

Fig. 3-5 shows the block diagram of error resilient source encoding procedure at the Dirac Encoder. Output of the DWT process of Dirac encoder is divided into S sub-frames according to the wavelet coefficient partitioning method mentioned in section 3.4.1. And then each of these sub-frames undergoes RDO quantization and arithmetic encoding independently so that S number of sub-bitstreams are generated. In the multiplexer, all the independent sub-bitstreams are combined together to get one serial bitstream. Multiplexing starts from bitstream 1 followed by bitstream 2 and so on until bitstream S is reached.

The resulting bitstream syntax after multiplexing is as shown in Fig. 3-6. It no longer follows the syntax of the current Dirac's specification [6] because of the result of multiple partitioning. But it is important to note that the resulting bitstream syntax in error-resilient format will be compatible with the current specification in [6] if the value of S is set to 1 i.e. no partitioning. The bitstream syntax after error-resilient source encoding is illustrated in Appendix C with the block diagram.

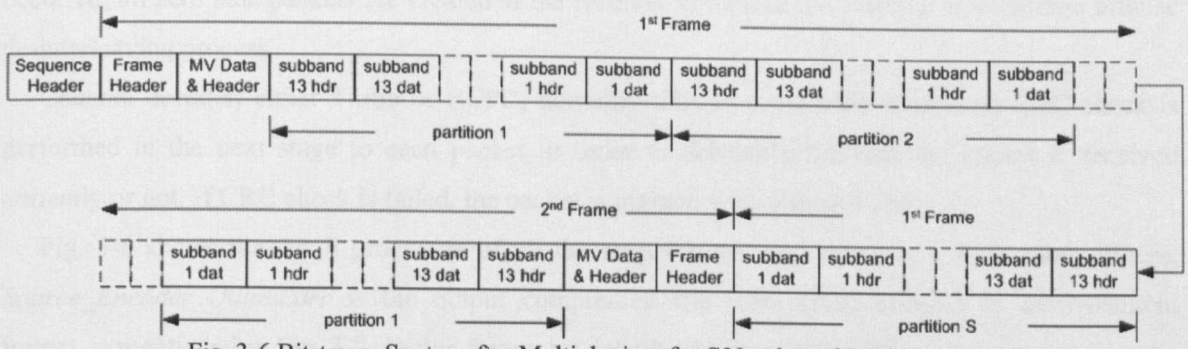


Fig. 3-6 Bitstream Syntax after Multiplexing, for S Number of Partitions

3.5.2 The Channel Coding/Decoding, Interleaving/Deinterleaving and Simulation

As discussed in section 3.4.3, safe delivery of header and MV information requires better protection at the expense of higher decoding complexity leading an un-equal level of protection to the compressed bitstream. This is called un-equal error protection in the error control coding literature. Header plus MV and data information bits of the coded sequence must be protected unequally in order to make sure that the former information is all the time available at the receiver. In order to perform un-equal error protection, it is required to divide the bitstream into two layers, namely header plus MV into layer 1 and data into layer 2 as shown in Fig. 3-7.

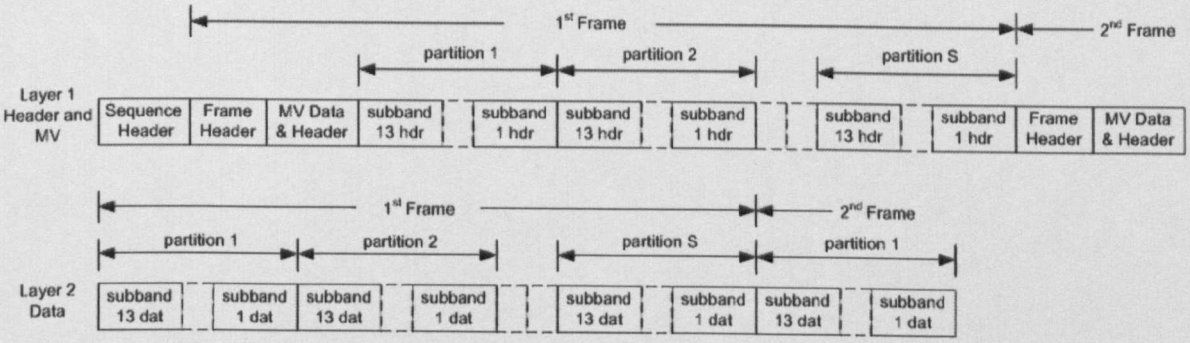


Fig. 3-7 Separation of Layer 1 and Layer 2 for Un-Equal Error Protection

And then applies un-equal error protection to these layers, i.e. layer 1 and layer 2 by using Turbo and RCPC encoder, respectively. The procedure of channel coding is to chop the bitstream of each layer into consecutive blocks of length L. Then, to each block, c checksum bits are calculated and added. Again, m zero bits are added to the end to flush the memory of the encoder (i.e. to terminate the trellis at zero stage). The resulting block of $L + c + m$ bits is then passed through a rate R, either Turbo or RCPC encoder depending upon the content of the L information bits.

Bitwise interleaving is then carried out by loading the pre-determined pseudorandom numbers which is known to both encoder and decoder to the buffer and interleaves the incoming binary sequence from the output of the channel coders, either Turbo or RCPC. The interleaver length is set to 100 times the length of the packet, i.e. $100 \times 1/R \times (L + c + m)$ bits.

Channel is considered to be the packet erasure wired channel and generating no bit errors inside each packet except the loss of the whole packet because of network congestion. If a packet lost is

occurred, all zero data packets are created at the receiver to replace the lost one and undergo bitwise de-interleaving process.

Channel decoder, either Turbo or RCPC, normally tries to correct the errors and CRC check is performed in the next stage to each packet in order to determine whether the packet is received correctly or not. If CRC check is failed, the packet is marked with the error code.

Fig. 3-8 shows the detail procedure of all the stages mentioned above in a flow chart. Where, *Source_Encoder_Output.drc* is the output compressed file from Dirac encoder in error-resilient format as mentioned in Fig. 3-5. In this figure, by default, the layer 1 and the layer 2 are protected by Turbo and RCPC channel coders, respectively. But, it is required to note that the lower rate RCPC (i.e. either 1/3 or 1/4) can also be used to protect the layer 1 for an environment where packet loss rate is minimal. Multiplexing in Fig. 3-8 ensures that the bitstream syntax of the *Channel_Decoder_Output.drc* which is received erroneously follows the same syntax specified in Fig. 3-6.

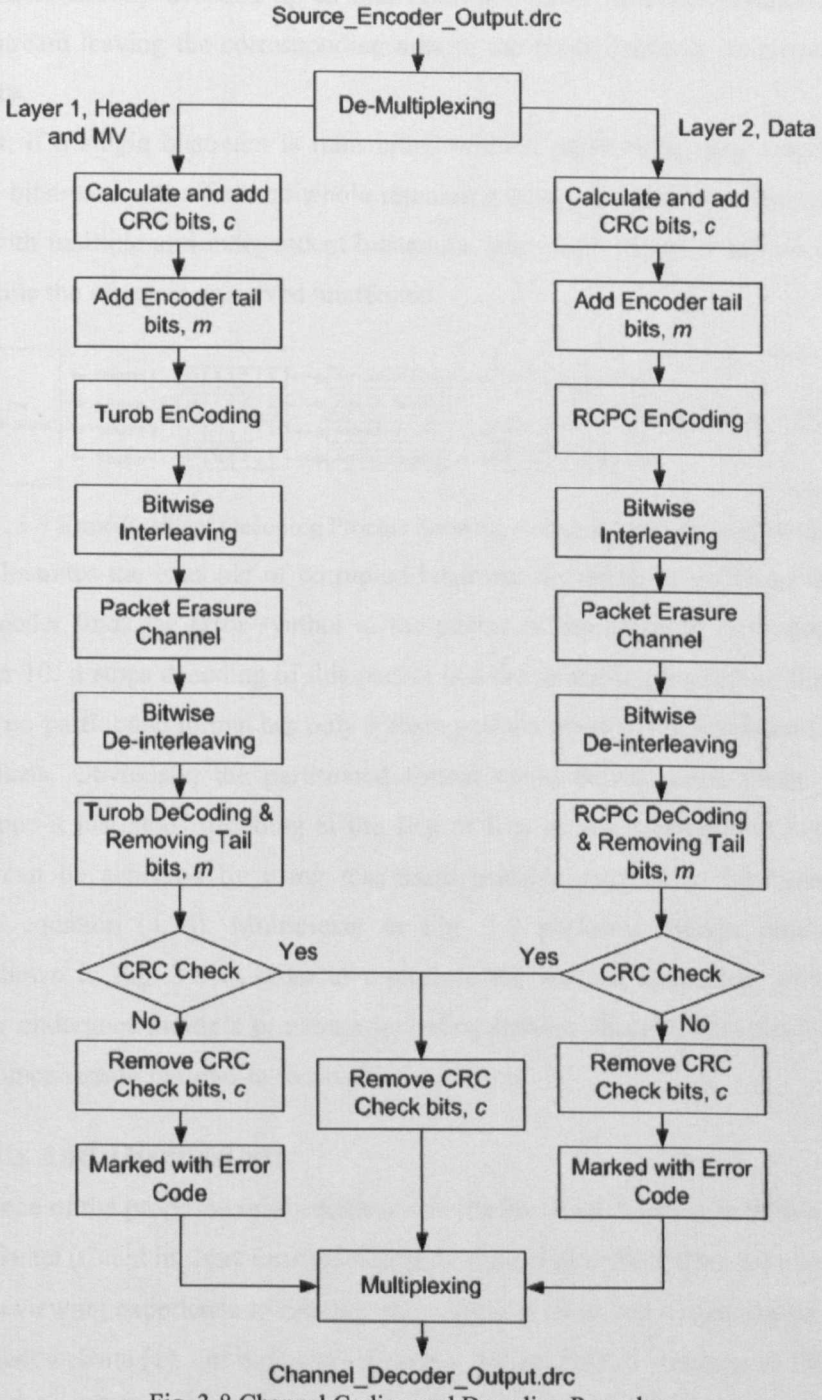


Fig. 3-8 Channel Coding and Decoding Procedure

3.5.3 The Error-Resilient Source Decoding Procedure

Decoding the multiplexer output from the previous stage which is *Channel_Decoder_Output.drc* can be carried out by undergoing the error-resilient decoding procedure shown in Fig. 3-9. De-multiplexer in Fig. 3-9 separates the input bitstreams into S number of independent bitstreams which is actually the reverse process of multiplexing in Fig. 3-5. The arithmetic decoder is modified so that it stops decoding once the error symbol is found in the bitstream being decoded and simply jumps to the other bitstreams. The decoder continues to decode the packets of the other streams so that the receiver still

has clean packets already decoded up to that point and loses only the remaining packets of the corrupted bitstream leaving the corresponding area in the entire frame to be reconstructed with the reduced quality.

In contrast, if a single bitstream is transmitted without partitioning, any single bit error in the middle of the bitstream will affect the whole remaining bitstream. Therefore, by coding the wavelet coefficients with multiple and independent bitstreams, any single bit error affects only one of the S bitstreams, while the others are received unaffected.

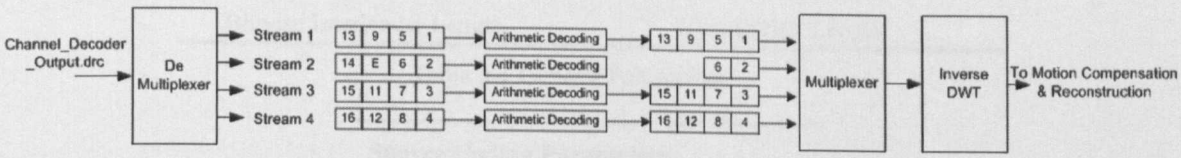


Fig. 3-9 Error-Resilient Decoding Process Showing Corrupted Packet in Bitstream 2

Fig. 3-9 illustrates the example of corrupted bitstream decoding in the Dirac decoder. Once the arithmetic decoder finds the error symbol in the packet of the received bitstream, i.e. bitstream 2 packet number 10, it stops decoding of this packet and the remaining packets of this bitstream. After decoding, the un-partitioned format has only 9 clean packets while in the partitioned format still retain 14 clean packets. Obviously, the partitioned format could deliver more clean packets than un-partitioned since it just stops decoding at the step of first error occurrence. A better error resilient performance can be achieved by using maximum possible number of bitstreams which can be calculated by equation (4.33). Multiplexer in Fig. 3-9 performs reverse process of coefficient partitioning shown in Fig. 3-5 in order to reproduce the wavelet coefficient subbands. After that, Dirac decoder undergoes multiple processes including Inverse Discrete Wavelet Transform (IDWT) and motion compensation in order to reconstruct the frame.

3.6 Results and Discussions

The performance of the proposed error-resilient scheme for Dirac is tested with two sequences: Canal Vertical Pan Street (Canal in short form) in CIF and Squirrel in SD576; they are chosen to envisage in providing two viewing experience to Internet users. Both of these sequences can be downloaded from the Dirac project website [1]. For both test sequences, a default GOP structure of Dirac is used, i.e. 36 for CIF and 12 for SD576 where L_1 frame separation is 3, and the number of L_1 frames between I frames is 11 and 3 correspondingly. In all the experiments which include channel simulation, Dirac's header and MV or layer 1 is considered received successfully from the channel, i.e. no packet loss is introduced to the layer 1 since it will be protected by using stronger channel code in actual delivery as shown in Fig. 3-8. In the experiment, it is assumed that a 16 bits synchronization marker is used to separate the independent bitstreams generated from the source coding. The objective quality is measured by the Peak Signal-to-Noise Ratio (PSNR). All the PSNR values are averaged over 10 independent runs. The general, source coding and channel coding parameters that are used in the

experiments are summarized in Table 3-1, Table 3-2 and Table 3-3, respectively. The idea behind the consideration of different number of partitions to different video components and format in source coding is detailed in Appendix D.

General Parameters	
Block Length ($B = L + c + m$)	200 bits
Number of CRC bits (c)	16 bits
Number of Information bits/Packet (L)	$B - c - m$
Packet Length	$1/R(L+c+m) = B/R$
Bitwise Interleaver Length	100×Packet Length

Table 3-1 General Parameters

Source Coding Parameters		
Components	Y	UV
CIF Video Format		
Number of Partitions (S_1)	33	33
Number of Partitions (S_2)	99	99
Number of Partitions (S_3)	6	3
Number of Partitions (S_4)	22	3
Number of Partitions (S_5)	22	9
SD576 Video Format		
Number of Partitions (S_6)	18	6

Table 3-2 Source Coding Parameters

Channel Coding Parameters	
Rate Compatible Punctured Code (RCPC)	
Number of Memory (M)	4
Number of Encoder Output (N)	4
Encoder Rates (R)	2/3, 1/2, 1/3, 1/4
Puncturing Interval (P)	8
Decision Depth (> P×N)	100 bits
Number of Encoder Tail bits (m)	4
Turbo Code (TC)	
Number of Memory (M)	4
Number of Encoder Output (N)	2
Interleaver Length	Block Length (B)
Number of Encoder Tail bits (m)	8

Table 3-3 Channel Coding Parameters

3.6.1 Compression Efficiency of Error-Resilient Source Coding

First of all, the compression efficiency of the Dirac encoder with error-resilient format source coding is tested with different number of partitions given in Table 3-2 and compared with the normal encoding result without having any partition. Table 3-4 lists the number of partitions used in the experiment for each video component, the compressed video file sizes shown in percentage of the

uncompressed YUV file and their corresponding average PSNR values. It is also possible to set the number of partitions for Y and UV components independently, i.e. different number of partitions for luma and chroma components.

The use of coefficient partitioning incurs compression inefficiency which is directly proportional to the number of partitions. It is because the arithmetic encoder needs to restart as many times as the number of partition to encode one frame in the multiple partitions format. It can be considered as the trade-off between the compression efficiency and the error-resilient performance. The compression inefficiency become prominent in employing the multiple partitions to the smaller frame size video sequence such as CIF format since there are less coefficients in each partition causing less statistical information providing to the context based Arithmetic encoder. But on the other hand, larger frame size video such as SD576 does not suffer much; the compression efficiency loss is only 0.12 % in 18-6P format where 18 and 6 represent the number of partitions for Y and UV components, respectively, because of having higher number of coefficients in each partition.

Moreover, average PSNRs of the multiple partitions formats are slightly less than that of un-partitioned one. It is because in the RDO quantization process, prediction of the optimum *QP* for each subband is based upon entire frame and applied on the coefficients of the subband in the partitioned frames leading the un-optimized quantizer values to be used at the encoder. The problem can be solved by modifying the RDO quantization algorithm in order to work with the subband of partitioned frame instead of the entire frame so that average PSNR of multiple partitions format can be the same or even better than un-partitioned one. Nonetheless, average PSNR loss is only a few portion of a dB and it can be considered as negligible for a video encoder which is optimized mainly for subjective quality [34].

		Format	Num. of Y Partitions	Num. of UV Partitions	Compressed File Size (% of YUV)	Avg. PSNR (dB)
Un-Partitioned	CIF	UP	1	1	1.62	32.53
	SD576	UP	1	1	2.60	32.23
Partitioned	CIF	6-3P	6	3	1.78	32.28
		22-3P	22	3	2.15	32.15
		22-9P	22	9	2.31	32.15
		33P	33	33	3.14	32.17
		99P	99	99	6.30	32.14
	SD576	18-6P	18	6	2.72	32.00

Table 3-4 The Compressed Video File's Sizes in Percentage and their Corresponding Average PSNR Values for Different Number of Partitions

Fig. 3-10 and Fig. 3-11 show the PSNR performance comparison between different partition formats of source coding with un-partitioned or without source coding result. Following the results in Table 3-4, Fig. 3-10 and Fig. 3-11 also show slight degrading of PSNR performance for each frame in the multiple partitions formats compared with un-partitioned one.

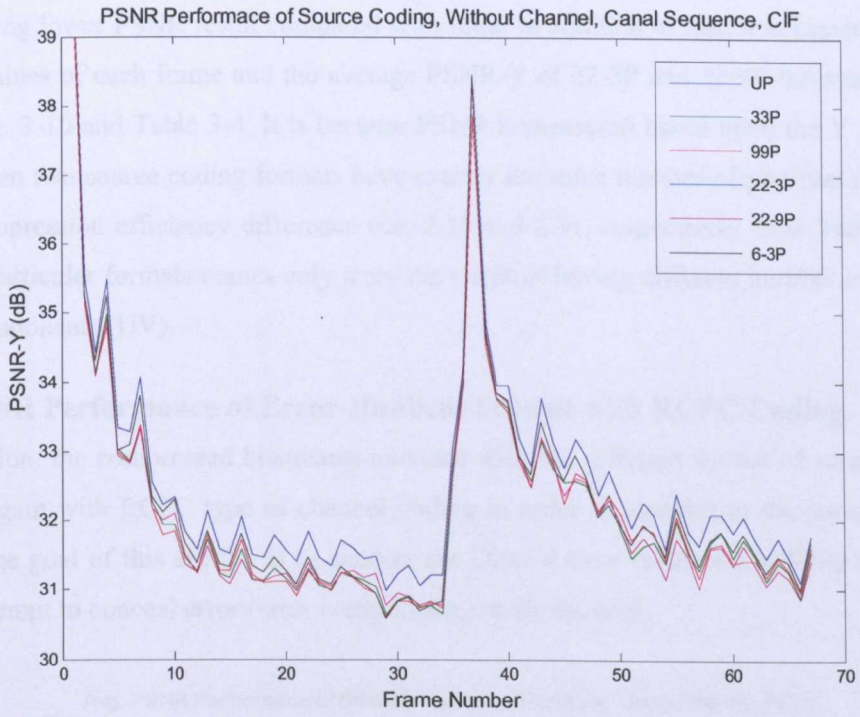


Fig. 3-10 PSNR Performance Comparison of Different Format of Source Coding, Without Channel, Using Canal Sequence in CIF Format

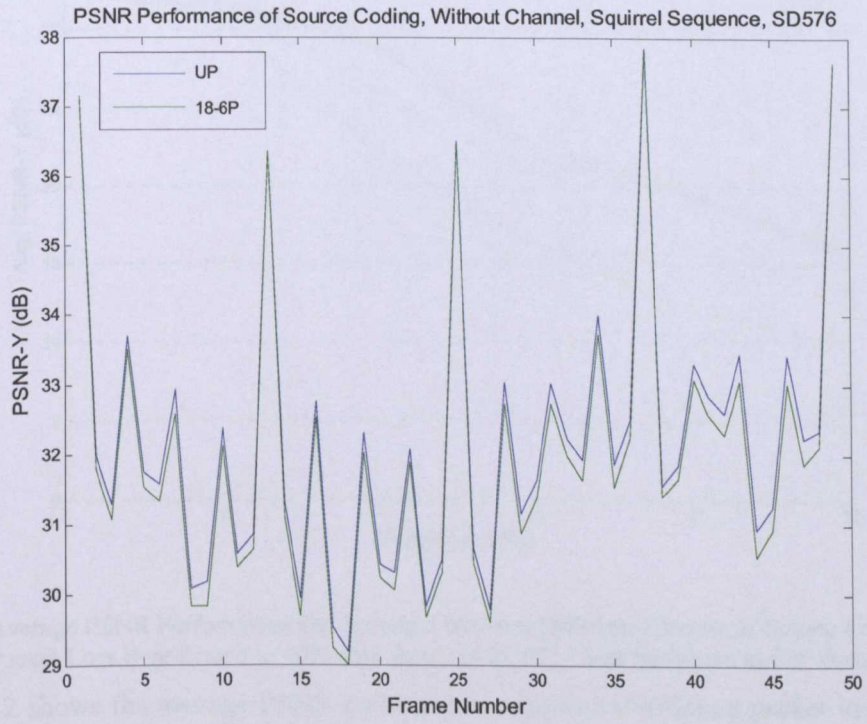


Fig. 3-11 PSNR Performance Comparison of Dirac, With and Without Source Coding, Without Channel, Using Squirrel Sequence in SD576 Format

But it is interesting to see that the PSNR reduction is more prominent at the inter frames (i.e. L_1 and L_2 frames) compared with intra frame (i.e. I frame). It is because inter frames which have lower

DWT coefficient weight because of the result of motion estimation are more sensitive to quantzation error resulting lower PSNR result compared with intra. In addition to this, it is important to note that PSNR-Y values of each frame and the average PSNR-Y of 22-3P and 22-9P formats are exactly the same in Fig. 3-10 and Table 3-4. It is because PSNR is measured based upon the Y component only and the given two source coding formats have exactly the same number of partition in Y component. So, the compression efficiency difference (i.e. 2.15 and 2.31, respectively from Table 3-4) between these two particular formats comes only from the result of having different number of partition in the chroma components (UV).

3.6.2 PSNR Performance of Error-Resilient Scheme with RCPC Coding

In this section, the compressed bitstreams encoded with the different format of source encoding are protected again with RCPC type of channel coding in order to transmit to the packet erasure wired channel. The goal of this section is to analyze the Dirac's error-resilient capability and so there has been no attempt to conceal error (error concealment) at the decoder.

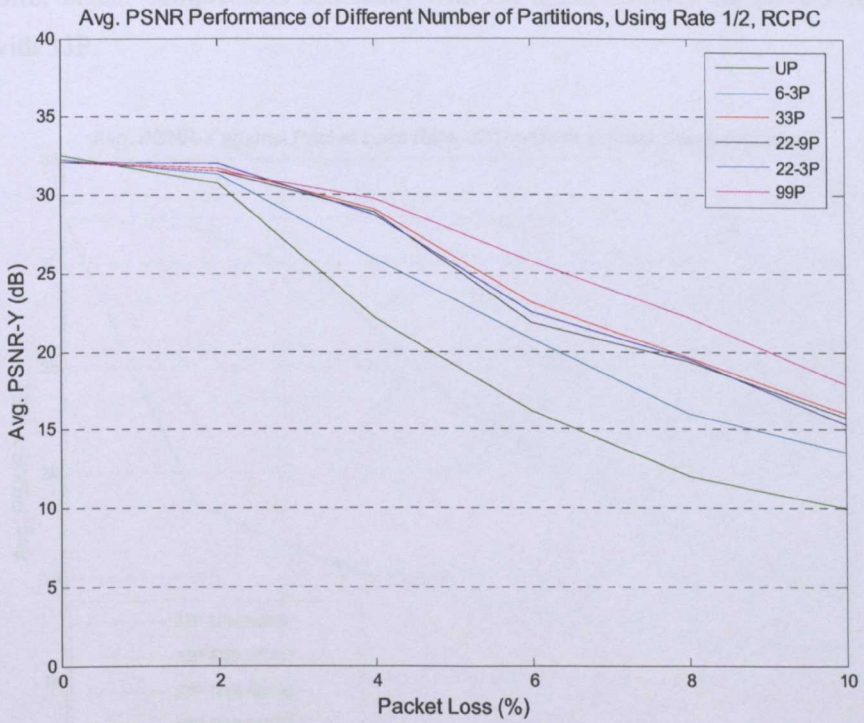


Fig. 3-12 Average PSNR Performance Comparisons between Different Formats of Source Coding for the Packet Loss Rate from 0 to 10% with Rate 1/2 RCPC, Canal Sequence in CIF Format

Fig. 3-12 shows the average PSNR performance against the different packet loss rate for Canal sequence in CIF format with rate 1/2 RCPC coding. As expected, 99P (99P Partitioned format) offers best protection from among the five different error-resilient source encoding formats and the partition gain (gain from the result of error-resilient source encoding only) is nearly 10 dB at 8 % packet loss rate when compared with Un-Partitioned (UP) format. From the figure, it can be seen that the partition

gain tends to increase gradually starting from 2% packet loss rate and the maximum partition gain achieved for the other formats i.e. 6-3P and 33P are 4 dB and 8 dB, respectively again at the 8% packet loss rate. A better error-resilient performance can be achieved if higher number of partitions is used at the expense of lower compression efficiency.

According to the experimental results shown in Fig. 3-12, there is no significant improvement in terms of partition gain or PSNR performance between the formats 33P, 22-9P and 22-3P. It is because all of these formats have approximately the same number (or exactly the same in latter two formats) of partitions in Y component resulting PSNR-Y performance which are very close to each other.

In the later part of this experiment, only 33P format will be used in all cases of source coding with CIF format since it offers optimum performance as far as the partition gain and complexity are concerned. Moreover, because of the usage of the same number of partition for both Y and UV components, the application of error concealment become more easier than other format where there are different number of partitions for different components. But, it is important to note that if the error concealment is not the major concern, either 22-9P or 22-3P formats should be used instead of 33P since they offer higher compression efficiency with the approximately the same PSNR performance compared with 33P.

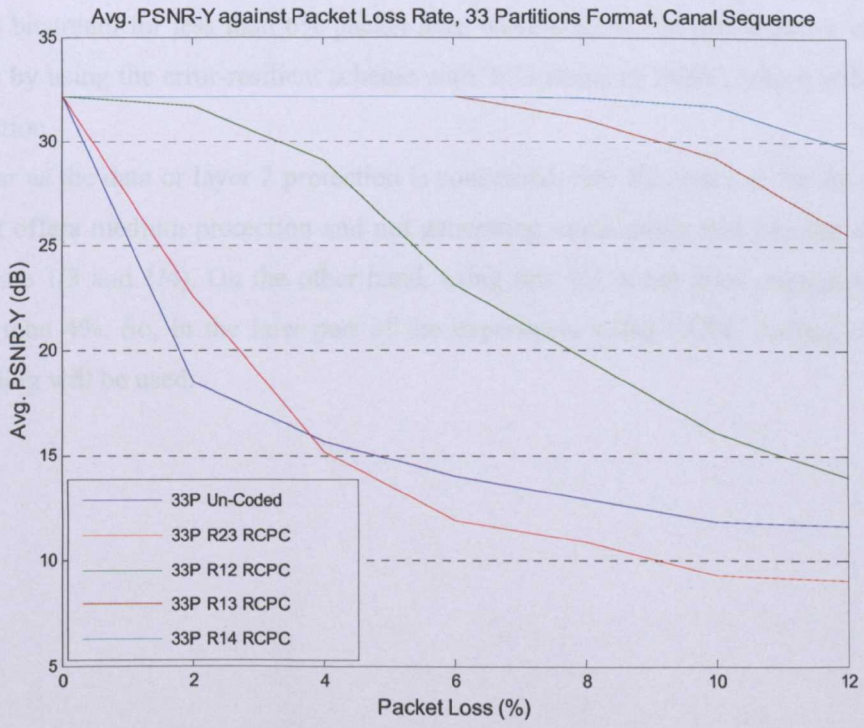


Fig. 3-13 Average PSNR Performance Comparisons between Rate 2/3, Rate 1/2, Rate 1/3, Rate 1/4 and Un-Coded (Without Channel Coding) of 33 Partitions Format for the Packet Loss Rate from 0 to 12% with RCPC Coding, Canal Sequence in CIF format

Fig. 3-13 shows the average PSNR performance comparison between rate 2/3 (33P R23 RCPC), rate 1/2 (33P R12 PCPC), rate 1/3 (33P R13 PCPC), rate 1/4 (33P R14 PCPC) and un-coded (i.e.

without channel coding) (33P Un-Coded) of 33P partitioned format source coding using RCPC channel coding. The packet lost rate is varied from zero to 12% and Canal test sequence in CIF format was used. It is interesting to note that the performance of rate 2/3 encoding achieves a few dB gains over the un-coded result for the packet loss less than 4% and cross over occurred after that. This is because the error correcting capability of the rate 2/3 RCPC decoder is relatively low and cannot correct the errors effectively when the packet loss rate increases. At this point, because of the usage of bitwise interleaving in rate 2/3 encoding, packet errors are spread over the interleaving length resulting the PSNR performance even lower than the un-coded case. But in the un-coded case where no channel coding is used, bitwise interleaving is not required and hence giving better PSNR performance for the condition where there is higher packet loss rate.

On the other hand, rate 1/2, 1/3 and 1/4 offer better error correcting capabilities and achieve much higher PSNR gain than the un-coded. The coding gain (the gain from the result of channel coding only) over un-coded case is around 4 dB, 17 dB and 20 dB for the rate 1/2, 1/3 and 1/4, respectively at the 10% packet loss. From Fig. 3-13 again, it is clear that there are no losses in terms of PSNR performance in the rates 1/3 and 1/4 encoding; according to the simulation results, there are no bit errors at the output of the RCPC decoder from 1% to 6% packet loss at these encoding rates. So, it is safe to conclude that encoder rates 1/3 and 1/4 are able to be used to protect the layer 1 of the Dirac's compressed bitstream for less than 6% packet loss. Nevertheless, the performance will be improved enormously by using the error-resilient scheme with TC instead of RCPC, which will be discussed in the next section.

But as far as the data or layer 2 protection is concerned, rate 1/2 seems to be the optimum coding rate since it offers medium protection and not generating much parity bits like the other lower rates encoding (rate 1/3 and 1/4). On the other hand, using rate 2/3 is not good enough to protect packet error more than 4%. So, in the later part of the experiment using RCPC coding, only the rate 1/2 channel coding will be used.

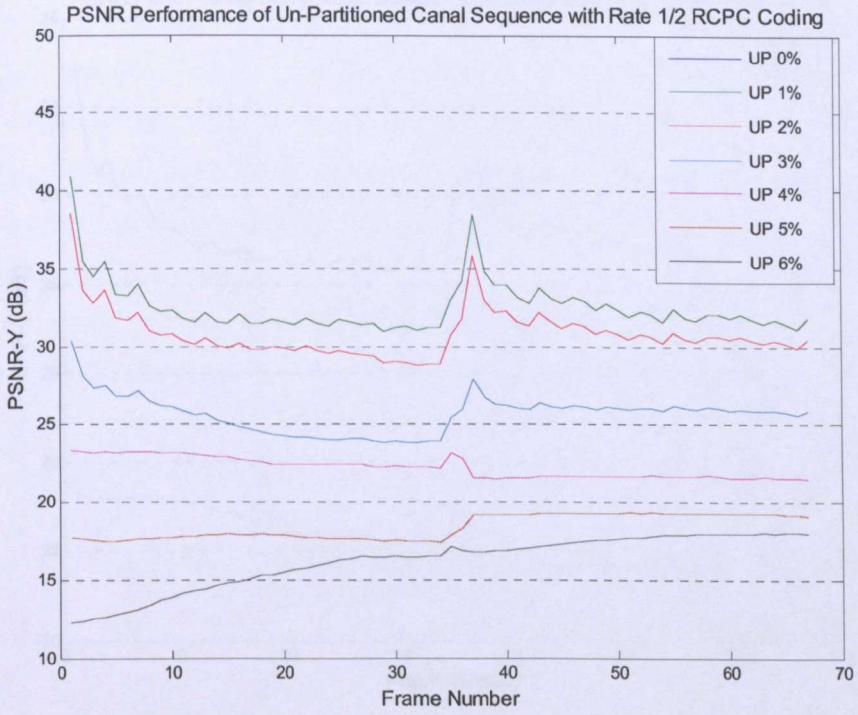


Fig. 3-14 PSNR Performance Comparisons between Different Percentages of Packet Loss for Un-Partitioned Format with Rate 1/2 RCPC Coding, Canal Sequence in CIF Format

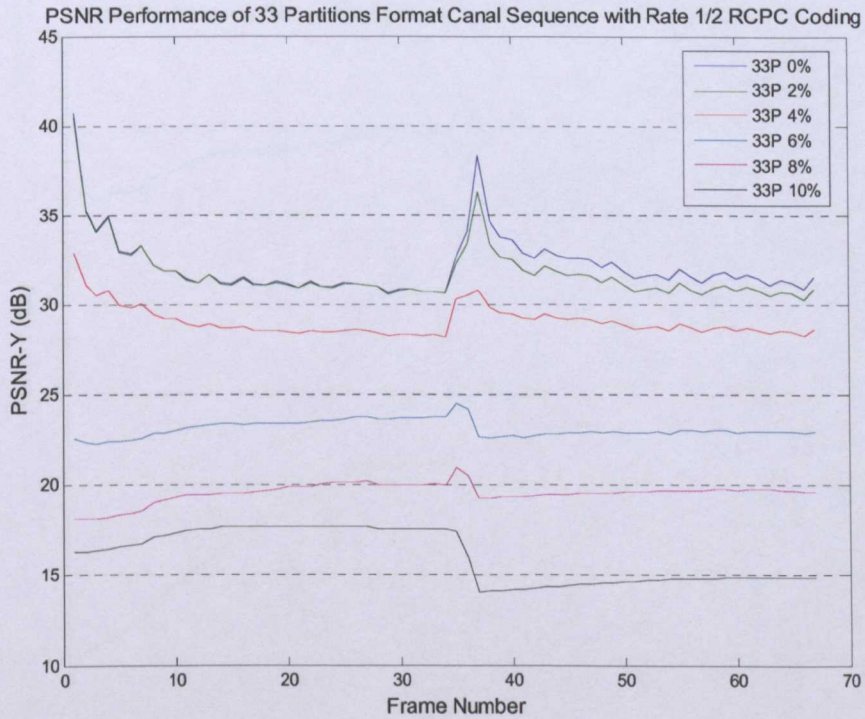


Fig. 3-15 PSNR Performance Comparisons between Different Percentages of Packet Loss for 33 Partitions Format with Rate 1/2 RCPC Coding, using Canal Sequence in CIF Format

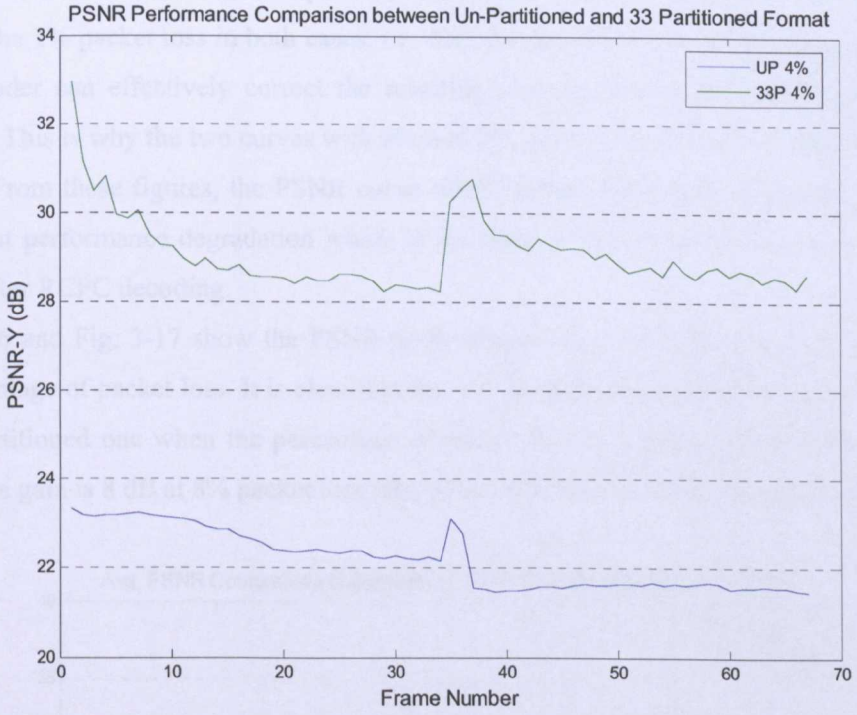


Fig. 3-16 PSNR Performance comparison between Un-Partitioned and 33 Partitioned Formats at 4% Packet Loss with Rate 1/2 RCPC, Canal Sequence in CIF Format

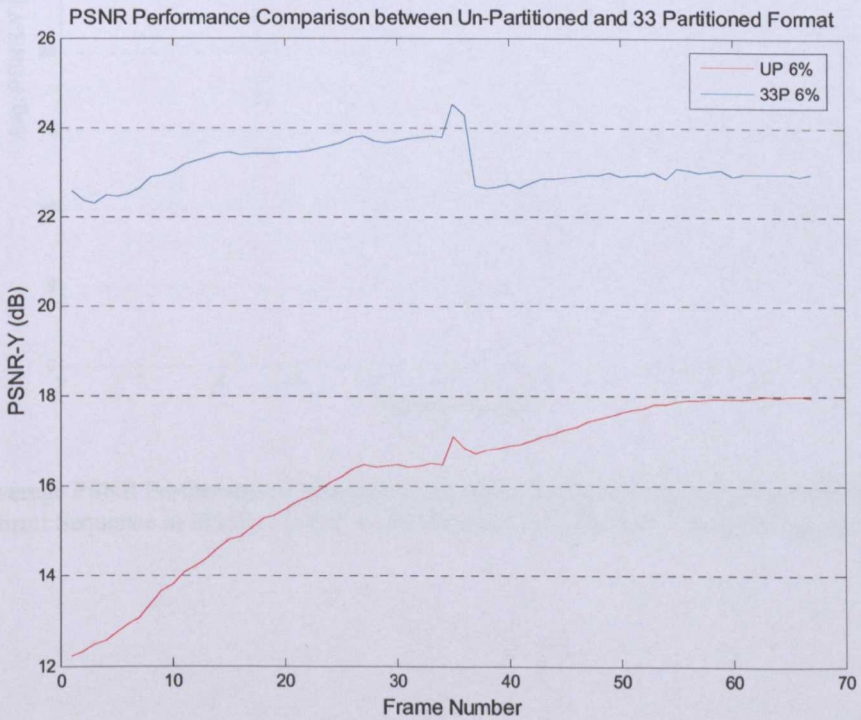


Fig. 3-17 PSNR Performance comparison between Un-Partitioned and 33P Partitioned Formats at 6% Packet Loss with Rate 1/2 RCPC, Canal Sequence in CIF Format

Fig. 3-14 and Fig. 3-15 show the PSNR performance comparison between different percentages of packet losses for both un-partitioned (without source coding) and 33P partitioned formats with rate

1/2 RCPC channel coder for Canal sequence in CIF format. There is no bit error at the RCPC decoder output for the 1% packet loss in both cases, i.e. with the use of bitwise interleaver at the encoder; the RCPC decoder can effectively correct the resulting error pattern at the output of the bitwise de-interleaver. This is why the two curves with 0% and 1% packet loss rate are overlapping each other in Fig. 3-14. From these figures, the PSNR curve of 2% packet loss is just below the error free curve with a slight performance degradation which is the result of having a few bit errors in the received sequence after RCPC decoding.

Fig. 3-16 and Fig. 3-17 show the PSNR performance comparison between two formats with the same percentage of packet loss. It is clear that the 33P partitioned format achieves at least 5 dB gains over un-partitioned one when the percentage of packet loss is 4 and 6 percents. But the maximum performance gain is 8 dB at 8% packet loss rate, which can be seen clearly in Fig. 3-12.

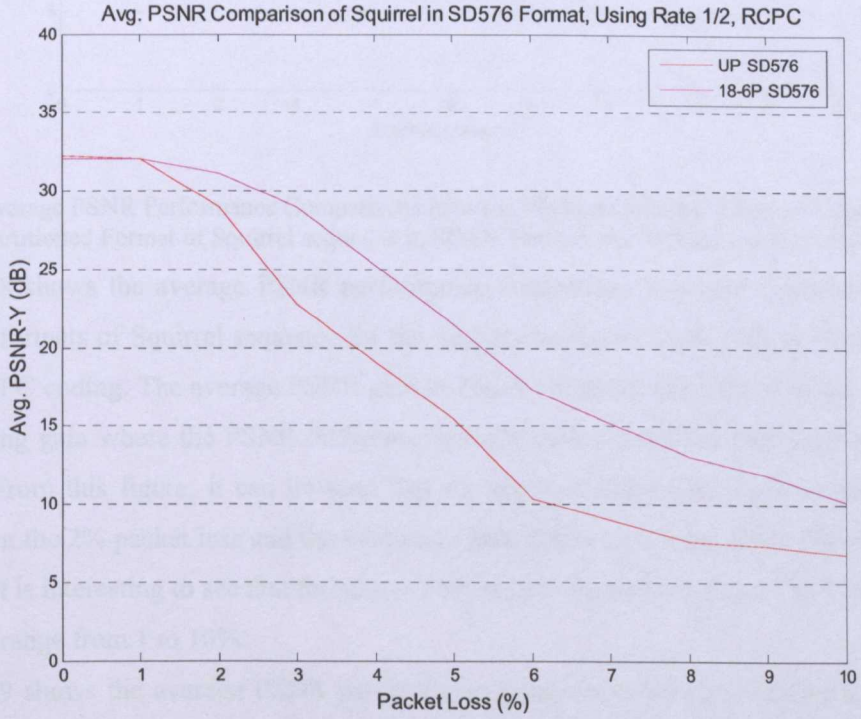


Fig. 3-18 Average PSNR Performance Comparisons between Un-Partitioned and 18-6 Partitioned Formats of Squirrel Sequence in SD576 Format for the Packet Loss from 0 to 10% with Rate 1/2 RCPC

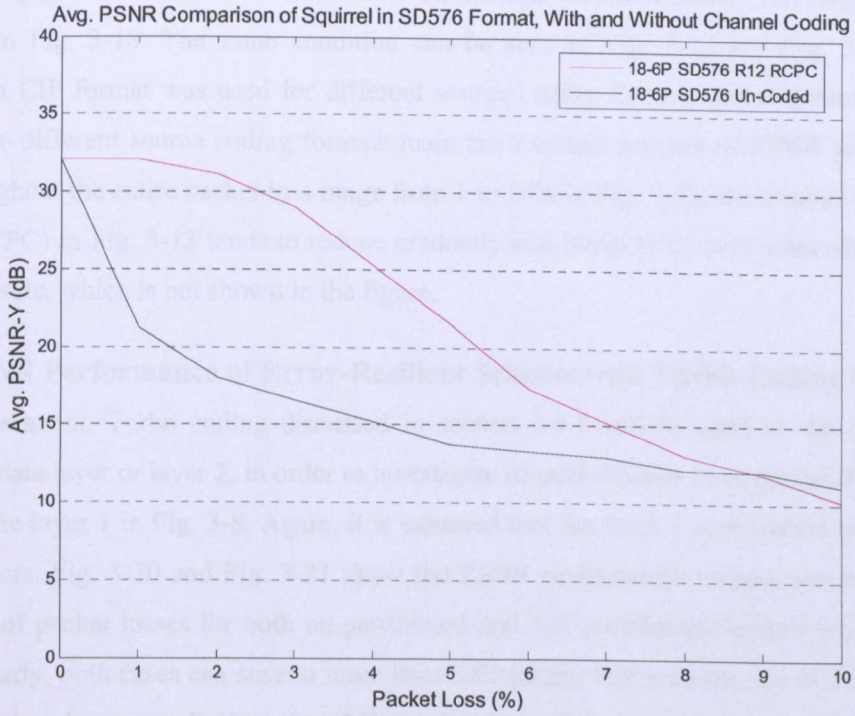


Fig. 3-19 Average PSNR Performance Comparisons between With and Without Channel Coding using 18-6P Partitioned Format of Squirrel sequence in SD576 Format, the Packet Loss from 0 to 10%

Fig. 3-18 shows the average PSNR performance comparison between un-partitioned and 18-6P partitioned formats of Squirrel sequence for the packet loss from 1% to 10% in SD576 format using rate 1/2 RCPC coding. The average PSNR gain in Fig. 3-18 can be considered as the partition gain or source coding gain where the PSNR difference actually comes from the partitioning of the encoded bitstream. From this figure, it can be seen that the source coding gain tends to increase gradually starting from the 2% packet loss and the maximum gain achieved is 7 dB at the 6% packet loss. From the figure, it is interesting to see that the source coding gain remains between 3 to 5 dB over the entire packet loss range from 1 to 10%.

Fig. 3-19 shows the average PSNR performance comparison between un-coded and coded with rate 1/2 RCPC encoding of Squirrel sequence (SD576) with 18-6P partitioned format. The average PSNR gain from Fig. 3-19 can be considered as the channel coding gain since the PSNR difference between the two curves actually comes from the application of channel coding (rate 1/2 RCPC). Unlike source coding gain in Fig. 3-18, channel coding gain increases abruptly starting from 1% packet loss and the maximum coding gain achieved is around13 dB at the 2% packet loss. Then the coding gain gradually reduces and becomes zero at about 9% packet loss rate. This means that the error correction power of the channel coding (i.e. rate 1/2 RCPC) is limited to a certain level of packet loss rate (9% in this case) and the decoder cannot correct the channel errors anymore beyond this point. After that, because of the application of bitwise interleaving, the PSNR performance of channel coded bitstream can be even lower than un-coded one. On the other hand, unlike channel coding gain,

source coding gain can maintain at least 3 to 5 dB through the entire packet loss range from 1 to 10% as shown in Fig. 3-18. The same condition can be seen in Fig. 3-12 and Fig. 3-13 when Canal sequence in CIF format was used for different source coding formats and different channel coding rates. While different source coding formats maintain a certain amount of PSNR gain between each other throughout the entire packet loss range from 1 to 10% in Fig. 3-12, the channel coding gain (e.g. rate 1/2 RCPC) in Fig. 3-13 tends to reduce gradually and likely to be zero somewhere beyond 12% packet loss rate, which is not shown in the figure.

3.6.3 PSNR Performance of Error-Resilient Scheme with Turbo Coding (TC)

In this experiment, Turbo coding discussed in section 3.4.3 will be used as the channel coder to protect the data layer or layer 2, in order to investigate its performance even though TC was presented to protect the layer 1 in Fig. 3-8. Again, it is assumed that the layer 1 is protected perfectly from the channel errors. Fig. 3-20 and Fig. 3-21 show the PSNR performance comparison between different percentage of packet losses for both un-partitioned and 33P partitioned formats with rate 1/2 Turbo coding. Clearly, both cases can sustain more than 30% packet loss with the use of TC. There is no bit error at the decoder output for less than 25% packet loss. This shows that the combined effect of the bitwise interleaver and channel encoder is much more efficient with the use of powerful channel coding mechanism.

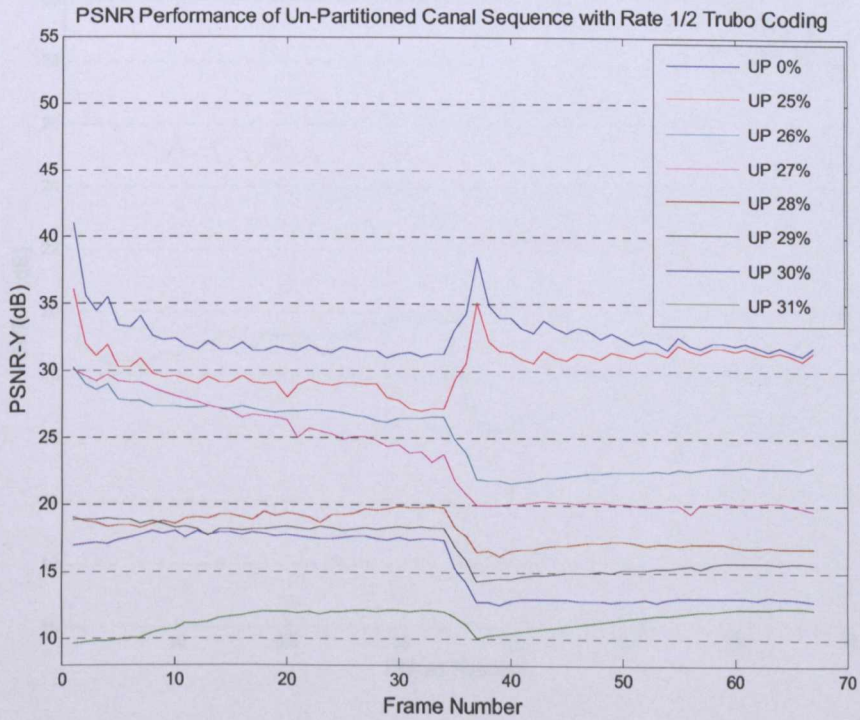


Fig. 3-20 PSNR Performance Comparisons between Different Percentages of Packet Loss for Un-Partitioned Format with Rate 1/2 Turbo Coding, Canal Sequence in CIF Format

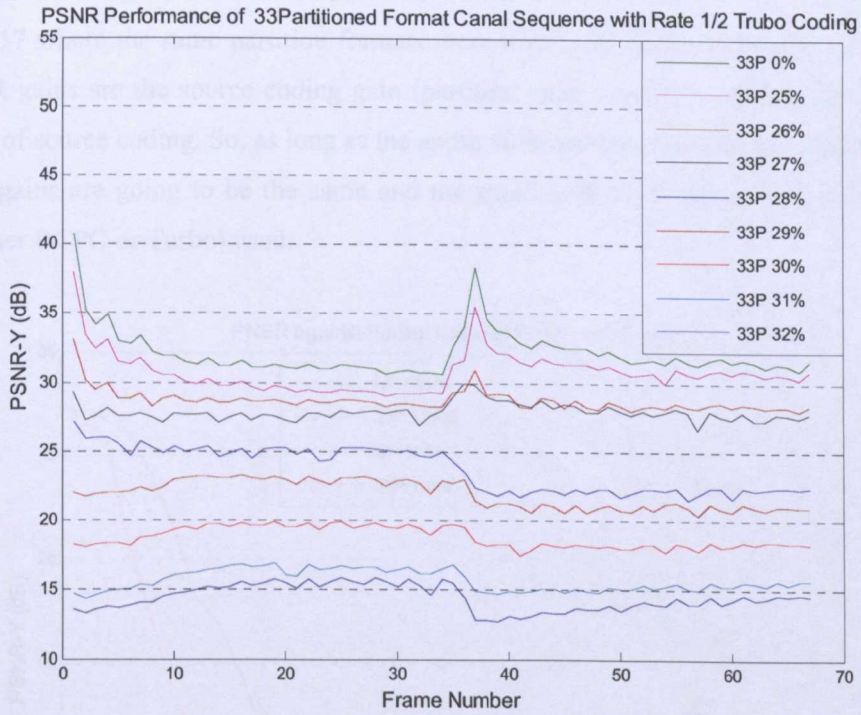


Fig. 3-21 PSNR Performance Comparisons between Different Percentages of Packet Loss for 33 Partitioned Format with Rate 1/2 Turbo Coding, using Canal Sequence in CIF Format

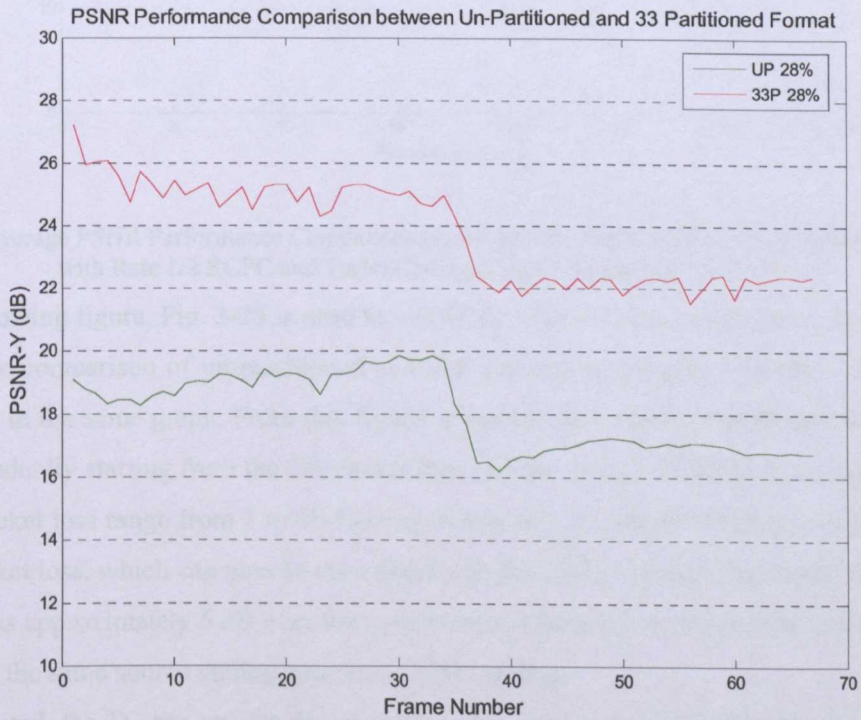


Fig. 3-22 PSNR Performance Comparison between Un-Partitioned and 33 Partitioned Formats at 28% Packet Loss with Rate 1/2 Turbo Coding, Canal Sequence in CIF Format

Fig. 3-22 shows the PSNR performance comparison between two formats shown above with the same percentage of packet loss (i.e. 28%). It is clear that the 33P partitioned with Turbo coding also

achieve at least 5 dB gains over the un-partitioned format. It is identical to the results from Fig. 3-16 and Fig. 3-17 where the same partition formats were used with RCPC coding. It is because both of these PSNR gains are the source coding gain (partition gain) where the actual gain comes from the application of source coding. So, as long as the application uses the same format of source coding, the amount of gains are going to be the same and the result will not depend upon the type of channel coding (either RCPC or Turbo) used.

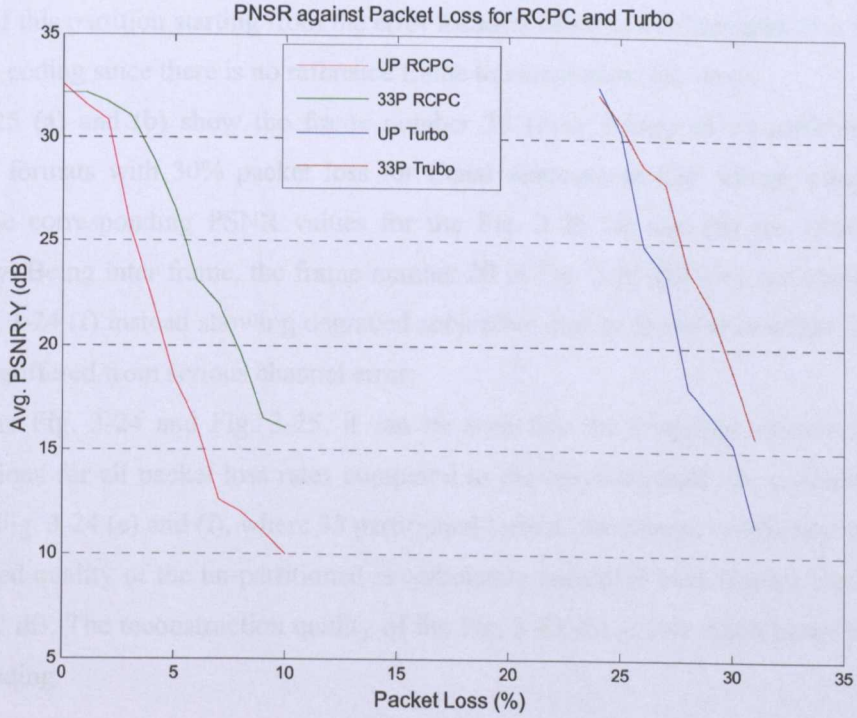


Fig. 3-23 Average PSNR Performance Comparisons between 33P Partitioned and Un-Partitioned Formats with Rate 1/2 RCPC and Turbo Coding, Canal Sequence in CIF Format

The following figure, Fig. 3-23 is used to clarify the above discussion again by drawing the PSNR performance comparison of un-partitioned and 33P partitioned formats with rate 1/2 RCPC and the rate 1/2 TC in the same graph. From this figure, it can be seen that the PSNR gain of RCPC tends to increase gradually starting from the 2% packet loss and the overall PSNR gain is approximately 5 dB over the packet loss range from 1 to 10 % even though the maximum PSNR gain achieved is 8 dB at the 8% packet loss, which can also be seen clearly on Fig. 3-12. On the other hand, the overall PSNR gain of TC is approximately 5 dB over the un-partitioned format over the packet loss range from 25 to 31% giving the same source coding gain as in RCPC coding.

As expected, the TC can protect the transmitted packet sequence much better then RCPC with the expense of higher decoding complexity and iteration delay at the receiver. Nevertheless, the advent of network-on-chip and high throughput turbo decoders would enable the application of turbo codes on the packet switched wired network [35][36].

3.6.4 Subjective Quality Assessment of the Scheme

Fig. 3-24 (a) - (f) show the frame number 37 (I-frame) for un-partitioned and the 33P partitioned formats with 2%, 4% and 6% packet loss for Canal sequence in CIF format using rate 1/2 RCPC coding. The corresponding PSNR values in Fig. 3-24 (a) - (f) are 38.45, 38.34, 20.82, 31.51, 19.45 and 22.22 dB, respectively. A vertical black patch at the lower left corner of the Fig. 3-24 (f) is the result of the loss of the whole partition. This happened when the bit error occurred in the lowest frequency subband (DC subband) of a particular bitstream or partition since the whole remaining bitstream of this partition starting from the error location needs to be discarded. It is happened only in the I-frame coding since there is no reference frame to compensate this error.

Fig. 3-25 (a) and (b) show the frame number 20 (inter frame) of un-partitioned and the 33P partitioned formats with 30% packet loss for Canal sequence in CIF format using rate 1/2 Turbo coding. The corresponding PSNR values for the Fig. 3-25 (a) and (b) are 18.92 and 24.15 dB, respectively. Being inter frame, the frame number 20 in Fig. 3-25 (b) does not show the black patch like in Fig. 3-24 (f) instead showing degraded subjective quality in the area where the corresponding partition is suffered from serious channel error.

From the Fig. 3-24 and Fig. 3-25, it can be seen that the proposed scheme offers very good reconstructions for all packet loss rates compared to the un-partitioned; for example, the 6% packet loss in the Fig. 3-24 (e) and (f), where 33 partitioned format provides an excellent protection while the reconstructed quality of the un-partitioned is completely corrupted even though the PSNR difference is only 2.77 dB. The reconstruction quality of the Fig. 3-25 (b) is also much better than Fig. 3-25 (a) in Turbo coding.

(a) Un-Partitioned Format, 2% Packet Loss

(b) 33P Partitioned Format, 2% Packet Loss

(c) Un-Partitioned Format, 4% Packet Loss

(d) 33P Partitioned Format, 4% Packet Loss

(e) Un-Partitioned Format, 6% Packet Loss

(f) 33P Partitioned Format, 6% Packet Loss

Fig. 3-24 Frame Number 37, I Frame of Canal Sequence in CIF Format with Rate 1/2 RCPC Coding

(a) Un Partitioned Format, 30% Packet Loss (b) 33P Partitioned Format, 30% packet loss

Fig. 3-25 Frame Number 20 of Canal Sequence in CIF Format with Rate 1/2 Turbo Coding

3.6.5 The Application of Error-Concealment

Fig. 3-26 shows the partition numbers and their locations in a CIF video frame having 33P partitioned format. According to that figure, the vertical black patch in the Fig. 3-24 (f) corresponds to the partition number 22. As discussed before, if the bit error is in the critical subband number (i.e. especially in low frequency subbands), the reconstructed video frame will have either a blurred vertical patch or even a totally black patch depending upon the level of subband from where the information bits were discarded because of non-correctable channel error.

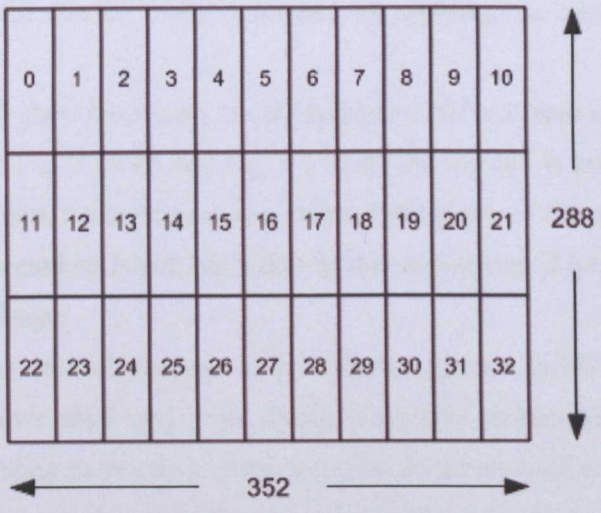
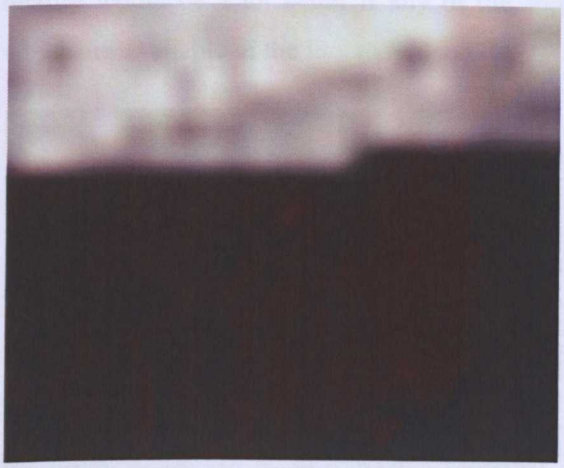
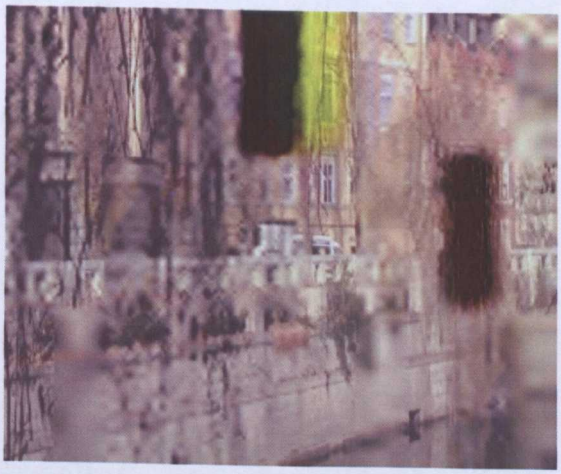


Fig. 3-26 Partitions Numbers and their Location in 33P Partitioned Format, CIF



(a) Error in Subband Number 13 Frame Number 1 (I frame), Un-Partitioned Format



(b) Error in Subband Number 13, Partition Number 5,6 and 20, Frame Number 37 (I frame), 33P Partitioned Format

Fig. 3-27 Comparison of the Effect on Reconstructed Quality of Bit Error in Subband Number 13 of I frames for both Un-Partitioned and 33P Partitioned Format, Canal Sequence in CIF Format

Fig. 3-27 (a) and (b) show the example of the loss of whole frame in un-partitioned format and some partitions in 33P partitioned format. If a bit error is occurred in the DC subband of the un-partitioned format, all the following bits equivalent to almost entire frame is required to discard and the quality of the reconstructed video is going to be un-acceptable condition as shown in Fig. 3-27 (a). However, 33P partitioned format can still maintain the acceptable video quality even though there are some losses in DC subbands of partitions 5, 6 and 20 as shown in Fig. 3-27 (b). These types of error patches in Fig. 3-27 (b) can be easily concealed by applying the appropriate error concealment method.

Fig. 3-28 (a) and (b) show the results of the application of the simple error concealment method to the video sequences in Fig. 3-24 (f) and Fig. 3-27 (b). The method is implemented by replacing the corrupted coefficients with the average values of the coefficients of the previous and next partitions. But unfortunately, this method is not applicable to the un-partitioned format since there is only one partition in one video frame.

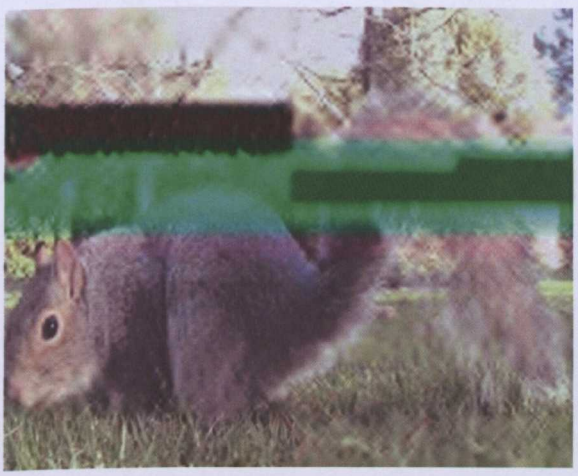
Fig. 3-29 (a) shows another example with Squirrel sequence in SD576 using18-6P partitioned format. The figure shows black and green colour horizontal patches because of some serious bits errors in the corresponding partitions at these locations in frame number 12 (*I* frame). The result of the application of error concealment method is shown in Fig. 3-29 (b). It can be seen from the figures that the subjective qualities of the resulting video become much better after applying even simple error concealment scheme. So, it is interesting to note that the application of the error concealment becomes easier since the error are localized in one partition and the loss data can be estimated either from the adjacent partitions (spatially) or the adjacent frames (temporally). The reconstructed video

quality would be a lot better if any of the sophisticated error concealment algorithms are applied, which are available in the literature.

(a) Application of Error Concealment to Fig. 3-24 (f)

(b) Application of Error Concealment to Fig. 3-27 (b)

Fig. 3-28 Application of the Simple Error Concealment Method to Fig. 3-24 (f) and Fig. 3-27 (b)



(a) Frame number 12 (I-frame) of 18-6P Partitioned Format, Squirrel in SD576 Format

(b) After Applying Error Concealment Method to (a)

Fig. 3-29 Application of Simple Error Concealment Method to *I* frame of Squirrel Sequence with SD576 Format

In the latter case, RCPC channel coding and interleaving are not applied to the transmitted bitstream. So that there will be only the message + CRC bits in each transmitted packets.

3.6.6 PSNR Performance Analysis over Burst Errors

Finally, the performance of bitwise interleaver is investigated by introducing a series of packet errors (burst error) to the transmitted packet sequence. Fig. 3-30 shows the PSNR performance comparison of introducing of uniform random and burst error to the 33P partitioned format video sequence with rate 1/2 RCPC coding using Canal sequence in CIF format. As expected, PSNR

performance degradation is occurred in the burst error compared with uniform random error and the amount of reduction is from 2 to 4 dB over the packet loss rate from 2 to 14 %. It is because the interleaver cannot break the series of packet error (burst error) effectively causing the decoder to correct longer number of error bits in series. Unfortunately, in the case of serial bits error, the number of correctable bits is limited to a certain amount at the channel decoder and any bits error longer than this limit is resulted as the bits error in decoded sequence. From this result, it is obvious to see that the choice of interleaver is also the important factor and careful selection of interleaver which can break both uniform random and a certain amount of serial error, should minimize this kind of performance reduction.

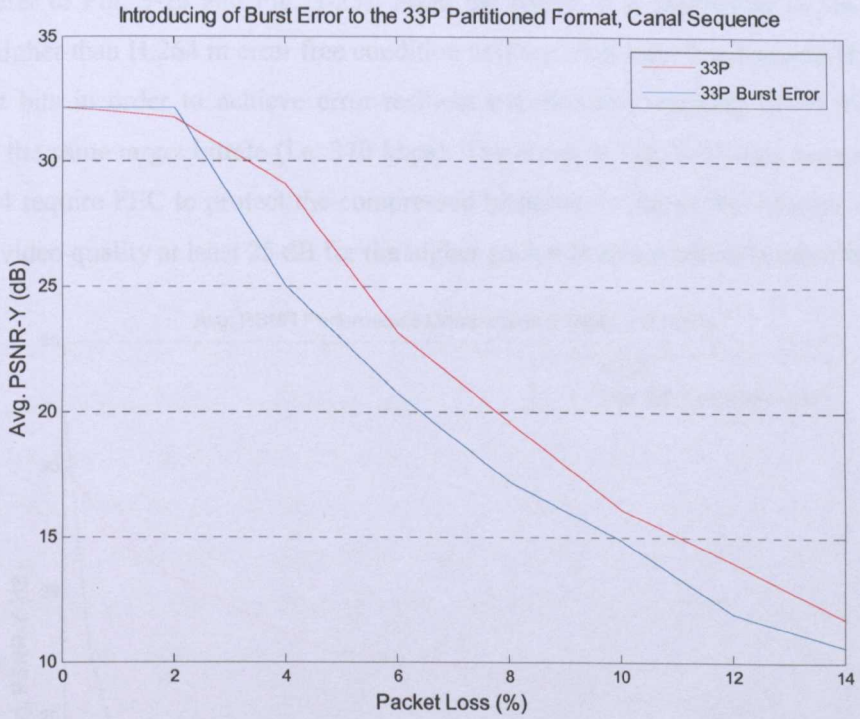


Fig. 3-30 Average PSNR Performance Comparisons between Uniform Random and Burst Error to the 33P Partitioned Formats with rate 1/2 RCPC Coding, Canal Sequence in CIF Format

3.6.7 Error-Resilient Scheme without FEC

This section analyzes the performance of the proposed error-resilient scheme of Dirac in comparison with H.264's error-resilient using Flexible Macroblock Ordering (FMO). To facilitate this experiment, H.264 (JM11 reference software) [22] is encoded in RTP packet mode and packet length is set to 30 bytes, which is approximately the same length of packet used in Dirac. H.264's rate control is used to regulate the bitrate at 370 kbps matching the bitrate generated by Dirac encoding with 33P partitioned format without FEC. In this comparison, both Dirac and H.264 are not using FEC and any sort of error concealment since the focus is to evaluate their corresponding error-resilient capability. The results of the two codecs are shown in Fig. 3-31.

As expected, Dirac with 33P partitioned format cannot survive even a low level of packet loss rate and its average PSNR falls very sharply while the H.264's PSNR decreases gradually with the increase in packet loss. This shows that the Dirac source coding (data partitioning) only is not enough for the error-resilient transmission of wavelet compressed bitstream and the requirement of FEC becomes crucial in this case. Due to the nature of wavelet transform, the coefficients from the lowest subbands are sensitive to channel error and even a few corrupted bits in these subbands, error become prominent on the reconstructed frame because of the magnifying effect that happens in the inverse wavelet transform process. The only way to protect these data is to use FEC for lower subband data and/or all subbands data in a frame in order to survive up to 30% packet loss with good video output (please refer to Fig. 3-23 and Fig. 3-25). From the figure, it is interesting to see that the PSNR of Dirac is higher than H.264 in error free condition at 0% packet loss. It is because H.264 requires more redundant bits in order to achieve error-resilient transmission resulting lower average PSNR than Dirac for the same target bitrate (i.e. 370 kbps). The result in Fig. 3-31 also suggests that both Dirac and H.264 require FEC to protect the compressed bitstream in the packet-erasure channel in order to maintain video quality at least 25 dB for the higher packet loss rate which is more than 5%.

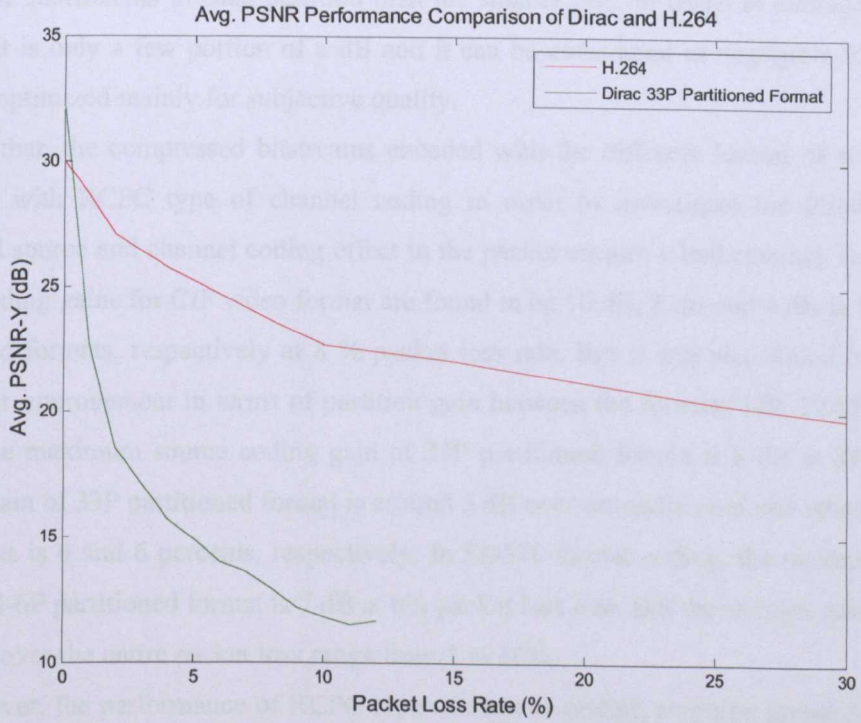


Fig. 3-31 Average PSNR Performance Comparisons between H.264 and Dirac with 33P Partitioned Format without Channel Coding using Canal Sequence in CIF Format

3.7 Chapter Summary

In this chapter, a scheme which provides the error-resilient transmission of the compressed video bitstream of Dirac video encoder over the packet erasure wired network, is presented. The scheme which is the combination of source and channel coding technique offers flexibility and simplicity to the implementation of both software and hardware. It is because the scheme can be used in any combination of source coding type (i.e. different number of partitions) and channel coding methods (i.e. either RCPC or Turbo). Moreover, the application of error-resilient source coding does not bring much complexity to the encoder. The performance of the scheme is investigated using different number of partitions and different channel coding techniques in the packet-erasure channel.

First of all, the compression efficiency of the Dirac encoder with different error-resilient source coding formats is tested before transmitted to the channel. There is a trade-off between the compression efficiency and the error-resilient performance, and found that the reduction in compression efficiency for multiple partitions format depends upon the video frame size. For the same number of partitions, larger frame size (e.g. SD576) seems to have lower compression efficiency reduction compared with the smaller video frame (e.g. CIF) because larger frame involves higher number of coefficients in each partition than the smaller one. In terms of average PSNR, there is a loss but it is only a few portion of a dB and it can be considered as negligible for a video encoder which is optimized mainly for subjective quality.

After that, the compressed bitstreams encoded with the different format of source encoding are protected with RCPC type of channel coding in order to investigate the PSNR performance of combined source and channel coding effect in the packet erasure wired channel. Resulting maximum source coding gains for CIF video format are found to be 10 dB, 8 dB and 4 dB in 99P, 33P and 6-3P partitioned formats, respectively at 8 % packet loss rate. But it was also found out that there is no significant improvement in terms of partition gain between the formats 33P, 22-9P and 22-3P. Even though the maximum source coding gain of 33P partitioned format is 8 dB at 8% packet loss rate, average gain of 33P partitioned format is around 5 dB over un-partitioned one when the percentage of packet loss is 4 and 6 percents, respectively. In SD576 format coding, the maximum source coding gain of 18-6P partitioned format is 7 dB at 6% packet loss rate. But the average gain remains between 3 to 5 dB over the entire packet loss range from 1 to 10%.

Moreover, the performance of RCPC type of channel coding was also investigated with different coding rates. The coding gains for the rate 1/2, 1/3 and 1/4 are around 4 dB, 17 dB and 20 dB, respectively at the 10% packet loss. It is interesting to find that the rate 2/3 channel coding can protect the transmitted sequence only up to 4% packet loss and its performance becomes lower than un-coded when the packet loss is more than 4% because of the application of bitwise interleaving. It was concluded that encoder rates 1/3 and 1/4 are able to be used to protect the layer 1 for less than 6% packet loss and rate 1/2 is optimum for the protection of data layer or layer 2. As for the Turbo coding, its performance was investigated by simulating the packet loss more than 30% and found that

there was no bit error at the decoder output for the packet loss less than 25%. It was also found that the source coding and channel coding gains are independent to each other and the source coding gain remains approximately the same for different types of channel coding (either RCPC or Turbo) used. The simulation results suggested that with the use of less powerful FEC e.g. RCPC, Dirac could only operate in a network congested up to 10% packet loss. Instead, using a more powerful FEC e.g. Turbo codes, Dirac could operate in a congested network up to 30% packet loss.

In terms of subjective quality, the encoding using multiple partitions sometime shows a vertical black patch because of the result of the loss of the whole partition. It is important to note that this kind of error which is mainly because of the bit error from the DC subband, can be found only in *I* frame since there is no reference frame to compensate this error. There is no such problem occurred in the inter frame since the lost data can be recovered from the reference frame. But the application of the error concealment becomes easier with the partitioned format since the errors are localized in one partition. The loss data can be estimated either from the adjacent partitions (spatially) or the adjacent frames (temporally) or using more sophisticated error concealment algorithms available in the literature. It was also found that source coding (data partitioning) only in Dirac is not enough for the error-resilient transmission of wavelet compressed bitstream. The requirement of FEC becomes crucial in this case in order to protect the sensitive information from the low frequency subbands.

As an overall, in the error-resilient coding scheme presented in this chapter, the coefficient partitioning process itself does not introduce much complexity to the encoder, and the use of FEC as a whole offer a simple and effective error-resilient scheme for Dirac to combat the harsh packet-erasure channel/network. Furthermore, the advent of network-on-chip with embedded high throughput FEC [35][36] provides the mean in applying Dirac to transport large volume of compressed video files to end-users through networked media applications e.g. on-demand video, IPTV etc.

Chapter 4

4 Rate Control Algorithm

4.1 Introduction

In real-time visual communication, having an efficient rate control algorithm at the encoder is very important to assuring the successful transmission of the coded video data. Basically, the rate control part of the encoder tries to regulate the varying bitrate characteristics of the coded bitstream by dynamically adjusting the encoder parameters in order to produce high quality decoded frame at the receiver for a given target bitrate. So that the compressed bitstream can be delivered through the available channel bandwidth without causing buffer overflow and underflow. In other words, without rate control, any video encoder would be practically hard to use for real-time end-to-end video communication.

Traditionally, rate control is not a part of the any video codec's standards but the standards group has issued non-normative guidance to aid in implementation. Therefore, the video coding standards usually recommend their own non-normative rate control schemes during the standardization process such as

- Test Model Version 5 (TM5) [37] for MPEG-2
- Test Model Near-term Version 8 (TMN8) [38] for H.263
- Verification Model Version 8 (VM8) [39] for MPEG-4 and
- Joint Video Team (JVT) [40][41] for H.264/AVC.

4.2 Related Work of the Rate Control Algorithm

Nowadays, the rate control has become one of the important research topics in the field of video compression and transmission. Many improved algorithms have also been developed over the past years. Among them, some of the algorithms for the H264 standard will be discussed here.

In [42], a novel rate control algorithm for H.264 is proposed, where bit allocation is performed on both frame level and MBs level, and the *QP* is calculated from the allocated number of bits. The algorithm gives the bitrate much closer to the target bitrate. The joint source-channel rate control strategy which considers the end-to-end distortion caused by source quantization and channel error is proposed in [43]. A novel achievement to optimum bit allocation for H.264/AVC is presented in [44], by using a simple linear rate instead of the well known MPEG-4 Q2 [39] quadratic rate model and a quadratic distortion model instead of a linear distortion model. A new rate control scheme for H.264 video encoding is proposed in [45]. In which, the inter-dependency between RDO and rate control in H.264 is resolved via *QP* estimation and update. The scheme also considers a rate model for the header information to estimate header bits more accurately since the number of bits for the header information may occupy a larger portion of the total bit budget, which is especially true when being

coded at low bit rates. However, the consideration in [42], [43], [44] and [45] focus only for the base line profile of H.264 which consists of IPPP coding and there is no bits allocation procedure for the B frames.

The mathematical model based rate control scheme for MPEG-2, which enables to predict the bits and the distortion generated from an encoded frame at a given *QP* and vice versa is proposed in [46]. Even though the scheme achieves 0.52 – 1.84 dB PSNR gain over MPEG-2 TM5 [37], the prediction error of generated bits and the distortion are still too high.

Some research has considered the coding rate and distortion as the percentage of zeros among the quantized transformed coefficients for low bitrate applications especially for H.263 in [47][48][49]. Derivation of Rate-Quantization model from the R–D function based upon the distribution of source data to be quantized is considered in [50] and [51]. It is assumed that the data to be quantized has Laplacian distribution in [50] and Generalized Gaussian distribution in [51]. However neither of these distributions is likely to occur in all types of video sequences and transformed methods.

Two-pass rate control scheme (JVT-F086) with each scheme applying a TM5 [37] based method is proposed in [52]. If the first pass fails to obtain an appropriate *QP*, the second pass is implemented as a refinement. The encoding complexity is consequently increased. Later, an improved version of [52] called partial two-pass algorithm with a linear Rate-Quantization (R-Q) model is proposed in [53]. It is based upon the jointly optimizing coding mode selection and rate control at the MB level. The algorithm's Rate-Distortion (R-D) model is utilizing the true quantization step size since the *QP* and step size are no longer linear in H.264. First of all, the predicted *QP* for the RDO based mode selection is calculated using the algorithm's R-D model. After the optimal mode selection, the estimated *QP* is refined according to the MB information and an estimated R-D cost is then calculated based on the R-D model. If the refined *QP* can reach better R-D cost than the predicted *QP*, the refined *QP* is used to do RDO again. In finding the optimum *QP*, the algorithm requires one-pass RDO process at frame level and two-pass RDO process at the MB level, which introduces unnecessary coding delay and complexity to the encoder. In [40], a one-pass RC algorithm, JVT-G012 is proposed with a linear Mean Absolute Difference (MAD) prediction model being used to circumvent the chicken-and-egg dilemma and the conventional MPEG-4 Q2 R-Q model [39] being employed to calculate *QP* given the allocated bits and predicted coding complexity. Due to its efficiency, JVT-G012 has been adopted in H.264 reference software. Unfortunately, the algorithm in [40] does not support intra frame-only coding mode which is supported by H.264 in their high profiles (e.g. High 10 Intra, High 4:2:2 Intra, Hi 4:4:4 Intra and CAVLC 4:4:4 Intra profiles).

RDO based rate control algorithm with an adaptive initial *QP* determination scheme is presented for H.264 in [54]. A linear distortion-quantization (D-Q) model is introduced first and thus a close-form solution is developed to derive optimal *QP* for encoding each MB. Then the initial *QP* is determined according to the content of video sequences. The experimental results demonstrate that the algorithm can achieve better R-D performance than that of JVT-G012 [40][41] which is the current

recommended rate control scheme implemented in the H.264 reference software (JM11). However, because of the consideration of the MB level rate control, the algorithm requires to undergo complex mathematical calculation in order to get the optimal *QP* for each and every MB introducing the significant complexity to the overall encoding process.

All the rate control techniques mentioned so far are mainly for the DCT based encoder. There is also numerous research carrying out the rate control work on the DWT based video encoders in the literature. Among them, rate control via bit allocation for each sub-band is proposed in [55] for baseline coder. Even though bit allocation is one of the fundamental approaches in controlling bit rate, distributing the bit budget among the sub-bands can be very complex and the complexity increases with the level of Wavelet Transform. In [56], methods for bit-stream allocation based on the concept of fractional bit-planes are reported for wavelet-based scalable video coding. In its low complexity method, it is assumed that the minimum rate-distortion (R–D) slope of the same fractional bit-plane within the same bit-plane across different subbands is higher than or equal to the maximum R–D slope of the next fractional bitplane. Even though the proposed idea performs very well for wavelet-based scalable video coding application, its performance for inter frame video coding were unknown since the algorithm is applied only to intra frames to compare with JPEG2000.

So, an accurate and less computationally complex rate control algorithm for DWT based encoder, which works on any video format (QCIF to HD) and any encoding format or profile either intra frame-only or inter frame (for both base line and main profile) coding becomes necessary.

4.3 The Objective of the Research

The main objective of this research is to implement the simple and efficient rate control algorithm for the Dirac video encoder [1] in order to be able to be used in real-time video broadcasting together with the error-resilient coding scheme presented in chapter 3.

The objective in video coding and transmission is to provide the best video quality at the receiver end given the constraint of certain network conditions. Given a bit budget, the best picture quality or minimum coding distortion can be achieved by optimum bit allocation and accurate rate control with minimum added complexity to the encoder.

As discussed in section 4.2, most of the rate control algorithms, e.g. in [43], [44] and [45] were focused only for the base line profile of H.264 which consists of IPPP coding. Their performance for the main profile of H.264, where B frame is also involved, remains unknown. Some of them were based upon the distribution of source data to be quantized, where the chosen distribution is less likely to occur in any types of video sequences. Others derive mathematical model with which the required *QP* is calculated in their MB level rate control strategy resulting increased computational load to the encoder since the algorithm requires to calculate *QP* for each MB using their proposed complex mathematical equations. Almost all of the rate control algorithms in DWT based video encoder use their complex bit allocation scheme to their subband level rate control causing to increase the

complexity together with the level of wavelet transform. So, it is still required to propose a rate control algorithm which is simple, efficient, computationally less expansive and suitable for wavelet transformed based Dirac video encoder.

The algorithm to be presented in this chapter is based upon the GOP level and frame level rate control only so that it is computationally less expansive since it is not required to do either subband level or MB level rate control. The algorithm is based upon the R-QF model since *QF* which is an integral parameter of Dirac encoder plays an important role in controlling the quality of the encoded video sequence or the number of bit generated. The current Dirac architecture is controlling constant quality rather than bitrate by using a single *QF* as quality indicator to maintain the desired quality. The algorithm presented in this research exploits this idea by considering *QF* as a varying parameter in order to achieve constant bitrate. It has the advantage of giving stable quality while delivering the desired constant bitrate.

4.4 Current Rate Control Algorithm of H.264 (JVT-G012)

The performance of the R-QF model based rate control algorithm for Dirac is to be compared with the rate control scheme in H.264 JM11 [40] [41] in order to justify the work since there is no such mechanism in its alpha release of Dirac. Rate control algorithm (JVT-G012) [40] of H.264 reference software (JM11) [22] consists of three tightly consecutive components: GOP level rate control, picture level rate control and the optional basic unit level rate control. The basic unit can be either a MB, a slice, or a frame and can be defined as a group of successive MBs in the same frame. All MBs in the same basic unit share a common *QP*. The choice of basic unit depends on the size of Coded Picture Buffer (CPB). In this section, an overview of the current rate control algorithm (JVT-G012) in H.264 will be discussed.

There exists a typical chicken and egg dilemma when the rate control is implemented for H.264. To perform RDO for a MB, a *QP* should be first determined for the MB by using the statistics of the current frame, which are MAD and the header bits of each MB. However, the statistics of the current frame is only available after performing the RDO. In order to overcome this, the algorithm in JVT-G012 is based upon a linear MAD prediction model to predict the MADs of the remaining basic units in the current stored picture by the actual MADs of the co-located basic units in the previous stored picture. Detail description of the algorithm can be found in [40].

The target bits for the current frame are computed by utilizing a fluid flow traffic model and linear tracking theory, and are determined by the frame rate, the current buffer occupancy, the target buffer level and the available channel bandwidth. The target bits are further bounded by two values derived by taking the Hypothetical Reference Decoder (HRD) into consideration. The remaining bits for the current frame are allocated to the remaining basic units according to their predicted MADs. A quadratic R-D model is used to compute the corresponding *QP*, which is then used to perform the

RDO for each MB in the current basic unit. The following sections will discuss the overall procedure. The detail description of the algorithm can also be found in [57].

4.4.1 A Linear MAD Prediction Model

Suppose that the j^{th} picture is a stored picture and the number of successive non-stored pictures between two stored pictures is L. The linear MAD prediction model can be written as follow in equation (4.1).

$$\tilde{\sigma}_{l,i}(j) = a_1 \times \sigma_{l,i}(j-L-1) + a_2 \tag{4.1}$$

Where, $\tilde{\sigma}_{l,i}(j)$ is the predicted MAD of the l^{th} basic unit in the current stored picture in the i^{th} GOP. $\sigma_{l,i}(j-L-1)$ is the actual MAD of the l^{th} basic unit in the previous stored picture. a_1 and a_2 are two coefficients of the prediction model. The initial values of a_1 and a_2 are set to 1 and 0, respectively. They are updated after coding each basic unit. It should be mentioned that $\sigma_{l,i}(j-L-1)$ is a reference value for the prediction.

4.4.2 GOP Level Rate Control

In this level, the total number of bits allocated to each GOP is computed and the initial QP of each GOP is set. When the j^{th} picture in the i^{th} GOP is coded, the total bits for the rest of the pictures in this GOP are computed as follows.

$$B_i(j) = \begin{cases} \dfrac{R_i(j)}{f} \times N_i - V_i(j) & j = 1 \\ B_i(j-1) + \dfrac{R_i(j) - R_i(j-1)}{f} \times (N_i - j + 1) - b_i(j-1) & j = 2,3,\ldots,N_i \end{cases} \tag{4.2}$$

For the first picture in a GOP (i.e. j = 1), the total bits are calculated from the upper formula in equation (4.2). f is the predefined coding frame rate. N_i is the total number of pictures in the i^{th} GOP. $R_i(j)$ and $V_i(j)$ are the instant available bit rate and the occupancy of the virtual buffer, respectively, when the j^{th} picture in the i^{th} GOP is coded.

For other pictures, the total bits are calculated from the bottom formula in equation (4.2). $b_i(j\text{-}1)$ is the actual generated bits in the $(j\text{-}1)^{th}$ picture. Considering the case of the dynamic channels, $R_i(j)$ may vary at different frames and GOPs. But, in the case of constant bit rate, $R_i(j)$ is always equal to $R_i(j\text{-}1)$. The formula can be simplified as shown in equation (4.3).

$$B_i(j) = B_i(j-1) - b_i(j-1) \tag{4.3}$$

The initial QP of the first GOP, $QP_1(1)$ is predefined based on the available channel bandwidth and the GOP length. The Instantaneous Decoding Refresh (IDR) picture and the first stored picture of the GOP are encoded by $QP_i(1)$. For the other GOPs, $QP_i(1)$ is computed by

$$QP_i(1) = \max\left\{QP_{i-1}(1) - 2, \min\left\{QP_{i-1}(1) + 2, \frac{SumPQP(i-1)}{N_p(i-1)} - \min\left\{2, \frac{N_{i-1}}{15}\right\}\right\}\right\} \quad (4.4)$$

$N_P(i\text{-}1)$ is the total number of stored pictures in the $(i\text{-}1)^{th}$ GOP, and $SumPQP(i\text{-}1)$ is the sum of average picture QPs for all stored pictures in the $(i\text{-}1)^{th}$ GOP. It's further adjusted by

$$QP_i(1) = QP_i(1) - 1 \quad if \quad QP_i(1) > QP_{i-1}(N_{i-1} - L) - 2 \quad (4.5)$$

$QP_{i-1}(N_{i-1} - L)$ is the QP of the last stored picture in the previous GOP, and L is the number of successive non-stored pictures between two stored pictures. Clearly, $QP_i(1)$ is adaptive to both the GOP length and the available channel bandwidth.

4.4.3 Frame Level Rate Control

In this level, the target bits and QPs for stored pictures and QPs for non-stored pictures are computed.

4.4.3.1 Quantization Parameters Calculation of Non-stored Pictures

The QPs of non-stored pictures are obtained through a linear interpolation method as follows: Suppose that the j^{th} and the $(j+L+1)^{th}$ frames are stored pictures and the QPs of these stored pictures are $QP_i(j)$ and $QP_i(j+L+1)$, respectively. The QP of the i^{th} $(1 \le i \le L)$ non-stored picture is given according to the following two cases:

Case 1: When $L = 1$, there is only one non-stored picture between two stored pictures. The QP is computed by

$$QP_i(j+1) = \begin{cases} \dfrac{QP_i(j) + QP_i(j+2) + 2}{2} & \text{if } QP_i(j) \neq QP_i(j+2) \\ QP_i(j) + 2 & \text{Otherwise} \end{cases} \quad (4.6)$$

Case 2: when $L > 1$, there are more than one non-stored picture between two stored pictures. The QPs are computed by

$$QP_i(j+k) = QP_i(j) + \alpha + \max\left\{\min\left\{\frac{(QP_i(j+L+1) - QP_i(j)}{L-1}, 2\times(k-1)\right\}, -2\times(k-1)\right\} \quad (4.7)$$

where $k = 1, \ldots, L$, and α is given by

$$\alpha = \begin{cases} -3 & QP_i(j+L+1) - QP_i(j) \le -2\times L - 3 \\ -2 & QP_i(j+L+1) - QP_i(j) = -2\times L - 2 \\ -1 & QP_i(j+L+1) - QP_i(j) = -2\times L - 1 \\ 0 & QP_i(j+L+1) - QP_i(j) = -2\times L \\ 1 & \mathrm{QP_i}(j+L+1) - QP_i(j) = -2\times L + 1 \\ 2 & \text{Otherwise} \end{cases} \tag{4.8}$$

The quantization parameter, $QP_i(j+k)$ is further bounded by 0 and 51.

4.4.3.2 Quantization Parameters Calculation of Stored Pictures

The QP of stored picture is computed by first determining the target bits for each stored picture and then computes the QP to perform RDO.

Step 1: Determining the target bits for each stored picture.

In this step it is required to determine the target buffer level first. A target buffer level is predefined for each stored picture according to the coded bits of the first IDR picture and the first stored picture, and the average picture complexity. After coding the first stored picture in the i^{th} GOP, the initial value of target buffer level is set to

$$S_i(2) = V_i(2). \tag{4.9}$$

The target buffer level for the subsequent stored picture is determined by

$$S_i(j+1) = S_i(j) - \frac{S_i(2)}{N_p(i)-1} + \frac{\overline{W}_{p,i}(j)\times(L+1)\times R_i(j)}{f\times(\overline{W}_{p,i}(j)+\overline{W}_{b,i}(j)\times L)} - \frac{R_i(j)}{f} \tag{4.10}$$

where $\overline{W}_{p,i}(j)$ is the average complexity weight of stored pictures, $\overline{W}_{b,i}(j)$ is the average complexity weight of non-stored pictures. They are computed by

$$\begin{aligned} \overline{W}_{p,i}(j) &= \frac{W_{p,i}(j)}{8} + \frac{7\times\overline{W}_{p,i}(j-1)}{8} \\ \overline{W}_{b,i}(j) &= \frac{W_{b,i}(j)}{8} + \frac{7\times\overline{W}_{b,i}(j-1)}{8} \\ W_{p,i}(j) &= b_i(j)\times QP_{p,i}(j) \\ W_{b,i}(j) &= \frac{b_i(j)\times QP_{b,i}(j)}{1.3636} \end{aligned} \tag{4.11}$$

When there is no non-stored picture between two stored pictures, equation (4.10) is simplified as

$$S_i(j+1) = S_i(j) - \frac{S_i(2)}{N_p(i)-1}. \tag{4.12}$$

After getting the target buffer level, the target bits can be determined as follow. The target bits allocated for the j^{th} stored picture in the i^{th} GOP are determined based on the target buffer level, the frame rate, the available channel bandwidth, and the actual buffer occupancy of as follows:

$$\tilde{T}_i(j) = \frac{R_i(j)}{f} + \gamma \times (S_i(j) - V_i(j)) \tag{4.13}$$

where γ is a constant and its typical value is 0.5 when there is no non-stored picture and 0.25 otherwise. Meanwhile, the number of remaining bits should also be considered when the target bit is computed.

$$\hat{T}_i(j) = \frac{W_{p,i}(j-1) \times B_i(j)}{W_{p,i}(j-1) \times N_{p,r} + W_{b,i}(j-1) \times N_{b,r}} \tag{4.14}$$

Where $N_{p,r}$ and $N_{b,r}$ are the number of the remaining stored pictures and the number of the remaining non-stored pictures, respectively.

The target bits are a weighted combination of $\tilde{T}_i(j)$ and $\hat{T}_i(j)$:

$$T_i(j) = \beta \times \hat{T}_i(j) + (1-\beta) \times \tilde{T}_i(j) \tag{4.15}$$

where β is a constant and its typical value is 0.5 when there is no non-stored picture and is 0.9 otherwise.

Step 2: Computation of the *QP* and perform RDO.

The MAD of the current stored picture, $\tilde{\sigma}_i(j)$, is predicted by a linear model given in equation (4.1) using the actual MAD of the previous stored picture, $\sigma_i(j-1-L)$. The quantization step corresponding to the target bits is then computed by using the following quadratic:

$$T_i(j) = c_1 \times \frac{\tilde{\sigma}_i(j)}{Q_{step,i}(j)} + c_2 \times \frac{\tilde{\sigma}_i(j)}{Q^2_{step,i}(j)} - m_{h,i}(j) \tag{4.16}$$

where $m_{h,i}(j)$ is the total number of header bits and MV bits, c_1 and c_2 are two coefficients. The corresponding quantization parameter $QP_i(j)$ is computed by using the relationship between the quantization step and the *QP* of AVC.

After encoding a picture, the parameters a_1 and a_2 of linear prediction model given in equation (4.1), as well as c_1 and c_2 of quadratic R-D model given in equation (4.16) are updated. A linear regression method similar to MPEG-4 Q2 [39] is used to update these parameters. The actual bits generated are added to the buffer. To ensure that the updated buffer occupancy is not too high; a number of pictures may be skipped by using the method similar to MPEG-4 Q2 [39].

4.4.4 Basic Unit Level Rate Control

Same as the frame layer rate control, the *QPs* for IDR picture and non-stored pictures are the same for all basic units in the same picture. It is computed the similar way as that at frame layer provided that $QP_i(j)$ and $QP_i(j+L+1)$ are replaced by the average values of *QPs* of all basic units in the corresponding picture. The basic unit level rate control selects the values of *QPs* of all basic units in a stored picture, so that the sum of generated bits is close to the frame target bits, $T_i(j)$.

The following is a step-by-step description of this method.

Step 1 Predict the MADs, $\tilde{\sigma}_{l,i}(j)$, of the remaining basic units in the current stored picture by model given in equation (4.1) using the actual MADs of the co-located basic units in previous stored picture.

Step 2 Compute the number of texture bits $\hat{b}_l$ for the l^{th} basic unit. This step is composed of the following three sub-steps:

Step 2.1 Compute the target bits for the l^{th} basic unit.

Let T_r denote the number of remaining bits for the current frame and its initial value is set to $T_i(j)$. The target bits for the l^{th} basic unit are given by

$$\tilde{b}_l = T_r \times \frac{\tilde{\sigma}_{l,i}^2(l)}{\sum_{k=l}^{N_{unit}} \tilde{\sigma}_{k,i}^2(j)} \tag{4.17}$$

Step 2.2 Compute the average number of header bits generated by all coded basic units:

$$\begin{aligned} \tilde{m}_{hdr,l} &= \tilde{m}_{hdr,l-1} \times (1-\frac{1}{l}) + \frac{\hat{m}_{hdr,l}}{l} \\ m_{hdr,l} &= \tilde{m}_{hdr,l} \times \frac{l}{N_{unit}} + m_{hdr,1} \times (1-\frac{l}{N_{unit}}); 1 \le l \le N_{unit} \end{aligned} \tag{4.18}$$

where $\hat{m}_{hdr,l}$ is the actual number of header bits generated by the l^{th} basic unit in the current stored picture. $m_{hdr,1}$ is the estimation from all basic units in the previous stored picture.

Step 2.3 Compute the number of texture bits $\hat{b}_l$ for the l^{th} basic unit.

$$\hat{b}_l = \tilde{b}_l - m_{hdr,l} \tag{4.19}$$

Step 3 Compute the quantization step for the l^{th} basic unit of j^{th} picture in i^{th} GOP by using the quadratic R-D model given in equation (4.16) and it is converted to the corresponding quantization parameter $QP_{l,i}(j)$ by using the method provided by AVC. The following three cases is needed to consider:

Case 1 The first basic unit in the current frame.

$$QP_{1,i}(j) = \overline{QP}_i(j-L-1) \tag{4.20}$$

Where, $\overline{QP}_i(j-L-1)$ is the average value of QPs for all basic units in the previous stored picture.

Case 2 When the number of remaining bits is less than 0, the QP should be greater than that of previous basic unit such that the sum of generated bits is close to the target bits, i.e.

$$QP_{l,i}(j) = QP_{l-1,i}(j) + \Delta_{Bu} \tag{4.21}$$

where Δ_{Bu} is the varying range of QP along basic units, the initial value of Δ_{Bu} is 1 if N_{unit} is greater than 8, and 2 otherwise. It is updated after coding each basic unit as follows:

$$\Delta_{Bu} = \begin{cases} 1; & \text{if } QP_{l-1,i}(j) > 25 \\ 2; & \text{otherwise} \end{cases} \tag{4.22}$$

To maintain the smoothness of perceptual quality, the QP is further bounded by

$$QP_{l,i}(j) = \max\{0, \overline{QP}_i(j-L-1) - \Delta_{Fr}, \min\{51, \overline{QP}_i(j-L-1) + \Delta_{Fr}, QP_{l,i}(j)\}\} \tag{4.23}$$

where Δ_{Fr} is the varying range of QP along frames, and is defined by

$$\Delta_{Fr} = \begin{cases} 2; & \text{if } N_{unit} > 18 \\ 4; & \text{if } 18 \geq N_{unit} > 9 \\ 6; & \text{otherwise} \end{cases} \tag{4.24}$$

Case 3 Otherwise, a quantization step is first computed by using the quadratic model given in equation (4.16), and convert it into the corresponding quantization parameter $QP_{l,i}(j)$. Similar to case 2, it is bounded by

$$QP_{l,i}(j) = \max\{QP_{l-1,i}(j) - \Delta_{Bu}, \min\{QP_{l,i}(j), QP_{l-1,i}(j) + \Delta_{Bu}\}\} \tag{4.25}$$

Step 4 Perform RDO for all MBs in the current basic unit and code them by AVC.

Step 5 Update the number of remaining bits, the coefficients of the linear prediction model given in equation (4.1), and those of the quadratic R-D model given in equation (4.16).

4.5 The Rate Control Algorithm for Dirac Codec

As mentioned in section 4.3, the current Dirac architecture is controlling constant quality rather than bitrate by using a user defined parameter, QF as quality indicator to maintain the desired quality. Since QF plays an important role in controlling the quality of the encoded video sequence or the number of bit generated in the encoding process of Dirac video codec, finding the optimum QF for a given set of target rate and video test sequence could lead to an algorithm which control the rate of the

encoder. As a consequence of the random nature of the contents of the video sequences, the complexity of the each and every frame in the sequence could be changing all the time. So, it is practically impossible to use constant *QF* to encode the entire video sequence to achieve the constant bit rate over a GOP because optimum *QF* to achieve the desire target bit rate for a previous frame would not be optimum for the current and the following frames. However, bit rate controlling over a GOP could be possible by adaptively changing the *QF* of each and every frame according to their complexities using a certain type of model before encoding. The algorithm to be presented here exploits this idea by considering *QF* as a varying parameter in order to achieve average bitrate which is constant over each GOP. Based upon this, a relationship between the bitrate, *R* and the *QF*, which can be used to estimate the *QF* for a given target bitrate, is derived. This model is known as R-QF model.

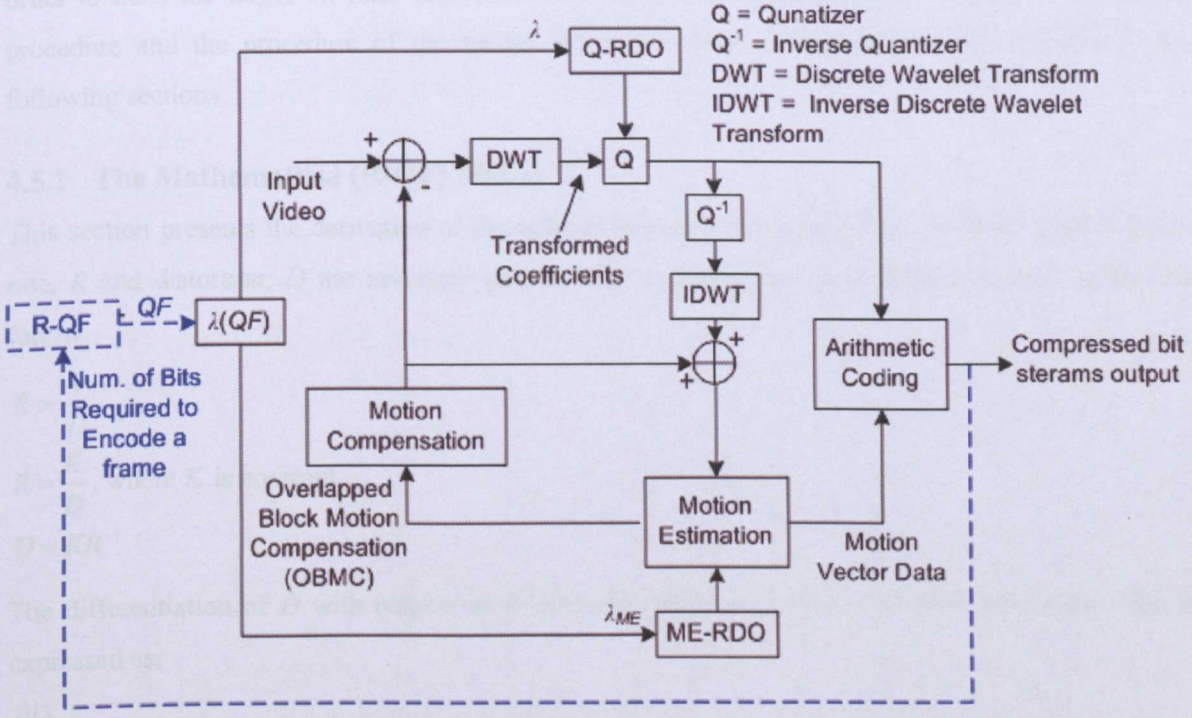


Fig. 4-1The block diagram of Dirac Encoder [4] Showing R-QF Model based Rate Control Idea in Blue Colour

Fig. 4-1 shows the overall block diagram of Dirac encoder from [4] showing R-QF Model based rate control idea with the blue colour. Using the generated number of bits required to encode a frame as the feedback parameter (bit rate, *R*), R-QF model adaptively calculates the optimum *QF* to encode the following frames in order to achieve the target bitrate. Given the value of *QF*, λ is calculated using the equation (2.3) in the next block, $\lambda(QF)$. Both λ_{ME} and λ are used in the RDO process of motion estimation and quantization as Lagrangian multiplier. λ_{ME} is the scaled version of λ and it is equal to two times the value of λ in Dirac reference software version 0.6[3].

The target bitrate is considered as the average bitrate over a GOP and the optimum bitrates contributed from the different types of individual frame (i.e. *I*, L_1 and L_2) in order to meet the target

bitrate still need to be calculated. In order to do this, a modified version of TM5 bit allocation procedure for MPEG-2 rate control [37] is employed, so that calculation of bitrate contributed from different types of individual frames becomes possible by using the allocated bits to each frame type and the overall frame rate. The rate control procedure in TM5 works in three steps. First of all, target bit allocation estimates the number of bits available to code the next picture. It is performed before coding the picture. In the next step, by means of a virtual buffer, the reference value of the *QP* for each MB is set. In the final step, adaptive quantization is carried out by modulating the reference value of the *QP* from the previous step according to the spatial activity in the MB to derive the final *QP* which is used to quantize the MB. The detail procedure TM5 can be found in [37]. In the R-QF model based rate control method, only the modified version of TM5's step one which is the target bit allocation is used to estimate the optimum number of bits required from the different types of frame in order to meet the target bit rate. The derivation of the mathematical (R-QF) model, bit allocation procedure and the procedure of the model based rate control algorithm will be detailed in the following sections.

4.5.1 The Mathematical (R-QF) Model

This section presents the derivation of the relation between Rate and *QF* in the R-QF model. Since rate, R and distortion, D are inversely proportional to each other, their relation can be written as follow.

$$R \propto \frac{1}{D}$$

$$R = \frac{K}{D}, \text{ where } K \text{ is constant.}$$

$$D = KR^{-1}$$

The differentiation of D with respect to R gives the slope of the Rate vs. Distortion curve that is expressed as:

$$\frac{\partial D}{\partial R} = \lambda = -KR^{-2}$$

where, λ is the slope of the Rate vs. Distortion curve or the Lagrangian parameter of RDO motion estimation mode decision and quantization where optimum *QP* selection is carried out.

$$\lambda = \frac{K}{R^2}, \text{ neglect the negative sign} \qquad (4.26)$$

Substituting the value of λ from equation (2.3) to (4.26) gives:

$$\frac{10^{(10-QF)/2.5}}{16} = \frac{K}{R^2}$$

$$\frac{2}{5}(10-QF) = \log_{10}\left(\frac{16K}{R^2}\right)$$

$$10-QF = \frac{5}{2}\log_{10}\left(\frac{16K}{R^2}\right)$$

$$QF = 10 - \frac{5}{2}\log_{10}\left(\frac{16K}{R^2}\right) \tag{4.27}$$

Equation (4.27) gives the relation between Rate, R and QF. The accuracy of the calculated QF in equation (4.27) can be verified by the practical value captured from the encoding of the "Canal vertical pan street sequence" or in short form "Canal sequence" in CIF which can be downloaded from [1], as shown in Fig. 4-2. For practical result, Canal sequence is encoded with different QF (from 1 to 10) without enabling the rate control. The resulting bit rates which are the average bit rate over entire sequence and their corresponding QF are drawn as the practical curve in Fig. 4-2. For theoretical curve, the required K value (which is 59458 in Fig. 4-2) can be calculated by substituting a set of bit rate and QF (in Fig. 4-2, a set of bit rate and QF when QF = 7 was used) from practical results in equation (4.27). After getting the value of K, the required bit rate in Kbps for theoretical curve can be calculated by changing the QF value from 1 to 10 in equation (4.27). It can be seen that the R-QF model given in equation (4.27) has a very accurate approximation to the practical results.

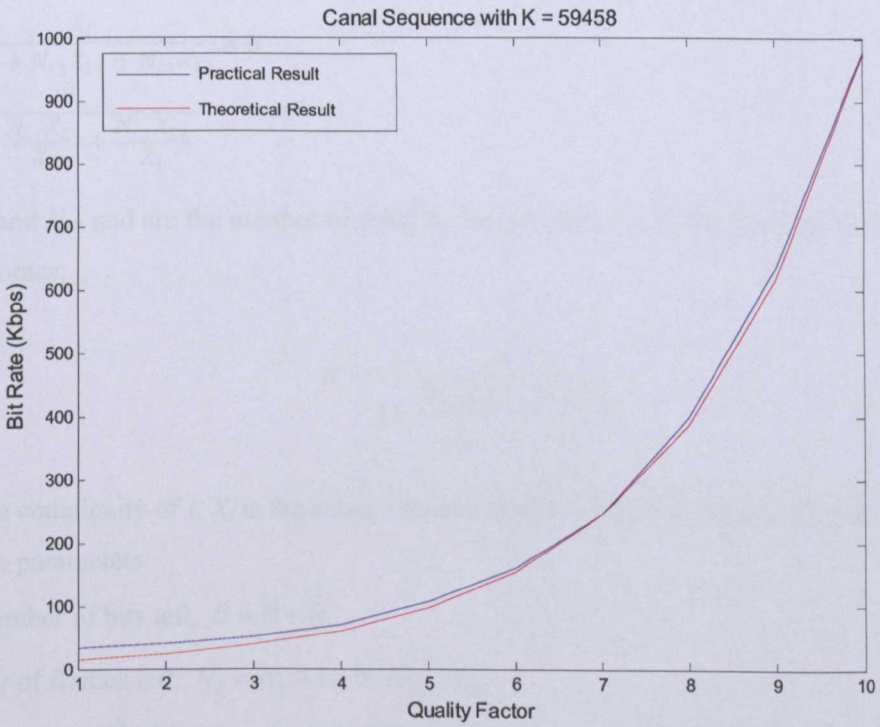


Fig. 4-2 The Approximation of the Rate and QF Relation with the R-QF Model

4.5.2 Bit Allocation Procedure

The bit allocation method used here is the modification of TM5's target bits allocation procedure for MPEG-2 rate control [37]. The complexity of each frame types is initialized as follows:

X_I = Num. of bit generated from the first I frame coding

X_{L1} = Num. of bit generated from the first L_1 frame coding

X_{L2} = Average number of bit generated from the first two L_2 frames coding (for $L_1L_2L_2L_1$ GOP structure)

where X_I, X_{L1} and X_{L2} are the complexities of I, L_1 and L_2 frames, respectively. Here, the number of bits generated from encoding a frame is considered as the complexity because the complexity of a frame is more or less related with the number of bits generated from encoding this frame if all the parameters are kept constant.

The number of frames in a GOP can be calculated as follows:

$$GOP_{Len.} = (N_{L1} + 1) \times L_{1Sep.}$$

where N_{L1} is the number of L_1 frames and L_{1Sep} is the L_1 frame separation. Furthermore, the number of bits allocated to a GOP is calculated as:

$$B = \frac{R_T \times GOP_{Len.}}{Frame\ Rate}$$

where, R_T is the target bitrate in bits per second.

4.5.2.1 *I* Frame Bit Allocation

For coding ratio $X_I : X_{L1} : X_{L2}$, the number of bits allocated for the first I frame can be calculated as:

$$B_I = \frac{X_I}{N_I X_I + N_{L1} X_{L1} + N_{L2} X_{L2}} \times B$$

$$B_I = \frac{B}{N_I + \frac{N_{L1} X_{L1}}{X_I} + \frac{N_{L2} X_{L2}}{X_I}}$$

where N_I and N_{L2} and are the number of I and L_2 frames. For $N_I = 1$, the number of bits allocated for I frame becomes:

$$B_I = \frac{B}{1 + \frac{N_{L1} X_{L1}}{X_I} + \frac{N_{L2} X_{L2}}{X_I}}. \tag{4.28}$$

Where, the complexity of I, X_I is the actual number of bits required to encode I frame.

Update the parameters:

i) Total number of bits left, $B = B - B_I$

ii) Number of frames left, $N_I = N_I - 1 = 0$, N_{L1}, N_{L2}

Total number of bits left is calculated based upon the result of equation (4.28) which is B_I since it was calculated based upon the coding or complexity ratio which can be considered as the more stable

source. On the other hand, updating the total number of bits using X_I instead of B_I could introduce un-stability especially when the QF used to encode the frame is not the optimum one (too large or too small). For example, if QF used to encode the frame is too large (it can happen especially in the encoding of the first I frame in the sequence since the initial value of QF is not the optimum one for the set target bit rate), encoder would generate larger information bits and hence the value of X_I would be too high resulting the un-acceptable number of bits left for the remaining frames in the current GOP after updating. The same idea applies for the updating of number of bits left for other frame types, L_1 and L_2.

4.5.2.2 L_1 Frame Bit Allocation

The number of bits allocated for the first L_1 frame can be calculated as:

$$B_{L1} = \frac{X_{L1}}{N_I X_I + N_{L1} X_{L1} + N_{L2} X_{L2}} \times B$$

$$B_{L1} = \frac{B}{\frac{N_I X_I}{X_{L1}} + N_{L1} + \frac{N_{L2} X_{L2}}{X_{L1}}}$$

Since there is only one I frame in a GOP, for $N_I = 0$, the number of bits allocated for L_1 frame becomes:

$$B_{L1} = \frac{B}{N_{L1} + \frac{N_{L2} X_{L2}}{X_{L1}}}. \tag{4.29}$$

Where, the complexity of L_I, X_{LI} is the actual number of bits required to encode L_I frame.

Update the parameters:

i) Total number of bits left, $B = B - B_{L1}$

ii) Number of frames left, $N_I = 0$, $N_{L1} = N_{L1} - 1$, N_{L2}

4.5.2.3 L_2 Frame Bit Allocation

Similarly, the number of bits allocated for the first L_2 frame can be calculated as:

$$B_{L2} = \frac{B}{N_{L2} + \frac{N_{L1} X_{L1}}{X_{L2}}}. \tag{4.30}$$

Where, the complexity of L_2, X_{L2} is the average of the actual number of bits required to encode two L_2 frames.

Update the parameters:

i) Total number of bits left, $B = B - B_{L2}$

ii) Number of frames left, $N_I = 0,\ N_{L1}, N_{L2} = N_{L2} - 1$

iii) The number of bits allocated for 2nd L_2 frame is calculated using equation (4.30) and again update B and N_{L2}

4.5.3 The Operation of R-QF Rate Control

4.5.3.1 Intra Frame-Only Coding

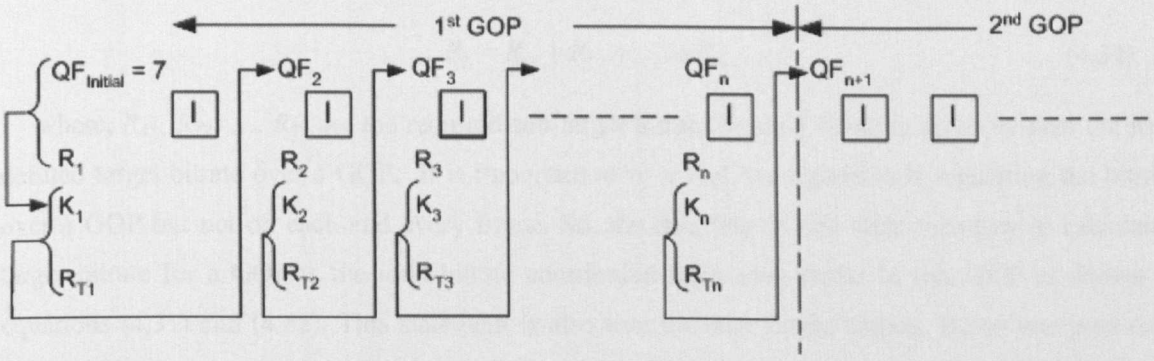


Fig. 4-3 Rate Control Procedure for Intra Frame-Only Coding

In Fig. 4-3, the first I frame is encoded by using the initial QF which is set to 7 (medium quality) and B_1 which is the number of bits generated from encoding this I frame is recorded. R_1 is calculated by using the number of bits required to encode the first I frame (B_1), frame rate and GOP length as follow.

$$R_1 = \frac{B_1 \times Frame\ Rate}{GOP_{Len.}}$$

After getting R_1, K_1 can be calculated by substituting QF_{Initial} and R_1 in equation (4.27). For I frame-only coding, the number of bits expected to be generated from encoding of each I frame in a GOP in order to meet the target bitrate is the division of the total number of bits allocated for a GOP (B) with the GOP length (N_I). So, the optimum number of bits required to encode the first I frame, B_{T1} to achieve the target bitrates can be calculated as follow.

$$B_{T1} = \frac{B}{N_I}$$

Equation (4.28) in I frame bit allocation process is the extension of the above equation for inter frame coding where encoding involves multiple frame types (I, L_1 and L_2). R_{T1} can be calculated by multiplying B_{T1} with frame rate and divided by length of GOP as before and it is constant for I frame only coding, i.e. $R_{T1} = R_{T2} = R_{T3} = \ldots$ etc. It can be considered as the required sub-target bitrate contributed from one I frame in order to meet the user defined target bitrate which is calculated over a GOP. From K_1 and R_{T1}, the optimum QF for the first I frame, which will be used to encode the next successive I frame as QF_2 can be calculated by using equation (4.27). The same procedure continues until the end of sequence. The resulting bitrate associated to a GOP (e.g. the first GOP in equation (4.31)) which has n number of I frames is equal to

$$R_{1^{st}\,GOP} = R_1 + R_2 + ... + R_n. \tag{4.31}$$

Where, R_1, R_2, … ,R_n are the sub-bitrate contributed from encoding of each *I* frame in a GOP. The bit rate of the first GOP in equation (4.31) is the actual resulting rate after encoding a GOP. The target bitrate for a GOP which has *n* number of *I* frames is the combination of sub-target bitrate contributed from each *I* frame and can be expressed as follows:

$$R_T = R_{T1} + R_{T2} + ... + R_{Tn} \tag{4.32}$$

where, R_{T1}, R_{T2}, …. R_{Tn} are the required sub-target bitrate for an *I* frame in order to meet the user defined target bitrate over a GOP. It is important to note that the algorithm is regulating the bitrate over a GOP but not on each and every frame. So, the resulting bitrate after encoding or calculated target bitrate for a GOP is the total bitrate contributed from each frame in this GOP as shown in equations (4.31) and (4.32). This statement is also true for inter frame coding. Being one pass only algorithm, the rate control algorithm in *I* frame-only coding mode is always lagging one frame behind. It is because the value of *QF* which is optimum for the n^{th} frame in order to meet the target bitrate is used to encode the $(n+1)^{th}$ frame. But it is expected to achieve an acceptable level of accuracy because the complexities of the adjacent frames are approximately the same as long as there is no abrupt scene change in the encoding video sequence.

4.5.3.2 Inter Frame Coding ($IL_2L_2L_1$)

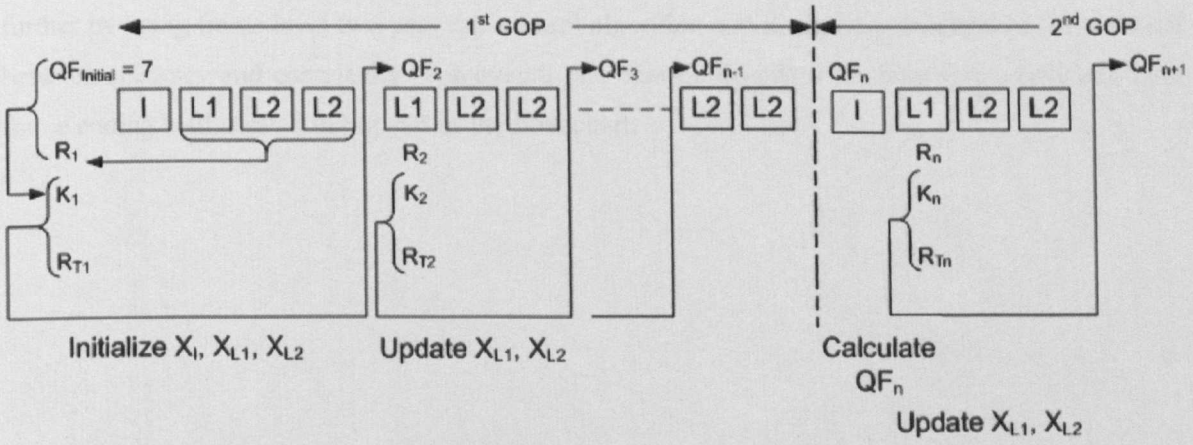


Fig. 4-4 Rate Control Procedure for Inter Frame Coding

In Fig. 4-4, the first *I* frame and the first sub-GOP which consist of L_1, L_2, L_2 frames are encoded by using the initial *QF* which is set to 7 (medium quality). R_1 is calculated by using the number of bits required to encode the L_1 and two L_2 frames without including *I*, frame rate and GOP length as shown in intra frame-only coding section.

After getting R_1, K_1 can be calculated by substituting $QF_{Initial}$ and R_1 in equation (4.27). The corresponding complexities of the frames, X_I, X_{L1} and X_{L2} are initialized with the actual number of bits required to encode these frames but the complexity of L_2 frame is the average value since there are

two L_2 frames. After that, the bit allocation process generates the optimum number of bits required to encode first sub-GOP frames (without including *I*) to achieve the target bitrates by using equation (4.29) and (4.30), and calculates R_{T1}. R_{T1} can be considered as the required sub-target bitrate contributed from the first sub-GOP in order to meet the user defined target bitrate which is calculated over a GOP. It is important to note that the bit allocation of TM5 [37] is used here in order to calculate the sub-target bit rate contributed from a sub-GOP (e.g. R_{T1}) which consists of different frame types. It is not required in intra frame-only coding since the ratio of the number of bits generated from the frames in a GOP are the same because of having the same frame type (*I* frame only).

From K_1 and R_{T1}, the optimum QF for the first sub-GOP, which will be used to encode the next successive sub-GOP (i.e. L_1, L_2, L_2) as QF_2 can be generated by using equation (4.27). The same procedure continues until the end of first GOP. The complexities of the L_1 and L_2 frames, X_{L1} and X_{L2} are updated following the encoding of each sub-GOP.

QF of next *I* frame which belongs to the 2nd GOP and is denoted as QF_n in the Fig. 4-4, is the average of the QF of *I* frame in the previous GOP and the QF of previous sub-GOP, which is QF_{n-1}. As discussed in *I* frame-only coding section, the rate control algorithm in inter frame coding mode is always lagging one sub-GOP (which is L_1, L_2, L_2) behind. It is because the value of QF which is optimum for the n^{th} sub-GOP in order to meet the target bitrate is used to encode the $(n+1)^{th}$ sub-GOP. But, the algorithm can still regulate the optimum birate over a GOP as long as there is no abrupt scene change in the encoding video sequence. However, the accuracy of the algorithm can be increased further by using frame level two pass rate control algorithm and it can be considered as the trade-off between accuracy and complexity. The overall rate control algorithm for Intra frame-only and inter frame coding is illustrated in Fig. 4-5 as the flow chart.

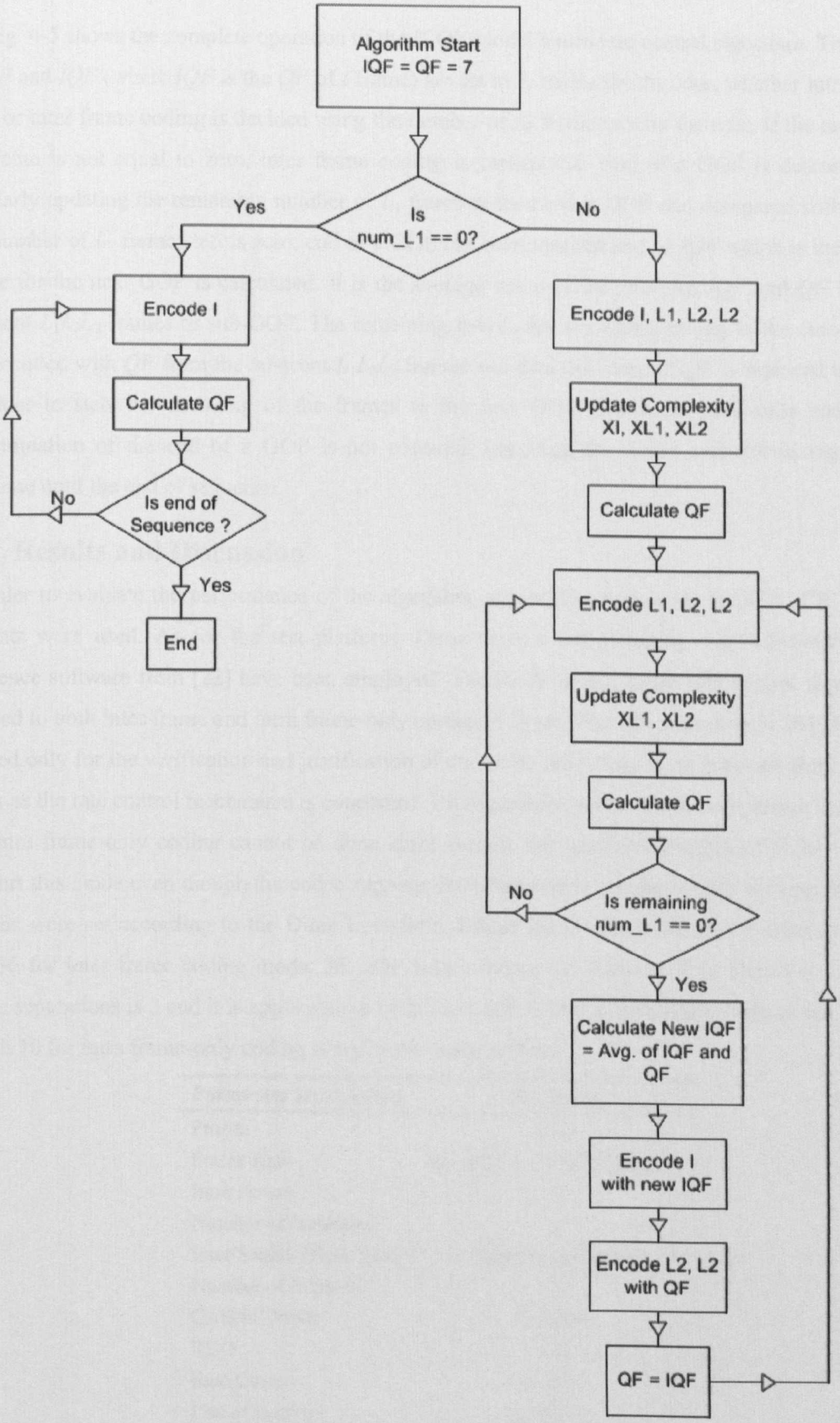


Fig. 4-5 Complete Operation of the R-QF Model based Rate Control Algorithm

Fig. 4-5 shows the complete operation of the R-QF Model based rate control algorithm. The values of *QF* and *IQF* (where *IQF* is the *QF* of *I* frame) are set to 7. Inside the encoder, whether intra frame-only or inter frame coding is decided using the number of L_1 frames set by the user. If the number of L_1 frame is not equal to zero, inter frame coding is carried out. End of a GOP is determined by regularly updating the remaining number of L_1 frame in the current GOP and compared with zero. If the number of L_1 frames left is zero, end of a GOP has been reached and so *IQF* which is the *QF* of *I* frame for the next GOP is calculated. It is the average value of the previous *IQF* and *QF* from the adjacent $L_1L_2L_2$ frames or sub-GOP. The remaining two L_2 frames which belong to the current GOP are encoded with *QF* from the adjacent $L_1L_2L_2$ frames and then the value of *QF* is replaced with *IQF* in order to start the encoding of the frames in the new GOP. For intra frame-only coding, the determination of the end of a GOP is not required. Encoding the frame and calculating the *QF* continue until the end of sequence.

4.6 Results and Discussion

In order to evaluate the performance of the algorithm, several test sequences in QCIF, CIF and HD formats were used. As for the test platform, Dirac version 0.6 from [3] and H.264/AVC JM11 reference software from [22] have been employed. The R-QF Model based rate control algorithm is applied to both inter frame and intra frame-only coding in Dirac. The rate control in H.264 JM11 [41] is used only for the verification and justification of our work since there is no previous work in Dirac as far as the rate control mechanism is concerned. Unfortunately, performance comparison with H.264 for intra frame-only coding cannot be done since current rate control algorithm of H.264 does not support this mode even though the codec supports intra frame-only coding in their high profile. GOP lengths were set according to the Dirac Encoder's default values which are 10 for intra frame-only and 36 for inter frame coding mode. 36 GOP length means the number of L_1 frames is 11 and L_1 frame separations is 3 and it is applicable to both Dirac and H.264 in inter frame coding. But the GOP length 10 for intra frame-only coding is applicable only to Dirac.

Parameter Description	Set Value
Profile	Main
Frame Rate	10 (QCIF), 15 (CIF), 24 (HD)
Intra Period	12
Number of Reference	2
Inter Search Block Sizes	16×16, 8×8, 4×4
Number of B frames	2
CABAC Mode	Disabled
RDO	On
Rate Control	Enabled
Use of FastME	3 (EPZS)

Table 4-1 H.264 Configuration File Parameters

The parameters in configuration file of H.264 were carefully chosen in order to have fair comparison with Dirac for inter frame coding. Table 4-1 shows the list of configuration parameter used in H.264 encoding.

The rate control results are presented in three separate sections: PSNR performance, deviation error from the target bitrate and the subjective quality. In PSNR performance, PSNR-Y performance of the algorithm's results using R-QF Model is compared with H.264 for different target bitrates and different video formats. Deviation error from the target bitrate is shown in percentage, where the two parameters, the maximum deviation error and the average deviation error are used to compare between the rate control algorithms of two codecs. Maximum deviation error is the maximum value of deviation errors which is occurred over each and every GOPs and the average deviation error is the error over the bitrate which is averaged over the whole sequence. In subjective quality assessment, some frames are extracted from the video sequence and their reconstruction qualities are discussed.

4.6.1 PSNR Performance

4.6.1.1 Intra Frame-Only Coding

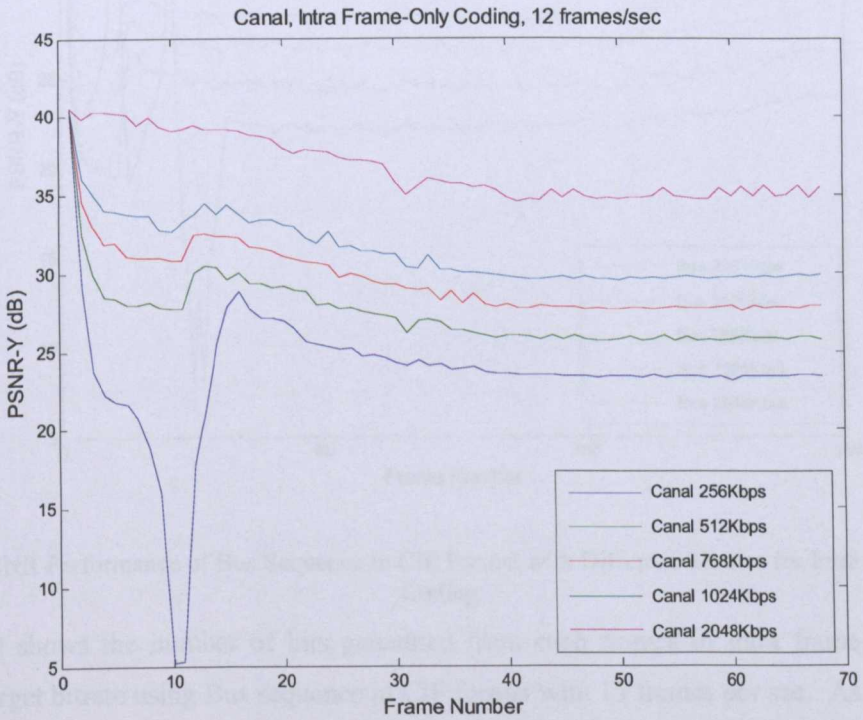


Fig. 4-6 PSNR Performance of Canal Sequence in CIF Format with Different bitrates for Intra Frame-Only Coding

Fig. 4-6 and Fig. 4-7 show the PSNR performance of Canal and Bus sequence in CIF format with Intra frame-only coding for the different bitrates. The algorithm tries to adjust its parameters in order to get the optimum *QF* for a given target bit rate and becomes stable after encoding about 30 frames or 3 GOP durations, which can be seen clearly in the figures. The PSNR deep fading in Fig. 4-6 and

fluctuation in Fig. 4-7 especially in the 256 kbps target bit rate coding, is the result of the initial *QF* setting which is too high for that particular target bitrate. Encoding the first frame gives the PSNR quality which is more than 35 dB because of the initial value of *QF*, which in turn causing the algorithm to lower down the quality of next successive frame in order to meet the target bit rate. The same cycle which causes PSNR over shooting and under shooting continues until the algorithm finds the optimum *QF* for the set target bit rate after encoding to a certain number of frames. The problem could be solved either by inserting a certain number of frames (about 30 to 40 frames) at the beginning as a training sequence or setting the proper initial *QF* approximated from the coding mode whether using Intra frame-only or inter frame coding, target bitrate and frame rate, instead of setting constant initial value which is set to 7 currently.

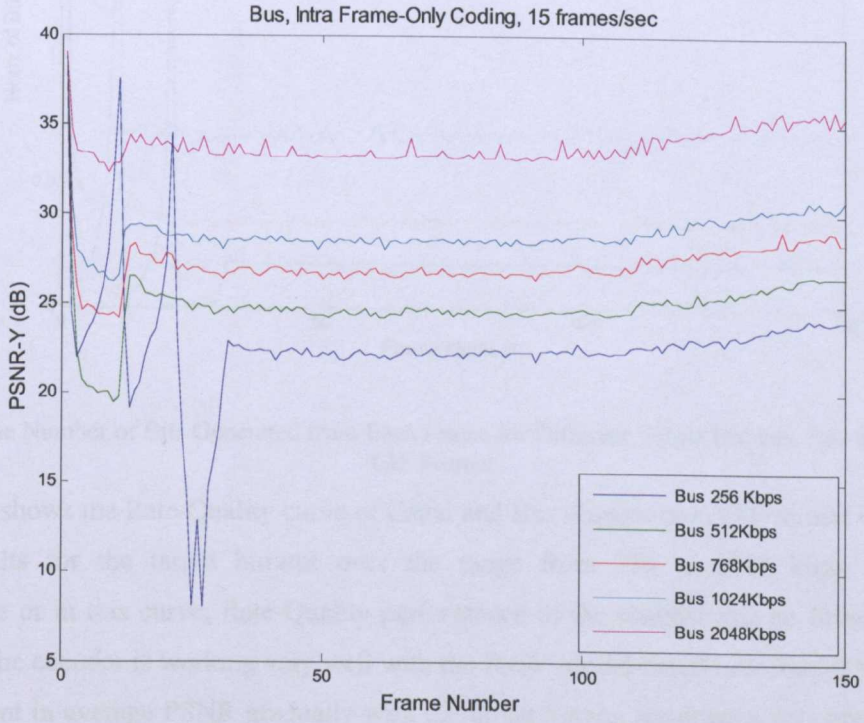


Fig. 4-7 PSNR Performance of Bus Sequence in CIF Format with Different Bitrates for Intra Frame-Only Coding

Fig. 4-8 shows the number of bits generated from each frames in intra frame-only coding for different target bitrate using Bus sequence in CIF format with 15 frames per sec. As expected, PSNR fluctuation in Fig. 4-7 for 256 Kbps target bitrate encoding is reflected in this figure as well in the form of bitrate overshooting and undershooting from the optimum value. The optimum number of bit required for each frame (bits/frame) in order to meet the target bitrate can be easily calculated by dividing the target bitrate (bits/sec) with frame rate (frame/sec) in intra frame-only coding. In Fig. 4-8, the algorithm becomes stable and starts generating the optimum number of bits for each frame (e.g. for 256 Kbps, $256/15 = 17.07\times10^3$ bits) after encoding approximately three GOPs. The curves in Fig. 4-8 are the perfect example in showing the accuracy of the algorithm where all the curves for different

target bitrate are in straight line after achieving the stability. This shows that the algorithm is generating approximately constant bitrate which is very close to target bitrate with the corresponding PSNR curve in slightly increasing trend as shown in Fig. 4-7.

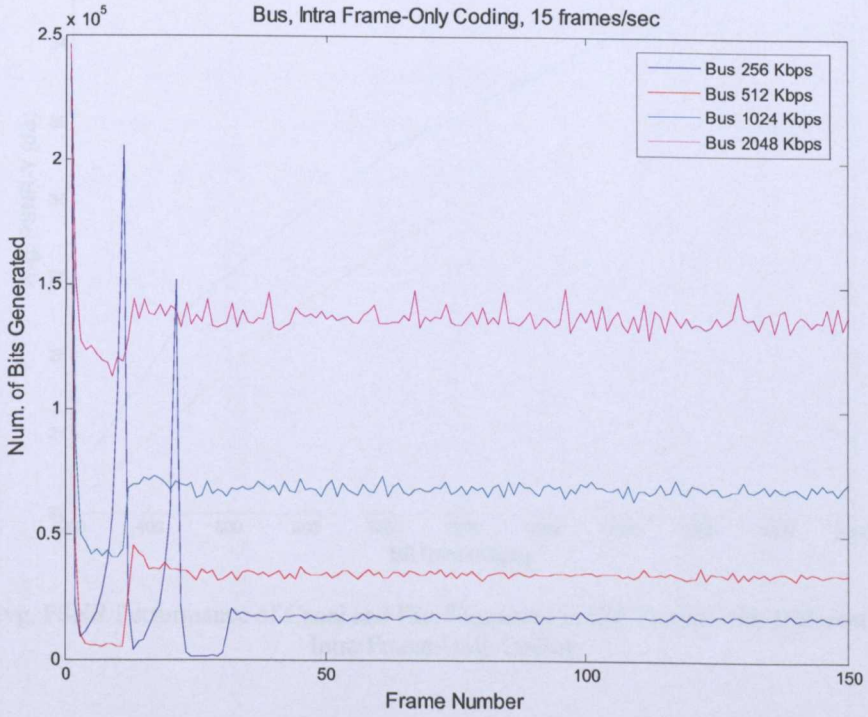


Fig. 4-8 The Number of Bits Generated from Each Frame for Different Target Bitrates, Bus Sequence in CIF Format

Fig. 4-9 shows the Rate-Quality curve of Canal and Bus sequences in CIF format with the average PSNR results for the target bitrates over the range from 256 to 2048 kbps. Rate-Distortion performance or in this curve, Rate-Quality performance of the encoder can be found clearly in this figure and the encoder is working very well with the R-QF Model based rate control algorithm giving the increment in average PSNR gradually with the target bitrate, resulting a very smooth curve. The maximum value corresponds to 2048 kbps is 36.73 and 33.84 dB for Canal and Bus sequences, respectively. In both curves in Fig. 4-9, average PSNR tends to increase very rapidly in lower bitrate coding (i.e. from 256 to 1024 Kbps) and then the increment becomes linear in higher bitrate coding (i.e. from 1024 to 2048 Kbps).

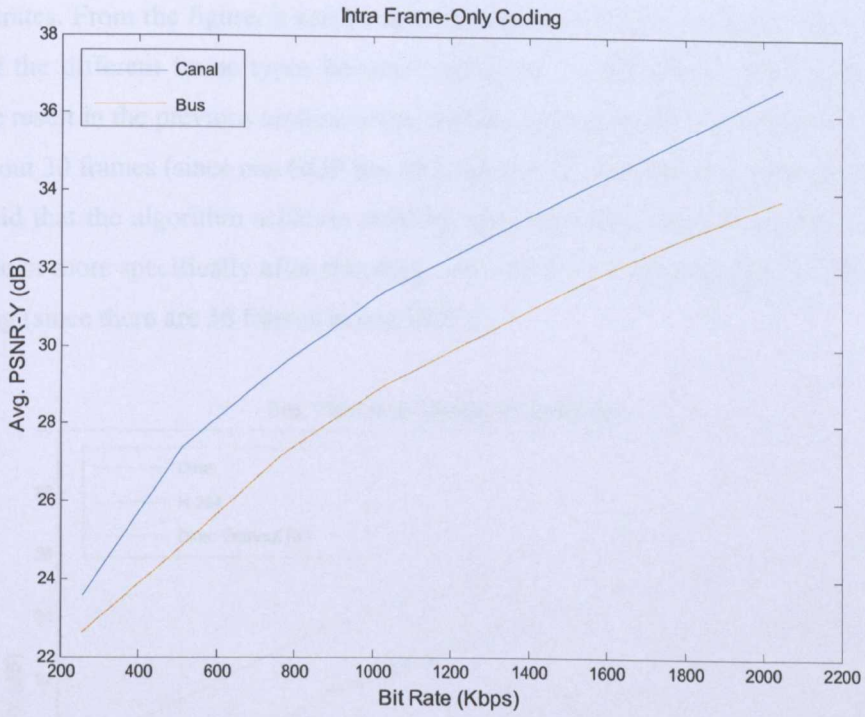


Fig. 4-9 Avg. PSNR Performance of Canal and Bus Sequences in CIF Format with Different Bitrates for Intra Frame-Only Coding

4.6.1.2 Inter Frame Coding

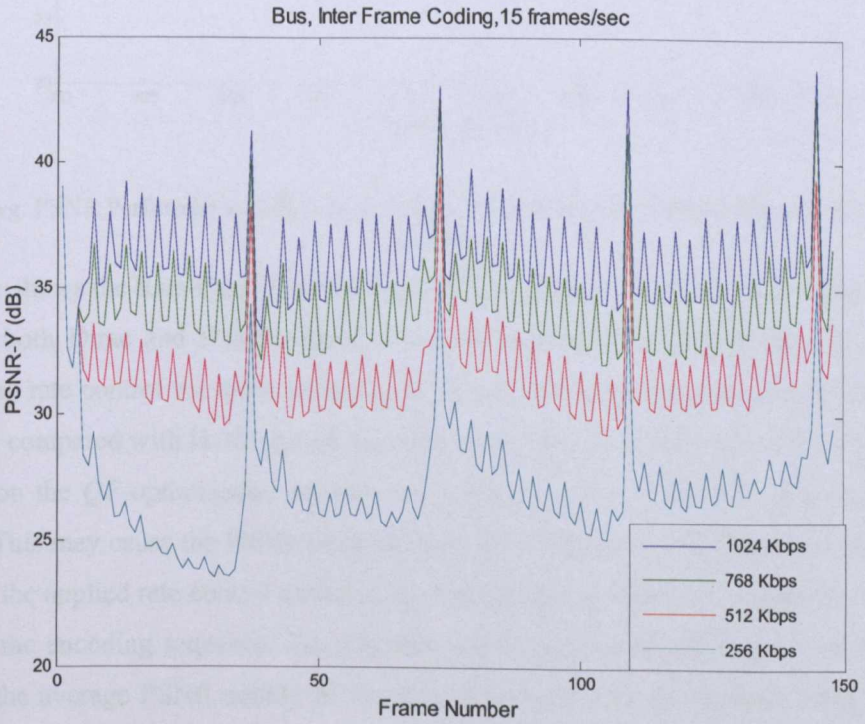


Fig. 4-10 PSNR Performance Comparison of Bus Sequence in CIF Format with different Bitrates for Inter Frame Coding

Fig. 4-10 shows the PSNR performance of Bus sequence in CIF format with inter frame coding for different bitrates. From the figure, it can be seen clearly that after encoding the first GOP, the PSNR variation of the different frame types become regular in the following GOPs. This statement also supports the result in the previous section where stability of the algorithm is achieved after encoding 3 GOPs or about 30 frames (since one GOP has 10 frames in intra frame-only coding). So, as an overall, it can be said that the algorithm achieves stability after encoding about 30 frames for both types of coding mode or more specifically after encoding first 3 GOPs in intra frame-only and 1st GOP in inter frame coding (since there are 36 frames in one GOP).

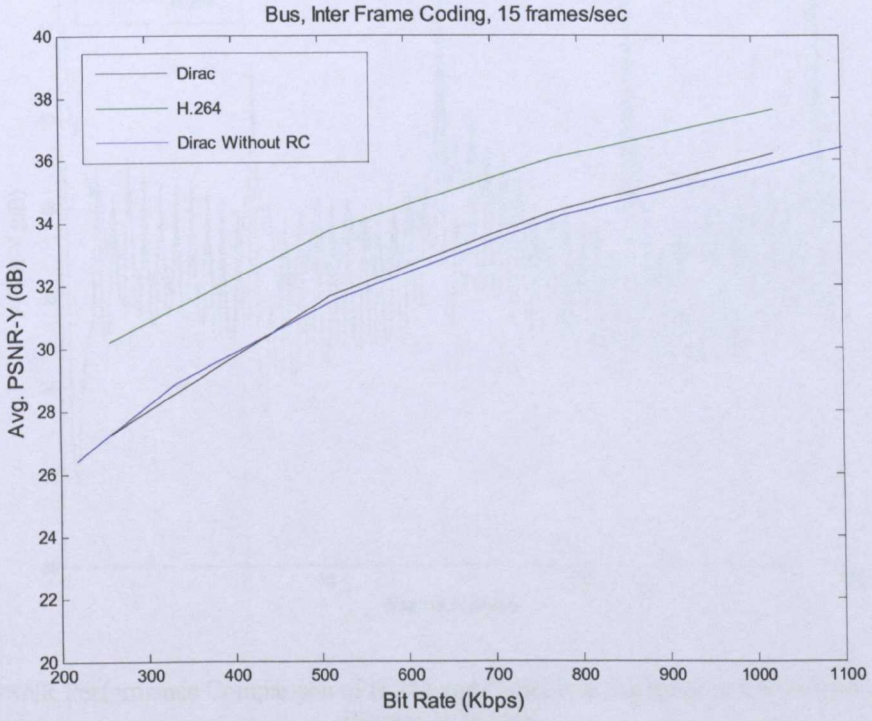


Fig. 4-11 Avg. PSNR Performance of Bus Sequence in CIF Format with different Bitrates for Inter Frame Coding

Fig. 4-11 shows the Rate-Quality performance results of bus sequence in CIF format with different bitrates for both Dirac and H.264 codecs. It can be seen that the performance of Dirac with and without using rate control are quite similar even though Dirac has lower average PSNR performance (about 4dB) compared with H.264 for all target bit rates. Since the principle of the algorithm in Dirac is based upon the *QF* optimization method, the value of *QF* in each frame is no longer a constant parameter. This may cause the PSNR fluctuation or noticeable quality difference to the viewers if the response of the applied rate control model is not fast enough to adapt the complexity level or types of contents of the encoding sequence. So, it is required to justify the performance of the algorithm by comparing the average PSNR quality of encoding sequence with and without using rate control for different target bitrates. According to the results from Fig. 4-11, there is no loss in terms of PSNR performance upon employing the rate control algorithm using R-QF Model. As shown in Fig. 4-11,

the two curves of Dirac with and without using rate control give the PSNR performances which are almost the same. The average PSNR curve without rate control is generated by encoding with the constant *QF* for the whole sequence. The value of *QF* used here are 5, 6, 7, 8 and 9 which correspond to the rate 218.95, 333.32, 516.23, 779.88 and 1173.03 Kbps, respectively. For constant *QF* coding, calculated bitrates are based upon the whole sequence and the bitrate averaged over each GOP can be arbitrary in this case.

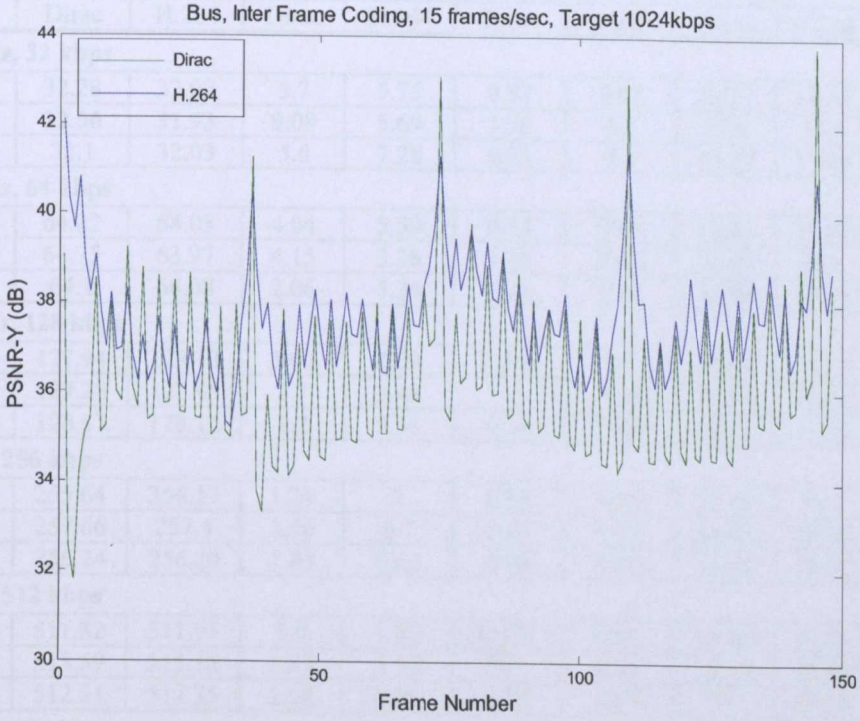


Fig. 4-12 PSNR Performance Comparison of H.264 and Dirac, Bus Sequence in CIF Format with Target Bitrates 1024 kbps

Fig. 4-12 shows the PSNR performance comparison of two codecs with Bus sequence in CIF format for the target bitrate 1024 kbps. Average PSNR-Y of the Dirac is 36.19 dB which is 1.47 dB lower than that of H.264. Even though Dirac suffers 1.47 dB loss in average, the PSNR value of *I* frame in Dirac is even higher than that of H.264 in Fig. 4-12. However, the PSNR difference between *I* frames and L_1 frames is much larger than that of H.264 and the same problem applies to the frames between L_1 and L_2, which gives the lower average PSNR value in Dirac.

From Table 4-2, it can be seen clearly that the maximum deviation error of the rate control technique [41] in JM11 reference software of H.264 is higher than that of the R-QF Model based technique with Dirac in most of the time especially in the larger frame size (e.g. CIF and HD) and smaller frame size with higher target bit rate (e.g. QCIF with 128 Kbps). Having better bitrate regulation is one of the important factors in real-time transmission since it can help to prevent buffer overflow and underflow. Even though average deviation error from the target bitrate of the presented algorithm with Dirac is higher than H.264 in most of the cases, especially for lower target bitrates, the

percentage of the average deviation error is always within 1% except Foreman sequence in QCIF encoding. But it is important to note that the R-QF Model based algorithm achieves stability only after encoding about 30 frames or 1 GOP in inter frame coding. Clearly, higher maximum and average deviation error of Foremen sequence in QCIF format with target bitrate 32 Kbps (i.e. 9.08 and 1.76 %) are the result of higher error in first GOP encoding.

Sequence	Avg. Rate (kbps)		Max. Dev. Error (%)		Avg. Dev. Error (%)		PSNR-Y (dB)		SSIM-Y	
	Dirac	H.264	Dirac	H.264	Dirac	H.264	Dirac	H.264	Dirac	H.264
QCIF, 10 Hz, 32 kbps										
Carphone	32.28	32.02	5.7	5.75	0.87	0.07	29.27	34.54	0.87	0.94
Foreman	32.56	31.93	9.08	5.69	1.76	0.2	27.6	33.6	0.81	0.93
Highway	32.1	32.03	5.6	7.28	0.31	0.1	33.37	37.89	0.91	0.95
QCIF, 10 Hz, 64 kbps										
Carphone	64.22	64.03	4.04	5.59	0.34	0.04	34.46	37.97	0.95	0.97
Foreman	64.75	63.97	4.15	3.28	1.18	0.04	33.97	36.8	0.94	0.96
Highway	64.1	64.08	2.06	5.34	0.16	0.13	36.98	39.95	0.95	0.97
QCIF, 10 Hz, 128 kbps										
Carphone	127.94	128.13	3.00	4.09	0.044	0.1	38.56	41.57	0.98	0.98
Foreman	129.28	128.45	2.7	3.49	1	0.36	38.16	40.08	0.97	0.98
Highway	128.17	128.12	1.2	5.38	0.13	0.09	39.42	41.75	0.97	0.97
CIF, 15 Hz, 256 kbps										
Bus	255.64	256.17	1.24	3	0.14	0.06	27.19	30.2	0.84	0.88
Foreman	257.66	257.4	2.56	6.7	0.65	0.55	34.83	36.89	0.93	0.95
Highway	256.24	256.20	1.63	8.62	0.09	0.07	37.62	39.07	0.94	0.94
CIF, 15 Hz, 512 kbps										
Bus	511.32	511.95	1.0	3.32	0.133	0.01	31.64	33.73	0.93	0.94
Foreman	515.59	515.18	1.87	5.18	0.7	0.62	38.23	39.89	0.96	0.97
Highway	512.21	512.35	2.02	4.04	0.04	0.07	39.1	40.28	0.95	0.95
CIF, 15 Hz, 1024 kbps										
Bus	1023.14	1023.9	1.24	3.76	0.084	0.01	36.19	37.66	0.97	0.97
Foreman	1029.38	1031.5	2.3	5.06	0.525	0.73	41.21	42.9	0.98	0.98
Highway	1023.67	1024.37	1.29	2.37	0.03	0.04	40.54	41.6	0.96	0.96
HD720P, 24 Hz, 2Mbps										
Knight Shield	1999.87	2002.17	0.79	4.3	0.0065	0.1086	35.58	36.25	0.92	0.93
HD1080P, 24Hz, 2Mbps										
Pedestrian area	2001.55	2002.2	1.44	3.62	0.077	0.11	35.998	36.5	0.91	0.93

Table 4-2 Comparison of Rate Control Results for Dirac and H.264

However, the average deviation error becomes comparable or even lower than that of H.264 when the target bitrate is higher in larger frame encoding especially in 1024 kbps for CIF and HD format encoding. In terms of PSNR, Dirac suffers lower PSNR performance compared with H.264 because of the higher PSNR difference between I, L_1 and L_2 frames.

Now a day, peak signal-to-noise ratio (PSNR) which is the most commonly used objective video distortion/quality metrics has been widely criticized for not correlating well with perceived quality measurement. In the last three decades, a great deal of effort has gone into the development of quality

assessment methods that take advantage of known characteristics of the Human Visual System (HVS). The majority of the proposed perceptual quality assessment models have followed a strategy of modifying the Mean Square Error (MSE) measure so that errors are penalized in accordance with their visibility. The Video Quality Experts Group (VQEG) was formed to develop, validate and standardize new objective measurement methods for video quality. Some of the developed models are commercially available. Among them, a measure of Structural SIMilarity (SSIM) index [58] is one of the popular metrics that compares local patterns of pixel intensities that have been normalized for luminance and contrast. In Table 4-2, SSIM index for all the encoded test sequences have been measured and presented in the last column. The index ranges from 0 to 1 from non-similarity to perfect similarity. According to the results from Table 4-2, SSIM index between the two codecs becomes comparable when the frame size increases (CIF and HD) even though encoded video sequence with Dirac shows lower index for smaller frame size (QCIF) especially with lower target bit rate.

In general, the presented rate control algorithm in Dirac shows lower percentage of maximum deviation error compared with H.264 in most of the cases. But in the measure of average deviation error, PSNR-Y and SSIM-Y, encoded video sequences with Dirac show lower performance especially with the lower target bitrate (32, 64 Kbps) in QCIF frame coding. But the average deviation error and SSIM performance becomes comparable with that of H.264 as the target bitrate increases (512, 1024 Kbps) in CIF and HD format video sequences while PSNR performance of Dirac is lower in all cases.

The encoder still needs to be developed further in order to achieve better or at least comparable PSNR performance with H.264. However, more importantly, Dirac was designed to optimize the subjective performance, if necessary at the expense of objective performance [59] and so it is less likely to obtain better PSNR than H.264 in future versions of the Dirac.

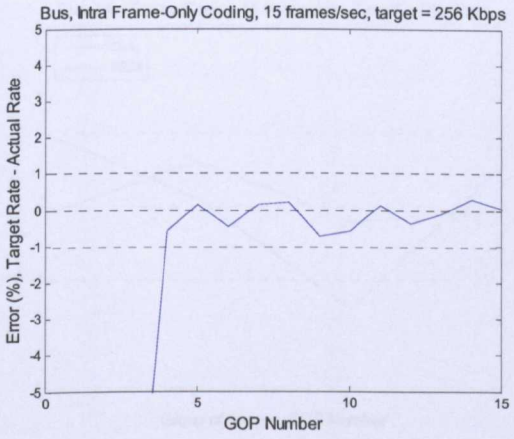
4.6.2 Deviation Error from the Target Bit Rate

Deviation error from the target bitrate is the percentage of bitrate error, which is the difference between target bitrate and resulting coding bitrate averaged over a GOP and can be calculated as follow.

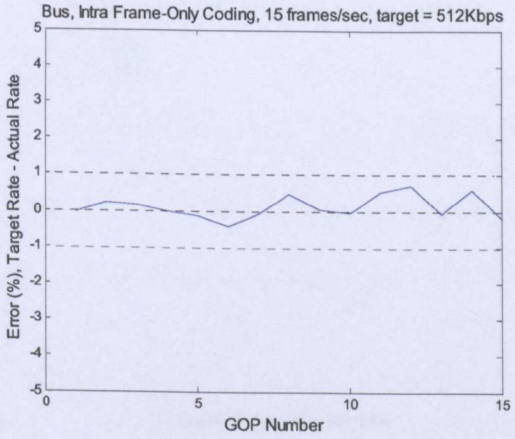
$$\text{Deviation Error (\%)} = \frac{\text{Target Rate - Actual Rate}}{\text{Target Rate}} \times 100 \tag{4.33}$$

All the curves in this section are drawn for the error percentage of resulting bitrate average over a GOP against their corresponding GOP number.

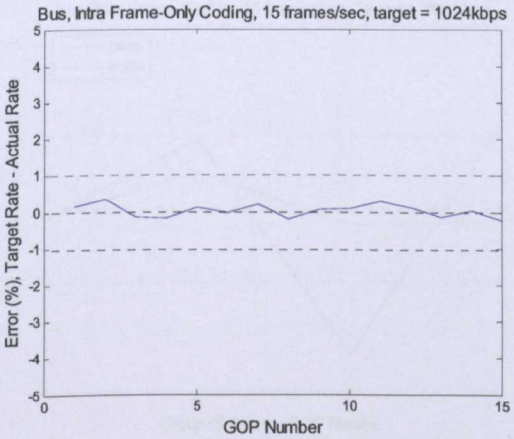
4.6.2.1 Intra Frame-Only Coding



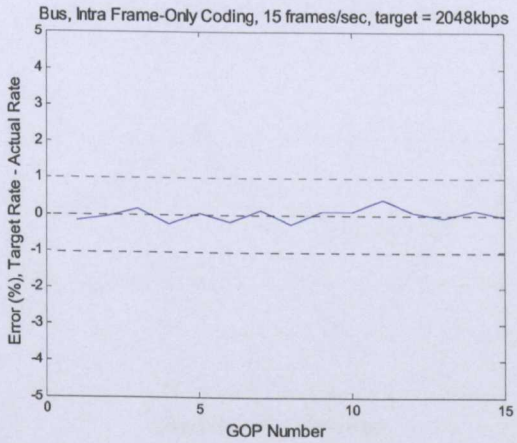
(a) 256 Kbps



(b) 512 Kbps



(c) 1024 Kbps

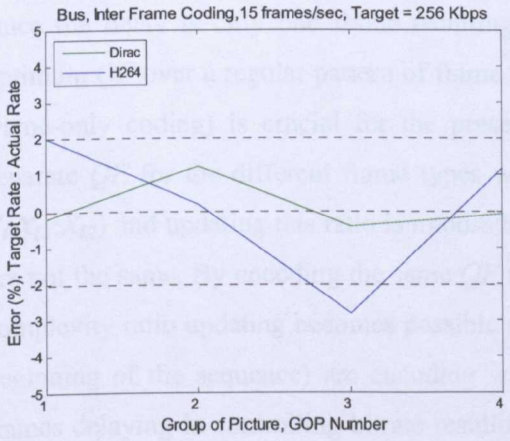


(d) 2048 Kbps

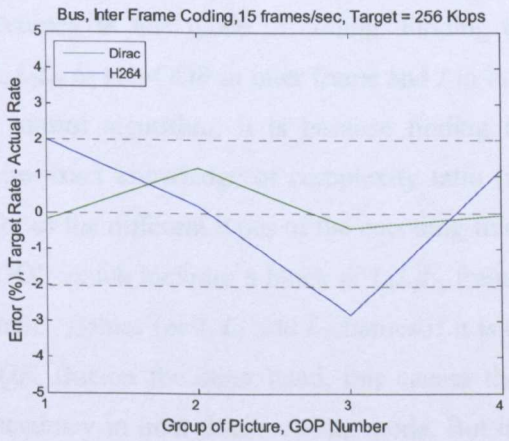
Fig. 4-13 Percentage of Resulting Bitrate's Error from the Target Bitrate, Bus Sequence in CIF Format

Fig. 4-13 (a) to (d) show the error percentage of resulting bitrates averaged over each GOP for Bus sequence in CIF format for the target bitrates 256, 512, 1024 and 2048 Kbps, respectively. From the figures, it can be seen clearly that the algorithm for Intra frame-only coding performs very well and the precision is within 1% of the all target bitrates except 256Kbps. In 256 Kbps target bit rate encoding, the bitrate averaged over a GOP overshoot the target bitrate resulting higher percentage of deviation error for the first few GOPs encoding. It is interesting to see that the percentage of deviation error becomes less and less as the target bitrate increases and the curve in Fig. 4-13 (d) where the target bitrate is 2048 Kbps is the smoothest one among all others curves shown in Fig. 4-13. It clearly shows that the performance of the presented algorithm increases with the target bitrate.

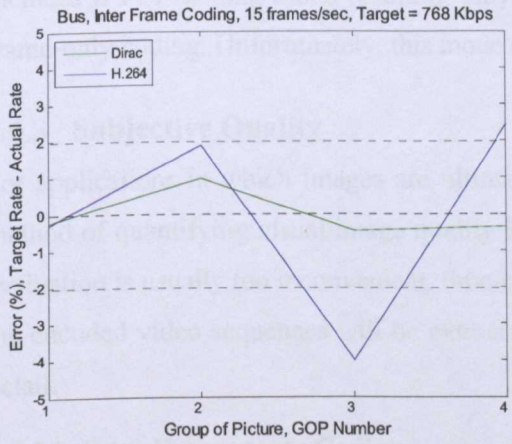
4.6.2.2 Inter Frame Coding



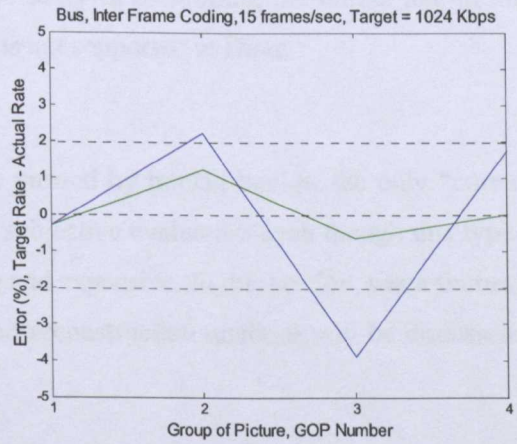
(a) 256 Kbps



(b) 512 Kbps



(c) 768 Kbps



(d) 1024 Kbps

Fig. 4-14 Percentage of Resulting Bitrate's Error from the Target Bitrate, Bus Sequence in CIF Format

Fig. 4-14 (a) to (d) show the error percentage comparison of resulting bitrates averaged over each GOP for Bus sequence in CIF format for the target bitrates 256, 512, 768 and 1024 Kbps, respectively for both codecs. The presented algorithm in Dirac performs very well also in inter frame coding and the precision is around 1% of target bitrates. From the figures, it can be seen clearly that Dirac's rate control algorithm offers better bitrate regulation over each GOP than H.264, which is really important in real-time transmission to prevent buffer overflow and underflow. For example, the maximum deviation error from the target bitrate of Dirac is only 1.24% for 1024 Kbps target bitrate, which is much smaller compared with 3.76% for H264 as shown in Table 4-2.

From Fig. 4-13 and Fig. 4-14, it can be seen that the presented rate control algorithm works more accurately in intra frame-only coding giving the deviation error which is much lower than inter frame coding mode. It is because in inter frame coding, as discussed in section 4.5.3, the algorithm have to calculate the optimum *QF* for a group of frames called sub-GOP and applied this optimum *QF* to the next adjacent sub-GOP resulting one sub-GOP or 3 frames delay in controlling the bitrate. But in intra

frame-only coding, since all the frames have the same type (*I* frame), finding the optimum *QF* for one frame and applying it to the next adjacent frame proved to be much faster in controlling the bitrate since the delay is only one frame resulting better accuracy in this mode of coding. Finding the optimum *QF* over a regular pattern of frame type (i.e $L_1L_2L_2$ or sub-GOP in inter frame and *I* in intra frame-only coding) is crucial for the presented rate control algorithm. It is because finding the separate *QF* for the different frame types would require exact knowledge of complexity ratio (i.e. X_I:X_{L1}:X_{L2}) and updating this ratio is impossible if the *QF* of the different types of the encoding frame are not the same. By encoding the same *QF* for a sub-GOP which includes a block of $L_1L_2L_2$ frames, complexity ratio updating becomes possible since L_1 and L_2 frames (or *I*, L_1 and L_2 frames if it is the beginning of the sequence) are encoding with same *QF*. But on the other hand, this causes three frames delaying in controlling bitrate resulting lower accuracy in inter frame coding mode. But it is expected to achieve higher accuracy if this algorithm is applied to basic profile of H.264 which includes *IPPPP* coding mode resulting only one frame delay in controlling the bitrate like in intra frame-only coding. Unfortunately, this mode of coding is not supported in Dirac.

4.6.3 Subjective Quality

For applications in which images are ultimately to be viewed by human beings, the only "correct" method of quantifying visual image quality is through subjective evaluation even though this type of evaluation is usually too inconvenient, time-consuming and expensive. In this section, some frames of the encoded video sequences will be extracted and their reconstruction qualities will be discussed in detail.

4.6.3.1 Intra Frame-Only Coding

Fig. 4-15 shows the subjective quality of the frame number 139 from Bus sequence with different target bitrates for Intra frame-only coding. The reconstruction quality gradually improves as the target bitrate increases and their corresponding PSNR values for the Fig. 4-15 (a) to (d) are 23.85, 26.14, 30.26 and 34.91 dB, respectively. There is noticeable edge blur and ringing artifacts at the area around discontinuities which is shown with red circle in Fig. 4-15 (a) and (b) because of the loss of high frequency components. It happens when there is high compression ratio or lower target bitrate. It is because the algorithm generates smaller *QF* (larger λ) for lower target bitrate encoding resulting larger *QP* because of RDO quantization. Quantization with larger *QP* causes most of the high frequency wavelet coefficients to be suppressed to zero resulting ringing artifacts at the sharp edges. The condition gradually improves as the bit rate increase (lower compression ratio) in Fig. 4-15 (c) and (d).

(a) Bus Sequence, 256 Kbps

(b) Bus Sequence, 512 Kbps

(c) Bus Sequence, 1024 Kbps

(d) Bus Sequence, 2048 Kbps

Fig. 4-15 Subjective Quality of Bus Sequence in CIF Format (Frame Num. 139) with Intra Frame-Only Coding using Dirac

4.6.3.2 Inter Frame Coding

Fig. 4-16 and Fig. 4-17 show the subjective quality of frame number 139 of Bus and frame number 83 of Foreman sequences, respectively with target bitrates 256 kbps for inter frame coding. Their corresponding PSNR values are 28.3, 32.21, 34.22 and 38.65 dB for Bus sequence in Fig. 4-16 (a)-(b) and Foreman sequence in Fig. 4-17 (a)-(b), respectively. From the figures, it can be seen clearly that the overall subjective quality of the H.264 is better than Dirac. Dirac suffers some ringing artifacts especially at the sharp edges of the solid white line in Bus sequence shown with red circle in Fig. 4-16 (a) and fast moving foreground object of the Foreman sequence shown with red circle in Fig. 4-17 (a).

(a) Dirac (b) H.264

Fig. 4-16 Subjective Quality of Bus Sequence in CIF format (Frame Num. 139) with Inter Frame Coding, Target Rate 256 kbps

(a) Dirac (b) H.264

Fig. 4-17 Subjective Quality of Foreman Sequence in CIF Format (Frame Num. 83) with Inter Frame Coding, Target Rate 256 kbps

It happens only for the fast motion with low bit rate coding because the algorithm generates the smaller QF (higher λ) for the lower target bit rate, which results higher λ_{ME} and λ for the RDO of motion estimation and quantization stages as shown in Fig. 4-1. According to equation (2.2), higher λ_{ME} causes the MVs over smoothing in the RDO process of the motion estimation yielding the less accurate MVs in the motion estimation stage. Similarly, higher λ causes the RDO process to choose higher quantization step size in the RDO quantization stage. The Dirac encoder still needs to be optimized in the procedure of motion estimation and quantization in order to give the optimum results even with the lower QF encoding especially for the fast motion sequences.

(a) the Whole Frame, Uncoded Original

(b) Cropped Uncoded

(c) Cropped, Dirac

(d) Cropped, H.264

Fig. 4-18 Subjective Quality of Knight Shield Sequence, HD720P (Frame Num. 145, I frame) with Inter Frame Coding

Fig. 4-18 (a) to (d) show the subjective quality of frame number 145 of Knight Shield sequences, where Fig. 4-18 (a) shows the entire frame of uncoded original and Fig. 4-18 (b)-(d) are the cropped version of uncoded original, coded using Dirac and H.264, respectively. The coded frames which are *I* frames in Fig. 4-18 (c) and (d) are encoded with the target bitrates 2 Mbps in inter frame coding mode and their PSNR are 38.3572 dB and 36.9255 dB, respectively. The cropped area is shown with red rectangular shape in the whole frame in Fig. 4-18 (a). Moreover, the cropped frames in Fig. 4-18 (b)-(d) are magnified 300% so that their detail texture can be seen easily.

The uncoded frame is shown here in order to compare the subjective quality of the coded frames with the original one. It is interesting to see that most of the textures were lost in the H.264 coded frame especially in the background of the knight shields while the frame coded with Dirac can still maintain the detail texture, which is really important in HD video encoding.

4.7 Chapter Summary

This chapter presented the rate control algorithm which is efficient, simple and easy to integrate to the Dirac encoder. The algorithm is based upon the *QF* optimization method where *QF* of the encoded video is adaptively changing in order to achieve average bitrate which is constant over each GOP. A relation between the bitrate, *R* and the *QF*, which is called R-QF model is derived in order to estimate the optimum *QF* of the current encoding frame for a given target bitrate, *R*.

First of all, PSNR performance of the intra frame-only and inter frame coding mode of the presented algorithm with different target bitrate are evaluated. It was found that the algorithm becomes stable after encoding about 30 frames or 3 GOPs in intra frame-only and around 30 frames or 1 GOP duration in inter frame coding mode. In intra frame-only coding mode, the PSNR deep fading and fluctuation at the beginning of the sequence especially in the 256 kbps target bit rate coding was found and it is mainly due to the result of the initial *QF* setting which is too high for that particular target bitrate. It was suggested that this kind of problem can be solved either by inserting a certain number of frames (about 30 to 40 frames) at the beginning as a training sequence or setting the proper initial *QF* approximated from the coding mode whether using Intra frame-only or inter frame coding, target bitrate and frame rate, instead of setting constant initial value which is set to 7 currently.

More importantly, it was also found that there is no loss in terms of PSNR performance upon employing the presented rate control algorithm. From the experimental results with different test sequences and different video formats, it was found that the maximum deviation error of the rate control technique [41] in JM11 reference software of H.264 is higher than that of the presented technique with Dirac in most of the time especially in the larger frame size (e.g. CIF and HD) and smaller frame size with higher target bit rate (e.g. QCIF with 128 Kbps). But in the measure of average deviation error, PSNR-Y and SSIM-Y, encoded video sequences with Dirac show lower performance especially with the lower target bitrate (32, 64 Kbps) in QCIF frame coding. But the average deviation error and SSIM performance becomes comparable with that of H.264 as the target bitrate increases (512, 1024 Kbps) in CIF and HD format video sequences while PSNR performance of Dirac is lower in all cases.

Even though average deviation error from the target bitrate of the presented technique with Dirac is higher than H.264 in most of the cases, especially for lower target bitrates, the percentage of the average deviation error is always within 1% in most of the cases. So, generally speaking, the performance of the Dirac encoder with the presented rate control algorithm increases both with the video frame size and the target bitrate showing better bitrate regulation and comparable SSIM index with H.264 especially in CIF with higher target bitrate and HD encoding.

In terms of deviation error, it was found that the algorithm works more accurately in intra frame-only coding giving the deviation error which is much lower than inter frame coding mode and also found that the accuracy of the algorithm increases with the target bitrate. In terms of subjective

quality, Dirac coded frames showed some ringing artifacts at the sharp edges of the foreground object especially when the objective is moving fast. But in relatively static sequence, for example Knight Shields, the subjective quality become even better than H.264 showing all the texture information especially at the back ground of each Knight shields.

As an overall, experimental results have shown that the presented algorithm which is based on the R-QF Model shows better bitrate regulation over each GOP than rate control algorithm of H.264. Being able to regulate the bitrate according to the target bitrate is crucial in real-time multimedia data streaming in preventing buffer overflow or underflow. Moreover, unlike rate control in H.264, the algorithm is based only upon the GOP level and frame level rate control reducing the complexity dramatically since it does not require MB or basic unit level bit allocation. In addition to this, the proposed algorithm is a complete one pass process and it is not required to iterate the calculation for finding the optimum QF value. The calculation of QF is based upon the simple mathematical equation and it does not even need to calculate the QF for each frame in inter frame coding mode. The algorithm is also capable of controlling the large range of bitrates from a few to several thousand Kbps and so it is practically applicable for all types of video frame sizes (QCIF to HD) supported from Dirac. It is also applicable to the different coding modes supported from Dirac, which are intra frame-only and inter frame coding.

Chapter 5

5 Motion Estimation Strategy

5.1 Introduction

Motion estimation is extensively used in most standard video codecs as a means to exploit temporal redundancy by removing the redundant pixels in temporal domain between frames of video using the block matching method. The basic principle in the block matching method is to estimate the motion of a block in a current frame with respect to a reference frame and it is represented by a MV. The most reliable ME algorithm is the Full Search Block Matching Algorithm (FS-BMA) where prediction of MV is performed under block-by-block basis and it is widely used in the reference software as a benchmark. However, FS-BMA requires high computational complexity since it attempts to match every possible candidate using a certain type of cost function in the given search window size making it impractical for a real-time video encoding. The larger the search window size, the higher number of computation would be required. The total number of computation required is $(2w+1)^2$ where w is the size of the search window. Hence, it is required to develop faster block matching algorithm.

Generally, motion estimation is conducted in two steps: the first is integer pixel motion estimation, and the other is fractional pixel motion estimation around the position obtained by the integer pixel motion estimation. For fractional pixel motion estimation, 1/2-pixel accuracy is frequently used (e.g. in H.263, MPEG-1, MPEG-2). Higher resolution MVs which is 1/4-pixel accuracy in MPEG-4 and both 1/4, 1/8-pixel accuracy in H.264 and Dirac are adopted to achieve more accurate motion description and higher compression efficiency.

Algorithms on fast motion estimation are always hot research topic, especially fast integer pixel motion estimation has achieved much more attention because traditional fractional pixel motion estimation (such as 1/2-pixel) only take a very few proportion in the computation load of whole motion estimation. In this research as well, only the integer pixel fast motion estimation will be focused.

5.2 Related Works

Over the past decade, many fast and efficient BMAs have been proposed in order to achieve the accuracy and speed at the same time. Among them, some of the well known algorithms are the Three-Step Search (TSS) [60], the New Three-Step Search (NTSS) [61], the Four-Step Search (FourSS) [62], the Diamond Search (DS) [63], the Adaptive Rood Pattern Search (ARPS) [64], the Adaptive Irregular Pattern Search (AIPS) [65] and Fast and Robust Search [66] etc.. Most recently, Multi-Direction Cross-Hexagonal Search Algorithms was proposed [67], in which search pattern is based upon the hexagonal shape instead of using traditional square and diamond shape patterns. The goal of these algorithms is to reduce the number of search points at the expense of motion estimation

accuracy and compression efficiency. With the use of these algorithms, the number of cost function calculation is greatly reduced with a certain level of accuracy. However, most of these algorithms were tested only on the non-standard prototype model encoders using IPPPP... Group of Picture (GOP) structure and hence may not work very well for the standard full functional encoder (e.g. H.264). Again, application of these algorithms over IBBP GOP structure may not give the same results because of the temporal distance of P frame which is normally very far from its reference.

From among the fast BMAs, the adaptive search algorithms [64][65][68] become increasingly popular because of their flexibility in choosing the centre of search location adaptively. There are several methods in predicting the centre point of search location. Most commonly used methods are the spatial prediction where centre search point of current block is predicted from the adjacent left, top and top left blocks and the temporal prediction where centre point is predicted from the corresponding block of the previous encoded frame. Since there is no limitation on their search range, adaptive based search algorithms can track the global minimum quite accurately. However, for higher speed or larger displacement motion, due to the intrinsic selective nature of these methods, they often converge to a local minimum of distortion.

One method of alleviating this problem is to subsample the image to smaller sizes, such that the motion speed is reduced by the sampling ratio. The process is done on a multilevel image pyramid, known as the Hierarchical Block Matching Algorithm (HBMA) [69]. Similar to the hierarchical motion estimation, in the Multi-Resolution Motion Estimation (MRME) technique originally proposed in [70][71], the hierarchical motion estimation process is carried out in the wavelet-transform domain. It starts with estimating the MVs at the lowest resolution level, where most of the image energy resides. MVs obtained are then used as predictions for higher resolutions, where they are either accepted as final MVs or further refined. The performance of various wavelets for the task of MRME has been evaluated in [72].

In [70], motion search process is carried out only in the LL subband, which is the smallest in size among all of the resolutions, requiring a small number of coding bits for the MVs, since only the MVs in the LL subband need to be encoded. But the motion estimation perfromance is not satisfactory due to the fact that not every prediction MV from the LL subband is accurate.

In[71], MVs in the LL subband are again estimated first as in [70]. Then, they are appropriately scaled according to different resolutions, and used as the initial biases for the MVs in all the other subbands, where these initial biases are further refined. The result is in a better motion estimation performance. But it requires much higher computational load to carry out the motion refinement process and a greater number of coding bits on the motion information for all of the motion refinements in all the subbands.

MRME with indexing is proposed in [73], where a predicted MV resulting from an appropriate scaling of the corresponding MV in the LL subband is not accepted unconditionally as the final MV as in [70], nor further refined as in [71]. Instead, the SAD between the current block and the block

pointed by this prediction MV in the reference subband is calculated and compared with the coefficient weight of the current wavelet block. Coefficient weight is the combination of absolute value of the wavelet coefficients at the current block. If the SAD is less than the coefficient weight, this prediction MV is accepted; otherwise, it is discarded, and the current wavelet block is labelled as intracoded. The predicted MV whether accepted or not is indicated by one bit in a binary index. As compared to [70], the idea in [73] is able to improve the motion estimation performance substantially and the cost for the coding of motion information is quite small, as compared to that of [71]. But, instead of further refining in the higher frequency subbands when required, introducing the intra blocks would downgrade the motion estimation performance of MRME scheme compared with the other spatial-domain motion estimation techniques.

The main assumptions in the conventional block matching model for motion estimation are that the image is composed of objects in translation and that all pixels inside each block undergo the same translational motion. As a result, despite its simplicity, the conventional model is unable to represent natural scenes. Real motion has a complex nature and can be decomposed into translation, rotation, shear and expansion among other deformation components. As an alternative, two forms of spatial transformations motion estimation techniques which can handle rotation and other complex motion in the two-dimensional image plane in addition to translation have been proposed. They are Quadrilateral Matching Motion Estimation (QMEE) [74] and Warping Motion Estimation (WME) or Control Grid Interpolation (CGI) [75][76]. Both methods achieves significant prediction improvements over block matching especially in the image areas where the motion is mainly non-translational at the expense of significant added complexity compared to block matching.

Combination of hierarchical and spatial transformations to overcome the inaccuracy with large displacement motion and the increased complexity is proposed in [77]. Complexity is reduced by initializing MVs at higher levels leaving only small displacement refinements for the more computational intensive lower levels. However, requiring spatial transformation in all hierarchical levels is the added complexity in this approach. More importantly, apart from the prediction improvements, motion estimation in the spatial transformed domain cannot replace the conventional block matching models absolutely because of its simplicity and acceptable prediction results.

5.3 The Objective of the Research

As discussed in section 5.2, most of the popular BMAs [60] to [67] were tested only in the platform independent IPPPP... GOP structure and cannot be applied directly on the functional video encoder because these algorithms provide complexity reduction in basic block matching only and optimization and modification are still required in order to work for a functional encoder with any types of video formats and GOP structure. On the other hand, adaptive search algorithms can tack the global minimum quite accurately because of their flexibility in choosing the centre of search location adaptively and no limitation on their search range. However, they often converge to a local minimum

of distortion for higher speed or larger displacement motion. MRME and spatial transform approaches require additional complexities in doing their wavelet transformation in the former one and spatial transformation in the latter, which are considered unacceptable since the main objective in motion estimation is the complexity reduction. So, a complete motion estimation algorithm which can be applied to a functional video encoder with any types of video formats and GOP structure is still required.

As discussed in section 2.3, current release of the Dirac encoder [1] employs fully hierarchical motion estimation for all types of inter frames (both L_1 and L_2 or P and B frames) coding and the number of hierarchical levels depends upon the frame format. For example, for CIF video encoding, it requires 4 levels of hierarchy and HD format requires up to 6 levels. Search pattern used are the diamond shape with 61 search points for lowest resolution level search and square shape with 9 points for other levels resulting almost 80% of the total encoding time to be spent in motion estimation stage only. Even though current motion estimation strategy gives optimum result for storage application, slower encoding causes the encoder hard to apply in real-time practical application, for example in encoding of High Definition (HD) video sequences. The main objective of this research is to present the fast and efficient motion estimation strategy, called semi-hierarchical fast ME or to be short fast ME strategy, which combines modified adaptive search algorithm and semi-hierarchical approach where hierarchical motion estimation is considered only for a certain type of inter frame encoding. In modified adaptive search, more prediction points are added in order to find the most probable location as quickly as possible in the lowest resolution level search and the optimum MVs are further refined in the additional levels using four points SDSP. Semi-hierarchical or un-equal level of motion estimation is achieved by employing hierarchical motion estimation only for L_1 or P frame effectively reducing the complexity of the L_2 or B frame's motion estimation. The accuracy of B frame motion estimation can be maintained by considering the best MVs from the nearest P frame as the temporal prediction point in the modified adaptive search strategy. Overview of some of the well-known BMAs will be discussed in the next section in order to give the basic idea to the reader before the fast ME strategy can be discussed in the following sections.

5.4 Fast Block Matching Algorithms

This section discusses some of the popular BMAs' strategies and their complexities including minimum and maximum number of cost calculation required per block. Actual performance comparison was carried out by implementing their algorithms in the Matlab environment and their results including the number of cost calculation per block and the weight of the motion compensated residual frame are compared. Their performance comparison can be seen in the results and discussions section.

5.4.1 Three-Step Search (TSS)

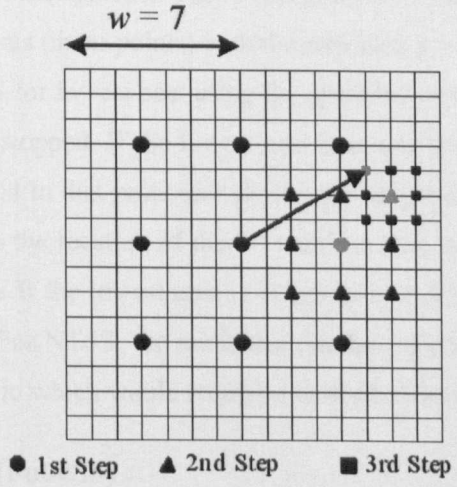


Fig. 5-1 Three Step Search Procedure, Showing the Best MV, i.e. (5, -3)

TSS [60] is one of the earliest fast BMAs. The general idea is shown in Fig. 5-1. It starts with the search location at the centre of search window with the initial step size, $s = 4$. The search pattern is rectangular. The algorithm finds the minimum cost from among 9 possible points including centre and it is set as centre for the next step. In Fig. 5-1, the point which gives minimum cost is shown in grey colour. In the second step, the step size is reduced by 2 and repeats the same procedure until the step size equal 1. The point which gives minimum cost in the last step is the best match for the centre block. The algorithm gives the flat reduction in computation by a factor of 9 compared with Full Search (FS). This mean, for search window, $w = 7$, FS would require $(2w+1)^2$ or 225 cost calculation while TSS requires to calculate only 25. In Fig. 5-1, the coordinate of the resulting MV which is shown with the arrow is (5, -3).

5.4.2 New Three-Step Search (NTSS)

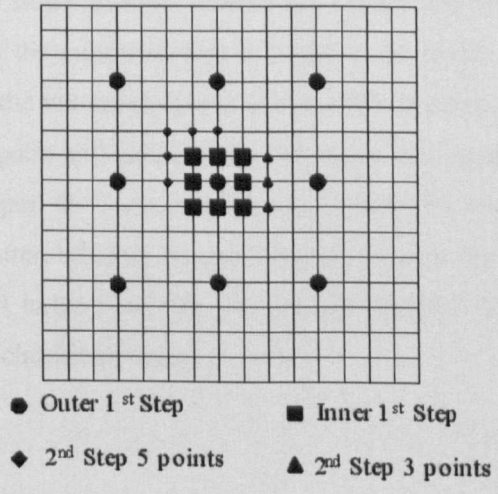


Fig. 5-2 New Three Step Search Procedure, NTSS

NTSS [61] improves on TSS by providing a centre biased searching scheme and having provisions for half way stop to reduce computational cost. The NTSS process is illustrated graphically in Fig. 5-2. In the first step, 8 additional points (inner points) with the step size, $s = 1$ are checked in addition to the 9 points in the first step of TSS for lowest cost using the specified cost function. If the lowest cost is at the origin then the search is stopped. If the lowest cost is at any points of inner 8 locations then the origin of the 2nd step is moved to that point and checks the neighbouring points adjacent to it for the lowest cost. Depending upon the location of the 2nd step's origin, neighbouring points to be checked could be either 5 or 3 points. If the lowest cost is at any points of outer 8 locations then the normal TSS procedure is followed. For NTSS, the minimum number of cost calculation required is only 17 but there is worst case scenario which would require a total of 33 locations to check.

5.4.3 Four-Step Search (FourSS)

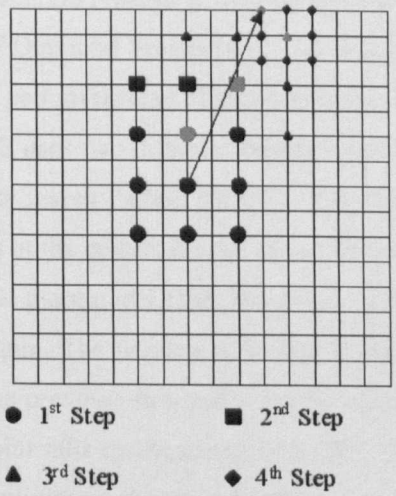


Fig. 5-3 Four Step Search, FourSS Procedure, Showing the Best MV, i.e. (3, -7)

Fig. 5-3 shows the FourSS [62] procedure. Similar to the NTSS, FourSS also employs centre biased searching and has a halfway stop provision. FourSS has a fixed step size, $s = 2$ for steps 1, 2, 3 and $s = 1$ for step 4. In any step, if the minimum cost is found at the centre of 8 points square pattern, the search jumps to 4th step. If the minimum cost is at one of the 8 points other than centre, the origin of the next step move to this point and calculate the minimum cost again. The number of points in the successive steps depends upon the location of minimum point in the previous step and it could be either 5 or 3 points. No matter, whether the minimum point is at the centre or at the edge in the 3rd step, the step size drops to 1 in the final step. This search algorithm has the best case of 17 checking points and worst case of 27 checking points.

5.4.4 Diamond Search (DS)

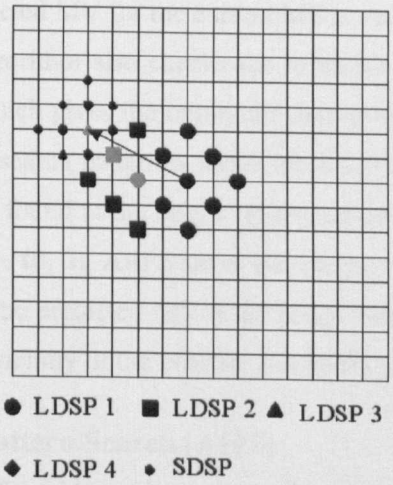


Fig. 5-4 Diamond Search, DS Procedure, Showing the Best MV, i.e. (-4, -2)

Fig. 5-4 shows the procedure of Diamond Search [63]. The idea of DS algorithm is exactly the same as FourSS except the use of DS pattern instead of rectangular one and there is no limit on the number of step the algorithm can go. DS uses two different types of patterns, Large Diamond Search Pattern (LDSP) and the Small Diamond Search Pattern (SDSP). The algorithm starts from the LDSP and at any step, if the minimum cost is at the centre; it goes to the final step with the SDSP. If the minimum cost is at one of the points on the margin of LDSP, the origin of the next step moves to this point and calculates the minimum cost again. The number of points in the successive steps depends upon the location of minimum point in the previous step and it can be either 5 or 3 points. The same procedure continues until the minimum point falls on the centre of LDSP. Since there is no limit on the number of steps, DS can find the global minimum very accurately.

5.4.5 Adaptive Rood Pattern Search (ARPS)

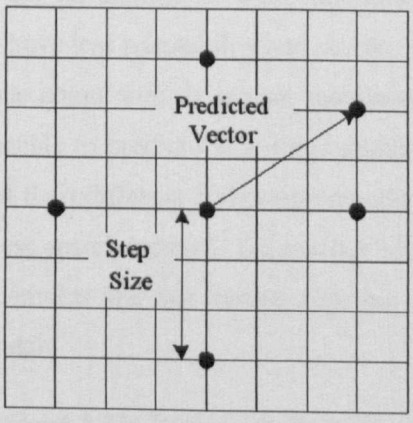


Fig. 5-5 Adaptive Rood Pattern Search, the Predicted MV is (3, -2)
and the step size is Max (|3|, |-2|) = 3

In ARPS [64], the algorithm directly starts the search in an area where there is a high probability of finding a good matching block in the first step. If the MBs around the current MB moved in a

particular direction then there is a high probability for the current MB to move in the similar direction. According to this idea, the predicted MV for the current MB is calculated and finds the corresponding cost. In addition to this, the algorithm also checks the locations of rood pattern in the first step as shown in Fig. 5-5. The point which gives the minimum cost in the first step is used as the origin for the subsequent steps where the search pattern is switched to SDSP and follows the procedure of DS until the minimum cost point is found at the origin. The main advantage of ARPS over DS is that if the predicted MV is at the centre (0, 0), ARPS saves the computational time by going directly to the SDSP without doing LDSP. If the predicted MV is far away from the centre, ARPS again saves the time by directly jumping to the vicinity of the possible minimum point and do SDSP.

5.4.6 Adaptive Irregular Pattern Search (AIPS)

In ARPS, only the magnitude of the MV of immediate left neighbouring block is used to control the arm size of the Adaptive Rood Pattern (ARP). However, the left neighbouring block is not always correlated with the current block and unavailable for the left margin blocks. Moreover, the performance of the ARPS can be further improved by considering the direction of the neighbouring blocks' MVs instead of magnitude only. Adaptive Irregular Pattern (AIP) is formed by covering the MV information from both the spatial and temporal domains. The spatial MV is predicted from immediate left, top and top right blocks' MVs. In summary, AIP has three MVs provided by the spatial (left, top and top right), their medium MV and the temporal MV. The search point of the AIP that yields the minimal matching error is used as the search centre of the followed-up refined search. Five points SDSP is used in the refined search as in ARPS.

All the fast BMAs discussed in this section more or less achieve their aim by reducing the number of cost calculation per block significantly compared with FS. In TSS, NTSS, FourSS and DS, the MV of each block is searched independently by using a fixed set of search patterns. Simplicity and regularity are the most important advantages of these methods, which make them attractive for implementation. However, they have less adaptability and search efficiency in tracking large motions. On the other hand, while the maximum number of cost calculation per block is predictable in TSS, NTSS and FourSS, it is not possible to predict the average number of cost calculation per block for DS, ARPS and AIPS because of their different style stopping criteria. Since there is no limitation on the number of steps for all of these search methods, the number of cost calculation in some blocks can be very high especially in the complex and fast motion sequences even though their search can find the global minimum quite accurately.

5.5 Semi-Hierarchical Fast ME Strategy for Dirac

In the existing ME search strategy of Dirac [1], even though the algorithm can give the estimated MVs with a certain level of accuracy, the whole process takes too long to complete because of the usage of multiple levels of hierarchies for all types of inter frames, i.e. for both P and B frames. For

example, encoding a CIF format video sequence would require the algorithm to generate four levels of hierarchy for both current and reference frames. 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 shape with the search range 5 and all other levels higher than the lowest use square shape search pattern with search range 1. So, there are altogether 61 search points in Diamond shape and 9 points in square shape. The algorithm searches the optimum MV in each block according to the diamond pattern as shown in Fig. 2-3 (a) for lowest resolution level and refines these MVs in the additional levels using square pattern as shown in Fig. 2-3 (b). After completing all these four levels, the final search is carried out again in the original frame level itself with the square search pattern. Obviously, it is the most time consuming stage and requires approximately 80% of total encoding time. So it is required to find the faster ME search strategy, which gives the same accuracy without introducing any additional complexity to the encoder.

In the fast ME strategy, the idea of adaptive search is combined with the power of hierarchical motion estimation in order to achieve the flexibility in choosing the centre of search location and at the same time prevent from being trapped in the local minimum. The overall complexity is further reduced down by introducing the idea of semi-hierarchical motion estimation.

Adaptive search method normally consists of two sequential search stages: (1) initial search and (2) refined local search. For each MB, the initial search is performed only once at the beginning in order to find a good starting point for the follow-up refined local search. By doing so, unnecessary intermediate search and the risk of being trapped into local minimum matching error points could be greatly reduced in long search case.

Basically, in pattern based motion estimation, a small search pattern made up by compactly spaced search points is more suitable than a large search pattern containing sparsely spaced search points in detecting motions with smaller displacement, because only a small number of positions around the search window centre are necessary to be checked. However, when searching for a large motion, the small pattern tends to be trapped into local minimum along the search path and leads to wrong estimation. On the contrary, the large search pattern has the advantage of quickly detecting large motions, but it will incur unnecessary search for a case where the displacement of the motion is only a few pixel away. So, instead of using larger pattern in order to track the larger displacement motion, motion estimation in the adaptive search strategy directly starts the search in an area where there is a high probability of finding a good matching block in its initial search. After that, local refined search is carried out by using one of the small search patterns, either square, diamond or hexagon.

As discussed in section 5.4.5, if the MBs around the current MB moved in a particular direction then there is a high probability for the current MB to move in the similar direction. Based upon this idea, the predicted MV for the current MB, which will be used as a most probable starting point in the initial search stage is calculated and finds the corresponding cost. In the fast ME strategy, more

predicted MVs are added at the initial search in order to find the most probable minimum point as quickly as possible. They are zero, three spatially predicted MVs (SPMV1, SPMV2 and SPMV3) and temporally predicted MVs.

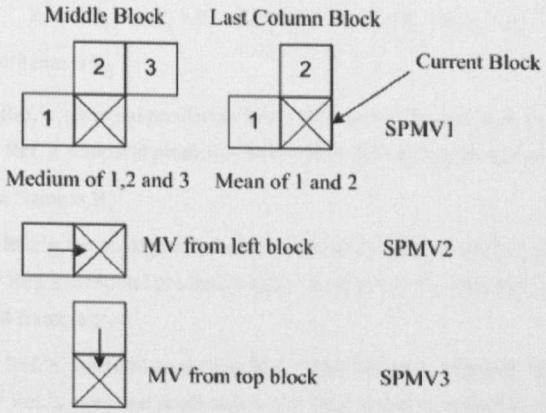


Fig. 5-6 Three types of Spatially Predicted MVs,
Current Block is the Block where ME is Being Carried Out

Here, zero MV is considered as one of the predicted MV in order to track the stationary or near stationary objects. Fig. 5-6 shows three types of spatially predicted MVs used in the fast ME strategy. SPMV1 is the spatially predicted MV used in the existing ME algorithm of Dirac and can be predicted by taking medium of neighbouring blocks 1, 2 and 3 if the current block is in the middle of the frame and mean of block 1 and 2 if the current block is in the last column of the frame. The other two spatially predicted MVs, SPMV2 and SPMV3 which are the best MVs from left and top blocks respectively, are added in order to get the better prediction for the horizontal and vertical camera panning. The last MV which is predicted temporally is added to the list in order to exploit the temporal redundancy of the video sequence. This is the vector resulting from the motion estimation of the previous successive frame at the corresponding block location.

Fig. 5-7 shows the prediction of temporal MV for the different types of frames either P or B. Note that there is no temporal predicted MV available for the first P and B frames. Again, the best MVs from the P frame cannot be used as the temporal predicted MV for the successive B frame since the prediction structure of the P and B are different as shown in Fig. 5-7. Furthermore, the prediction of the temporal MV for B frames requires scaling up or down since the temporal distances to the references for a particular reference type (either reference 1 or 2) are different for B frames. For example, in the temporal MV prediction of B_2 frame, the temporal distance for B_1 to its 1st reference, which is I_1, is 1 but the distance for B_2 to its 1st reference, which is also I_1, is 2. So, it is required to multiply the best MV of B_1 to its 1st reference by 2 in order to get the proper predicted temporal MV for the 1st reference of B_2. The same reason applies for the 2nd reference. It is important to note that in the fast ME strategy, the temporal predicted MV is applied only to the level 0 motion estimation and available for any frame types.

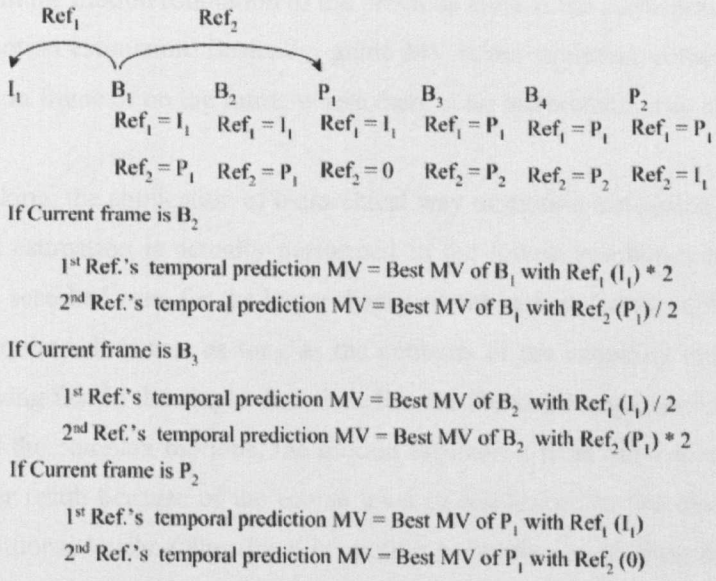


Fig. 5-7 Temporal MV Prediction for P and B frames

The performance of the adaptive search strategy heavily depends upon the accuracy of the predicted MVs used in the initial search step. Currently, predicted MVs are mainly calculated based upon the spatial and temporal domain. So, if there is an error in the motion estimation of the previous block (spatial) or previous frame (temporal), error propagation likely to be occurred since the predicted MVs of the current block is calculated based upon these information. In this case, local refined search stage cannot help because it is intended only for the MVs refinement using small search pattern. In order to avoid this, it is required to introduce one more predicted MV independent from both spatial and temporal information.

It is well known that hierarchical method of motion estimation is low in computational cost and provides robust MV predictions, especially at larger block sizes. Moreover, it can also track the higher speed or larger displacement motion because of the sub-sampling affects which effectively reduce down the motion speed by the sampling ratio. It is really important because using only adaptive search strategy can still lead to a local minimum of distortion especially when the object belongs to the current block is moving very fast. It is because the predicted MV for the initial search is calculated based upon the adjacent blocks where the objects belong to these block can be relatively stationary resulting inaccurate predicted MV for the current block. So, combining the idea of hierarchical motion estimation together with adaptive search strategy would solve all the problems mentioned above. Introducing the hierarchical way of motion estimation gives one more predicted MV which is called guide. It is the result of motion estimation on the lowest resolution frame and can be used as one of the predicted MV for the adjacent levels at the corresponding block location. It can help to prevent the error propagation since it is independent from other predicted MVs. So, now there are altogether six predicted MVs in the fast ME strategy which are zero, three spatially predicted MVs (SPMV1, SPMV2 and SPMV3), temporally predicted and guide MVs. Where, the exact definition of guide MV

is the best MV from the motion estimation of the previous level at the corresponding block location of the hierarchical motion estimation. Basically, guide MV is not available in the motion estimation of the lowest resolution frame or on the frame where there is no hierarchical way of motion estimation is employed.

Generally speaking, the application of hierarchical way of motion estimation reduce computational cost since motion estimation is actually performed in the lowest resolution level resulting smaller window size to be searched even for the larger displacement motion. In the additional levels, only the refining step is required. It is true as long as the contents of the encoding video sequence involves larger objects moving fast in the single direction. But for the video sequence which involves smaller objects moving in the complex motions, the motion estimation from the lowest resolution level will not give the proper result because of the coarse level of resolution. In this case, refining only is not enough in the additional levels. Other form of motion estimation apart from refining the MVs from the previous level would be required in order not to miss the fine details movement of the smaller objects. Instead of relying only on the MVs from the previous level and refining them, six predicted MVs mentioned above are now distributed systematically to the different level of hierarchies. So, in the intermediate levels of combined system, apart from the guide MVs, there will be two or more predicted MVs involved in the refining stage. This will be the added complexity in the combined method of adaptive search and hierarchical motion estimation.

In order to reduce the level of complexity, the hierarchical motion estimation is employed only for P frame. The idea of un-equal level of motion estimation for P and B frames or semi-hierarchical motion estimation comes from the following facts. According to the nature of GOP structure, the reference for P frames are typically far away (e.g. three to six frames in temporal separation for IBBP GOP structure) from the current frame and clearly it is unlikely to find the best match near the vicinity of the current block. So, it is required either to increase the search range or to introduce the hierarchical way of motion estimation. Another factor is that the quality of the P frame plays an important role in getting the lower residual error weight in the motion estimation of B frames. The only way to maintain the quality of the P frame without losing the compression efficiency is to increase the accuracy of the motion estimation in order to reduce the residual error weight. Removing the hierarchical way of motion estimation on B frame does not affect much on its motion estimation accuracy because of the application of temporal predicted MV from the adjacent P or B frames.

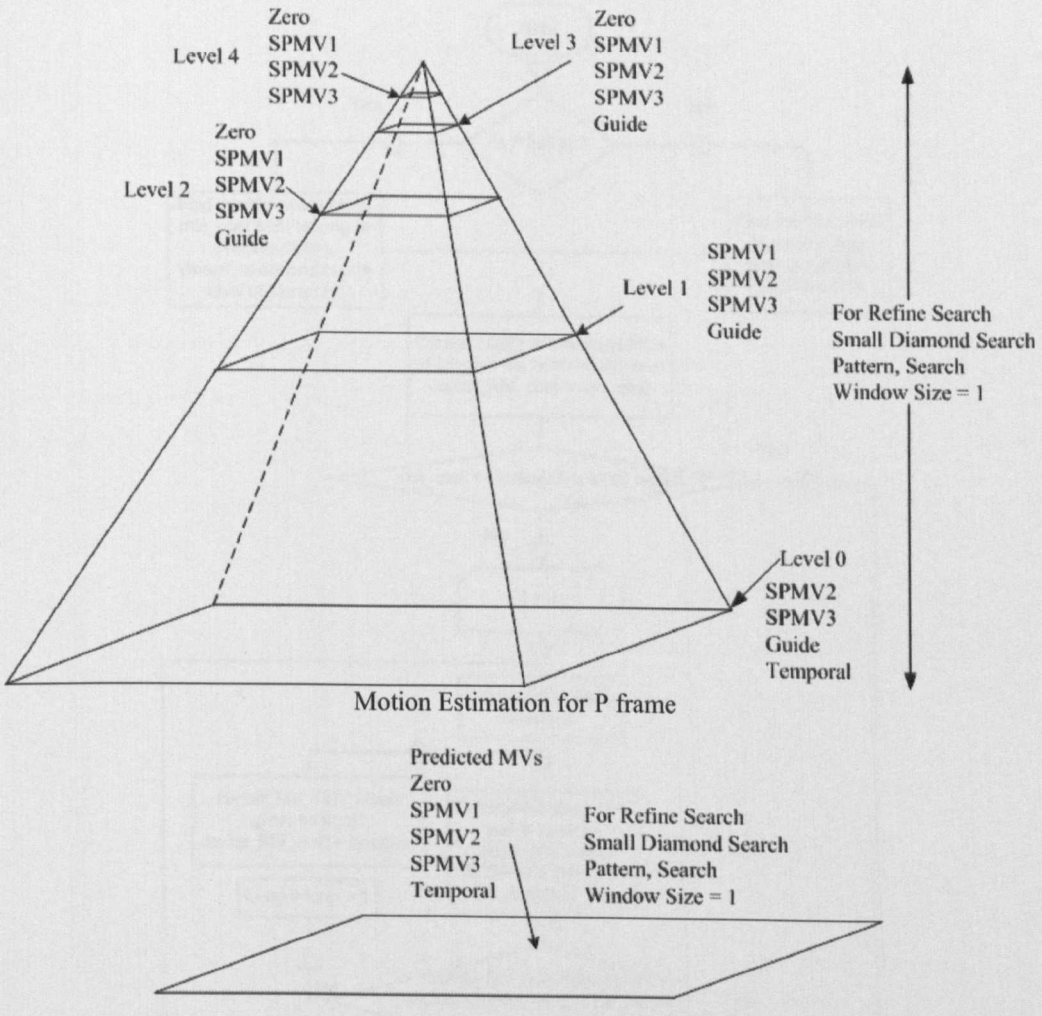


Fig. 5-8 Semi-Hierarchical Fast ME for CIF Video Format Showing All Predicted MVs in Each Level

Fig. 5-8 (a) and (b) show the semi-hierarchical fast ME strategy of P and B frames for CIF format. Level n (where $n = 4$ for CIF video format in Fig. 5-8 (a), which can be calculated using equation (2.1)) or the lowest resolution level starts with four predicted MVs in P frame since the rest two predicted MVs (i.e. guide and temporal) are not available. Guide MV is added to the predicted MV list starting from levels (n-1). It is the best MV at the corresponding block location of previous adjacent level multiplied with scaling factor 2 since the sampling ratio of the hierarchical motion estimation is 2, i.e. frame dimensions (both height and width) are reduced by 2 in each level of hierarchy. Zero MV is removed starting from level 1 since it is not required to search the MV of stationary object in all levels. In level 0, SPMV1 is replaced with temporal MV leaving only four essential predicted MVs in the highest resolution frame level. For one level B frame search in Fig. 5-8 (b), it includes altogether five predicted MVs except guide since there is no hierarchical way of motion estimation.

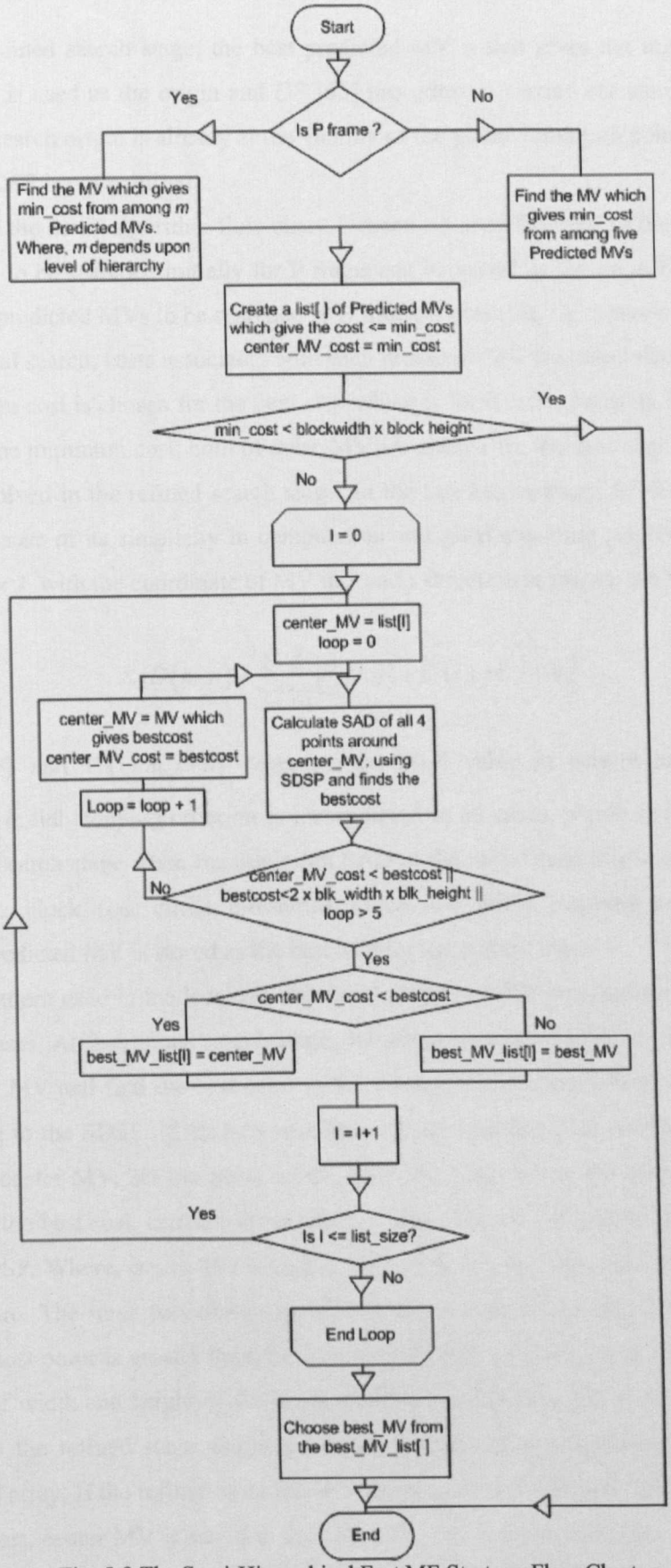


Fig. 5-9 The Semi-Hierarchical Fast ME Strategy Flow Chart

At the local refined search stage, the best predicted MV which gives the minimum cost at the initial search step is used as the origin and DS [63] procedure is carried out using only SDSP. It is assumed that the search origin is already at the vicinity of the global minimum point and so the search beginning with LDSP is not required.

Fig. 5-9 shows the detail algorithm flow chart. Depending upon the level of hierarchy, the number of predicted MVs to be searched initially for P frame can be varied as shown in Fig. 5-8 (a). But the initial number of predicted MVs to be searched in B frame is constant, i.e. 5 predicted MVs excluding guide. At the initial search, costs associated with each predicted MV are calculated and the one which gives the minimum cost is chosen for the next step which is local refined search. If the two predicted MVs give the same minimum cost, both of these MV are chosen for the next step so there can be one or more MV involved in the refined search stage. In the fast ME strategy, SAD will be used as the cost function because of its simplicity in computation and good matching performance. Assuming a block V of size $I \times J$ with the coordinate of MV in x and y direction as (m, n), the SAD is defined as

$$SAD(m,n) = \sum_{j=1}^{J} \sum_{i=1}^{I} \left| V_c(i,j) - V_r(i+m, j+n) \right| \tag{5.1}$$

where, $V_c(i,j)$ and $V_r(i+m,j+n)$ represent the pixel value in current and reference block, respectively. An initial stopping criterion is incorporated in all cases, which allows the algorithm to skip the refined search stage when the minimum SAD at the initial search is less than the number of coefficients in a block (i.e. initial threshold). Once this initial stopping criterion is met, the corresponding predicted MV is stored as the best MV for the current block.

The search pattern used in the local refined search stage is SDSP with search window, w which is set to 1 for all cases. At the refined search stage, the chosen predicted MVs are extracted one by one, set as the centre MV and find the best costs or the minimum cost from among the four surrounding points according to the SDSP. If the best cost from the surrounding four points is less than the cost given from the center MV, set the point which gives the best cost as the centre, update the center MV's cost with the best cost, increase the number of loops by one and search again its surrounding 4 points using SDSP. Where, center MV's cost is the minimum cost from the initial search step if the loop count is zero. The same procedure continues until the refined stopping criteria is met, which is either the best cost point is greater than the centre MV's cost or best cost is less than two times the multiplication of width and height of the block (refined threshold) or the number of looping is more than five. Once the refined stage stopping criteria is met, the corresponding MV is saved to the best_MV_list[ ] array. If the refined stopping criteria is met because the cost given from center MV is less than best cost, center MV is saved in the best_MV_list[ ]. Otherwise, best MV corresponding to the best cost is saved. The looping continues until there are no more MVs to be set as the centre MV

in the list to be searched. Finally, by comparing their corresponding cost, the best MV for the current block is chosen from the best_MV_list[ ].

5.6 Existing Motion Estimation Algorithm in H.264

Similar to former video standards such as H.261, MPEG-1, MPEG-2, H.263, and MPEG-4, H.264 is also based on hybrid coding framework, inside which motion estimation is the most important part in exploiting the high temporal redundancy between successive frames and is also the most time consuming part in the hybrid coding framework. Specifically multi prediction modes, multi reference frames, and higher MV resolution are adopted in H.264 to achieve more accurate prediction and higher compression efficiency. As a result, the complexity and computation load of motion estimation increase greatly in H.264 and the motion estimation can consume 60% (1 reference frame) to 80% (5 reference frames) of the total encoding time of the H.264 codec. Many fast search algorithms have been proposed to speed up ME in H.264 over the last decade. In this section, overview of the fast motion estimation strategies which are already adopted in H.264 reference software [22] will be discussed. They are (1) Spiral Search [78] (2) Hybrid Unsymmetrical-cross Multi-Hexagon-grid Search (UMHexagonS) [79] (3) Simplified UMHexagonS [80] and (4) Enhanced Predictive Zonal Search (EPZS) [81]. After that the implementation of the semi-hierarchical fast ME strategy on H.264 encoder will be discussed in detail.

5.6.1 Spiral Search (SS)

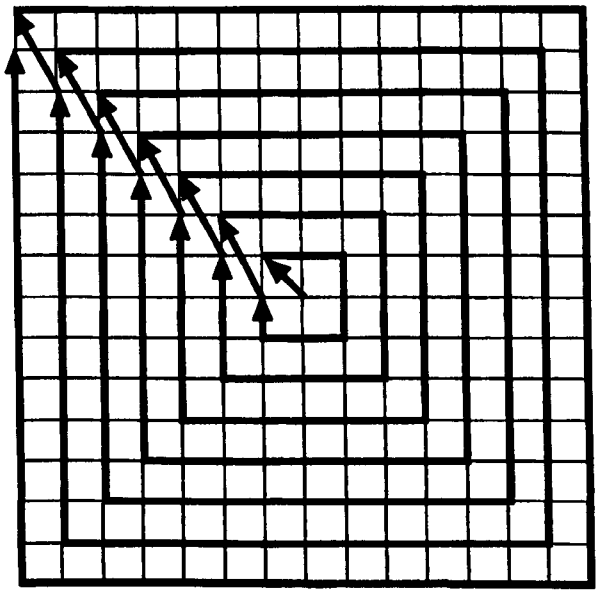


Fig. 5-10 Spiral Search Scanning Path

Spiral Search (SS) [78] algorithm is similar to the FS algorithm even though it is computationally less intensive than FS. The search begins at the origin checking point and then moves outwards with a spiral scanning path, as shown in Fig. 5-10.

```
/* Pseudo Code of the Spiral Search Algorithm*/
/* Selecting block of the frame */
for (x = 0; x < N/B; x++)
    for(y = 0; y < M/B; y++)
    {
        /* Spiral Index */
        for (l = 1; l <= MAX_Spiral_index)
            /*Block index into the Spiral*/
            for (k = 1; k <= MAX_Number_Block_in_Spiral)
            {
                for (m = 0; m < B; m++)
                    for (n = 0; n < B; n++) /* For all pixel of the block*/
                    {
                        Calculate_SAD();
                        if (SAD > SAD_previous) break;
                    }
                    step++;
                    Calculate next k and l;
            }
    }
```

The pseudo code of spiral search algorithm consists of a number of nested loops. The first two external loop indexes divide the frames into $B \times B$ blocks size, where N and M are the width and height of the frame. The loop with index l refers to the spiral index and in Fig. 5-10, maximum spiral index is 7. The k-loop describes the location of the block within the spiral and its maximum value depends upon the spiral index. For example, maximum number of block in spiral for spiral index 1 is 8. SAD is computed from all pixels of the $B \times B$ block. If the accumulated SAD value for the current block becomes higher than the previous best SAD, the SAD calculation for this block is stopped because a better match cannot be found in this block. The algorithm continues to calculate the SAD of block locations in each spiral until the outermost spiral has been reached and stores the block location which gives best SAD as the best MV for the current block.

5.6.2 Hybrid Unsymmetrical-Cross Multi-Hexagon-Grid Search (UMHexagonS)

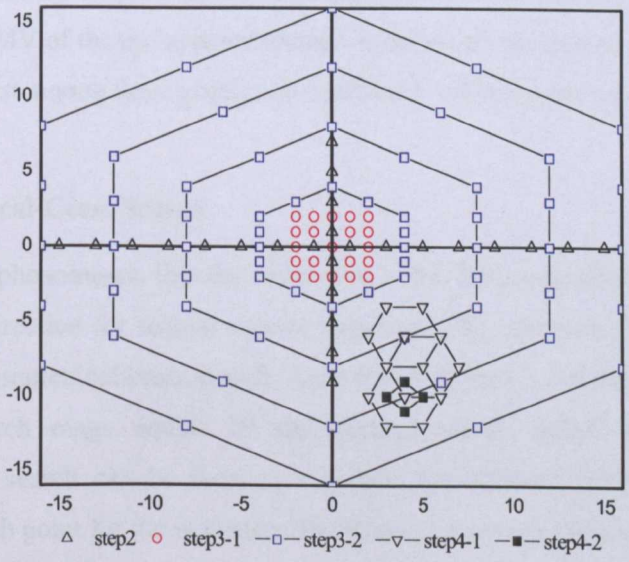


Fig. 5-11 Search Process of UMHexagonS Algorithm, $W = 16$

UMHexagonS [79] is called hybrid because it includes four steps with different kinds of search patterns: (1) initial search point prediction, (2) Unsymmetrical-cross search, (3) Uneven Multi-

Hexagon-grid search and (4) Extended hexagon based search. Fig. 5-11 demonstrates a typical search procedure in a search window with search range equals 16. It is assumed that the initial search point is (0, 0) vector in this figure.

5.6.2.1 Initial Search Point Prediction

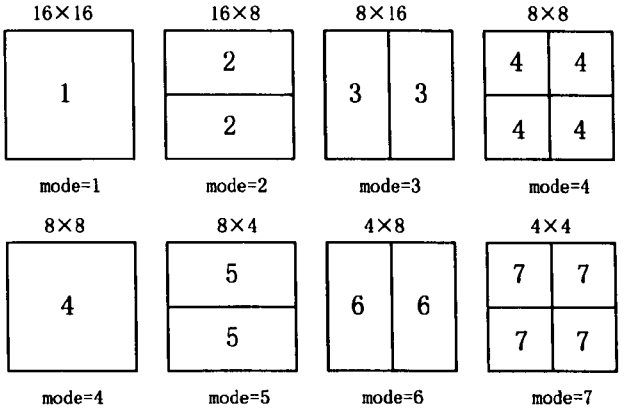


Fig. 5-12 Seven Prediction Modes in H.264

In initial search point prediction, the median value of the adjacent blocks on the left, top, and top-right (or top-left) of the current block is used to predict the MV of the current block. As shown in Fig. 5-12, there are seven inter prediction modes defined in H.264 and a hierarchically search order from mode 1 to 7 is used in the motion estimation of the current version of reference software, JM11 [22]. The MV of the up layer block (for example, mode 5 or 6 is the up layer of mode 7, and mode 4 is the up layer of mode 5 or 6, etc.) is used as one of the prediction candidates for the lower layer. Therefore for mode 1, the median prediction, the (0, 0) vector and the MVs of adjacent blocks on the left, top, and top-right are chosen as the prediction candidates; and for other modes, the median prediction, the (0, 0) vector and the MV of the up layer are chosen as the vector prediction candidates. The prediction with the minimum cost among these prediction candidates will be chosen as the initial search position of next search step.

5.6.2.2 Unsymmetrical-Cross Search

Based on a common phenomenon that the movement in the horizontal direction is much heavier than that in the vertical direction for natural picture sequences, the optimum MV can be predicted more accurately by an unsymmetrical-cross search. As in Fig. 5-11 step 2 indicates, an unsymmetrical-cross with horizontal search range equals W and vertical search range equals $W/2$ is used. The unsymmetrical-cross search can be seen as a simple but efficient prediction method to give an accurate staring search point for the next step. The distance between search points is chosen to be 2 in this strategy. For some special sequences with heavier vertical motion, the vertical search range can be expanded to W. The MV with the minimum cost will be chosen as search centre, i.e. the starting search point, of next search step.

5.6.2.3 Uneven Multi-Hexagon-Grid Search

There are two sub-steps in this step: first, a FS with search range equals 2 is carried out around the search centre, as shown in Fig. 5-11 step3-1. And then a Multi-Hexagon-grid search strategy is carried out, as in Fig. 5-11 step3-2 shows. This search strategy is based on the consideration that the unsymmetrical-cross search will give an accurate starting search point and the uneven Multi-Hexagon-grid is used to handle the large and irregular motion cases. Considering the fact that horizontal motion is much heavier than vertical motion for natural video, a Sixteen Points Hexagon Pattern (16-HP) is used as the basic search pattern. And then the uneven Multi-Hexagon-grid is constructed by extend 16-HP with different scale factors (from 1 to $W/4$) and the search process starts from the inner hexagon to the outer hexagon. The best MV derived in this step will be chosen as the search centre, i.e. the starting search point, of the next search step.

5.6.2.4 Extended Hexagon-Based Search (EHS)

The multi-grid search may obtain optimum different accuracy of MVs according to the distance between the search window centre and the search point. When the optimum MV in the previous step locates in the outer concentric area, the search result has relatively low accuracy and MV refinement by some centre biased search method is adopted. And for small prediction mode (such as mode 7), the search strategy can go directly to this step by skipping over the step 2 and step 3, because the predicted MV for small prediction mode is accurate enough. EHS which is the centre biased search algorithm used in this step is derived from Hexagon-Based Search (HEXBS) [6]. The only difference between EHS and HEXBS is that when switching the search pattern from a larger to a smaller size of hexagon, the search will continue until the Minimum Block Distortion (MBD) point is the centre of the newly formed hexagon as shown in Fig. 5-11 step-4. The algorithm stops when the MBD point is at the centre of the newly formed hexagon and the centre point is stored as the best MV for the current block.

5.6.3 Simplified UMHexagonS

MV's correlation is rather high in spatial domain because of the consistency of object and also in time domain because of consistency of motion, on which Simplified UMHexagonS [80] method is based. Considering multi prediction modes in Fig. 5-12 and multi reference frames, correlations of spatial and time domain can be described as follow. For spatial correlations, the correlations can be between neighbouring blocks in the same mode or between neighbouring modes in the same block. And in time correlations, the correlation can be between neighbouring frames with the same reference frame or between neighbouring reference frames with the same frame. Based upon the idea of these four correlations, the corresponding predictions which are Median Prediction (MP), UpLayer Prediction (ULP), Last Frame Prediction (LFP) and Last Reference Frame Prediction (LRFP) are considered in Simplified UMHexagonS method.

The predicted MV in MP is the medium of MVs from left, top and top right blocks. For UpLayer prediction, the hierarchical search order from mode 1 to 7 as shown in Fig. 5-12 is chosen as the mode search order and the MV of the up layer block is used as one of the prediction candidates of lower layer. For mode 2 and 3, their up layer is mode 1; for mode 4, its up layer mode 2; for mode 5 and 6, their up layer is mode 4; for mode 7, its up layer mode 5 and there is no up layer for mode 1. In LFP, the MV of the corresponding block in the last frame is used as one of the predicted MV. The predicted MV in the LRFP is the scale version of the corresponding MV of the last reference frame in the same mode.

In simplified version of UMHexagonS, all these predicted MVs are added in the step 1 of UMHexagonS and then search the surrounding four points of each prediction like step 4-2 in Fig. 5-11. The early termination strategy is applied in different steps of UMHexagonS. Fig. 5-13 shows the whole algorithm's flow chart of Simplified UMHexagonS in which blocks with bigger font size are the added extra stages to UMHexagonS.

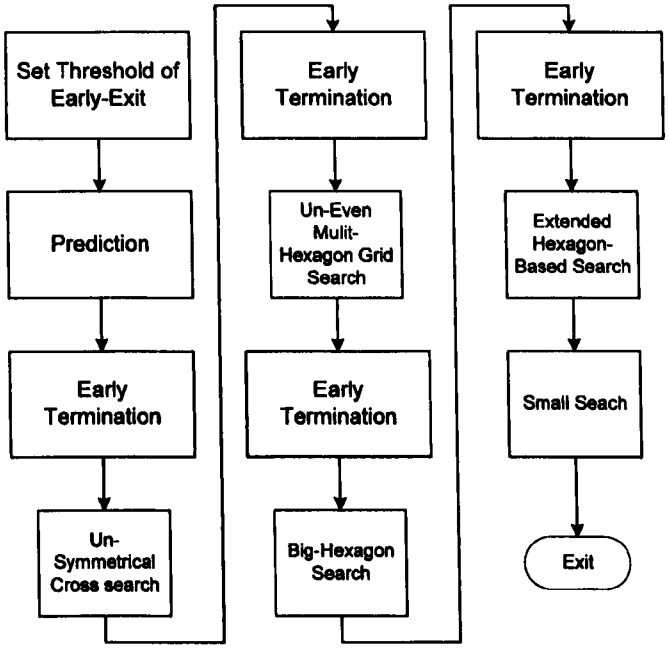


Fig. 5-13 Flow Chart of the Simplified UMHexagonS

Thresholds of the early termination are defined also with the same idea as in predicted MV calculation. Similar to MVs, SAD of spatial adjacent or temporal adjacent blocks should also have a high Correlation. Based upon this idea, predicted SAD threshold using Median Prediction (SAD_{pred_MP}), UPlayer Prediction (SAD_{pred_ULP}) and, Last Ref-frame Prediction (SAD_{pred_LRFP}) are defined. Predicted SAD threshold using MP is defined by

$$SAD_{pred\_MP} = \min\left\{SAD_{Vx\_median}, SAD_{Vy\_median,}\right\} \tag{5.2}$$

Where, SAD_{Vx_median} is the SAD value contributed from the x coordinate part of the MV which is median of the MVs of left, top and top right blocks. The same definition applies for the SAD_{Vy_median}. Predicted SAD threshold using ULP is defined by

$$SAD_{pred\_UP} = SAD_{UpLayer} / 2 \tag{5.3}$$

SAD_{pred_LRFP} is the SAD of corresponding block location in the last reference frame.

The early-termination strategy is

if (SAD_{curr} - SAD_{pred} < $SAD_{pred} \times \beta_1$)

go to step 4-1

else if (SAD_{curr} - SAD_{pred} < $SAD_{pred} \times \beta_2$)

go to step 4-2

where SAD_{curr} is the SAD of current MV candidate and

if (reference frame number > 0)

SAD_{pred} adopts SAD_{pred_LRFP}

else if (mode > 1)

SAD_{pred} adopts SAD_{pred_ULP}

else

SAD_{pred} adopts SAD_{pred_MP}

Where β_1 and β_2 are modulating factors and detail explanation of their calculation can be found in [80].

5.6.4 Enhanced Predictive Zonal Search (EPZS)

EPZS [81], similar to other predictive algorithms, mainly comprises 3 steps. Initial predictor selection selects the best MV predictor from a set of potentially likely predictors. Adaptive early termination allows the termination of the search at given stages of the estimation if some rules are satisfied, while the prediction refinement employs a refinement pattern around the best predictor to essentially improve the final prediction.

5.6.4.1 Predictor selection

Predictors are mainly based on the spatial and temporal correlations such as the Median predictor, the three spatially adjacent MVs (left, top, top-right), and temporal predictors such as the MV of the co-located and its surrounding blocks and the co-located acceleration predictor. Stationary was also considered through the consideration of the (0, 0) MV predictor. The previous calculated best MV, MV_1 for the current block pointing to a reference frame with temporal distance TD_1, can he used as a predictor, MV_2 for a different reference frame with temporal distance TD_2 to exploit temporal correlation that may exist between the two frames even further. MV_2 is calculated as $(TD_2 \times MV_1)/TD_1$.

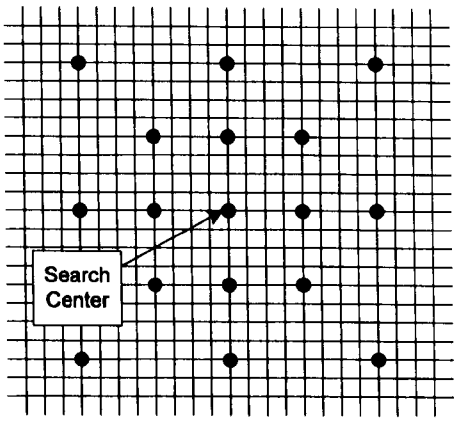


Fig. 5-14 Additional Fixed Predictors Grid for Search Range ±8

Although the above predictors could be sufficient in most cases, sequences with random or complex motion at low frame rates and at very low quality might still be insufficient for the prediction and could lead to poor performance. An additional set of predictors, which is search range dependent that are equally or logarithmically spaced around a prediction centre (e.g. the median or zero) as shown in Fig. 5-14 is added.

5.6.4.2 Adaptive Early Termination

Similar to MVs, distortion of adjacent blocks tends to be highly correlated. Considering this fact, several early termination criteria based on the distortion of adjacent blocks are introduced. After examining the median predictor, if its distortion is smaller than a threshold T_1, the ME process can be terminated immediately. This threshold is set equal to the number of pixels of the examined block type. The search can also be terminated if a different reference previously satisfied this threshold. Otherwise, all other predictors are considered. A second threshold T_2 is defined as:

$$T_2 = a \times \min\left(MinJ_1, MinJ_2, ..., MinJ_n\right) + b \tag{5.4}$$

where a and b can be fixed values (a=1.1, b=T_1) and $MinJ_1$, $MinJ_2$,. . . ,$MinJ_n$ correspond to the minimum distortion values of previously examined blocks. For the calculation of T_2, it is sufficient to use the 3 spatially adjacent blocks (left, top, top-right) and the co-located block in the previous frame. More specifically, if the reference frame is not 0, the block type is smaller than 8×8, or the distortion of the median predictor is smaller than T_2, the examination of the fixed predictor set is not required. Furthermore, after calculating the distortion for all predictors, if the distortion of one of these predictors is smaller than T_2, then the search is terminated immediately. To reduce the possibility of erroneous and inadequate early termination, a limit within the calculation of T_2 is introduced by considering an additional fixed distortion predictor, $MinJ_i$ within the above calculation in equation (5.4), which is set equal to $MinJ_i = 3 \times T_1$.

5.6.4.3 Motion Vector Refinement

If the early termination criteria are not satisfied, ME is refined further by using an iterative search pattern localized at the best predictor. Unfortunately, the application of SDSP and Square patterns in the refined stage is easier to be trapped into local minima areas mainly due to the introduction of RDO concepts within the ME, thus leading to reduced performance. To reduce this problem, a more aggressive pattern, an extended EPZS (ExtEPZS) pattern, as shown in Fig. 5-15 is introduced. It is actually the extended diamond pattern and in the figure, the first stage of the pattern is shown with squares and the next stage with circles. The corresponding minimum distortion point in the first stage is shown with red colour.

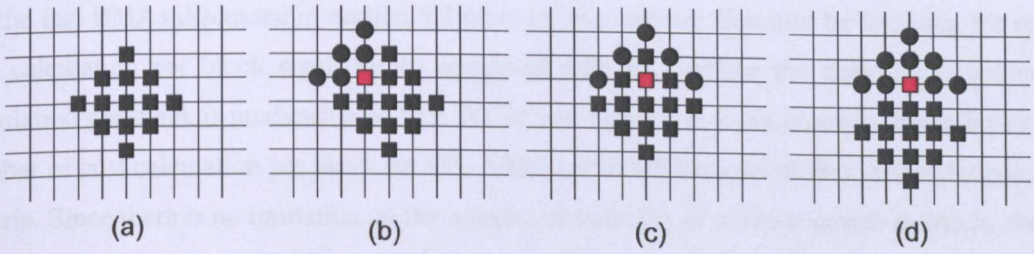


Fig. 5-15 The Extended EPSZ Pattern (extEPZS)

Even though this pattern needs to examine a larger number of checking positions, its use helps in escaping local minima positions and improving encoding performance. Moreover, considering that the larger patterns require considerably more points, a pattern selection process within the scheme is introduced. First, if the current minimum distortion is below T_1+T_2 then the MV of the best predictor is also examined. If this is equal to (0, 0) then the small diamond is used, otherwise the square pattern is selected. If distortion is larger than the above threshold, but the block type is smaller than 8×8 then the square pattern is used. In all other cases, the extended diamond is chosen.

5.6.5 The Semi-Hierarchical Fast ME for H.264

The semi-hierarchical fast motion estimation strategy is implemented in H.264 JM 11 reference software [22] in order to compare its performance with the existing ME strategies of H.264. The distribution of the predicted MVs over different levels of hierarchy for P frame motion estimation and the use of predicted MVs over B frame motion estimation are exactly the same as shown in Fig. 5-8 (a) and (b). The refined process also uses SDSP with the same early termination criteria with two thresholds.

However, the processes of pixel accuracy motion estimation between the two encoders are different resulting the implementation of the Fast ME strategy in H264 slightly different from Dirac. As shown in Fig. 5-12, there are seven inter prediction modes defined in H.264 and a hierarchical search order from mode 1 to 7 is used in the motion estimation of the current version of reference software, JM11 [22]. But in Dirac, even though there are different prediction modes and block sizes

(e.g. block, subMB and MB as shown in Fig. 2-6), pixel accuracy motion estimation is done only in block level. After going through the subpixel refined stage, motion estimation for subMB and MB levels are later performed with the subpixel accuracy either 1/4 or 1/8 before the mode decision can be made. So in Dirac, the Fast ME strategy is required to implement only for smallest block level since it is intended for pixel accuracy motion estimation only. On the other hand, H.264's pixel accuracy motion estimation involves multiple block sizes starting hierarchically from mode 1 to 7 or from biggest block size (16×16) to smallest (4×4) requiring the Fast ME strategy for pixel accuracy to go through all block levels starting from biggest one.

5.7 Partial Cost Function Calculation

All the fast BMAs discussed in section 5.4 more or less achieve their aim by reducing the number of cost calculation per block significantly compared with FS. While the maximum number of cost calculation per block is predictable in TSS, NTSS and FourSS, it is not possible to predict the average number of cost calculation per block for DS, ARPS and AIPS because of their different style stopping criteria. Since there is no limitation on the number of steps for all of these search methods, the number of cost calculation in some blocks can be very high especially in the complex and fast motion sequences even though their search can find the global minimum accurately.

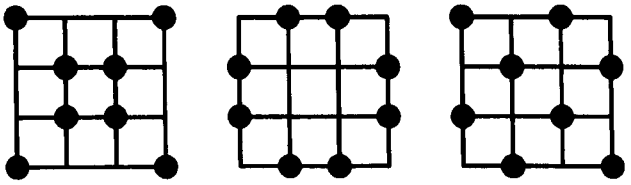
But, it is important to note that these algorithms calculate the cost on a full block basis where cost calculation is performed for each and every pixel in the block. This means the computational load can be further reduced by calculating the cost on a partial block basis. Partial cost calculation can be carried out by following a certain pixel pattern in the cost calculation instead of a whole. So, implementing the partial cost function calculation idea would be an added advantage to these types of algorithms since the combined method would achieve the accuracy and lower computational load at the same time.

Some previous works [82][78][83] on the partial cost calculation has been previously reported in the literature. Alternating between the sub-sampling patterns [82] for each checking point allows the algorithm to use all the pixels of the block within different search locations. Its performance is similar to that of exhaustive search but with computation reduced by a factor of 4 for four sub-sampling patterns. However, the algorithm reduces only the number of pixels used in the Block Distortion Matrix (BDM) instead of the number of checking points and was designed for exhaustive search only. The idea of pattern formation in Normalized Partial Distortion Search (NPDS) algorithm [78] is basically the same with alternating sub-sampling search algorithm [82] except the use of sixteen sub-sampling patterns together with halfway-stop technique giving the maximum possible speedup ratios of up to 16 times in theory. Its mean square error performance is very close to FS with the average computation reduction of 12–13 times, with respect to the full-search algorithm. In Adjustable Partial Distortion Search (APDS) [83], each sub-sampling patterns in NPDS [78] is further divided into either 16, 8, 4 etc. smaller patterns giving maximum possible speed up ratio of up to 256 times in theory.

This means that there is only one pixel per block involved in cost calculation in some case where the distortion given by a pixel is already higher than current minimum distortion. But in reality, it is very unlikely to happen and the speed up factor given in APDS [83] will largely depend on the contents of the test video sequence.

Even though all the related works mentioned above can effectively reduce down the computation from 4 to 16 times (except APDS where speed up factor depends upon test sequence) compare with FS, the patterns are designed to work with either exhaustive or spiral search only. Because of their non-uniform or irregular shape of patterns, it is not possible to apply their proposed patterns on the fast BMA mentioned in section 5.4, such as TSS, NTSS, FourSS,.. etc.

In this section, a method which is called partial cost function calculation is presented for faster cost calculation in finding the exact match for any type of BMAs including exhaustive or FS. In this method, the speed up factor depends only upon the number of pixels used in the pattern and will give the flat reduction in computation either with FS or any of BMAs. In order to get the pattern with optimum performance, a number of regular shape partial cost calculation patterns are presented. Their performance and speed are evaluated using exhaustive search technique and chooses the optimum one. Finally, the chosen pattern is implemented in some of the above mentioned well known BMAs for performance comparison against without using partial cost calculation.



(a) (b) (c)

Fig. 5-16 Partial SAD Calculation Patterns for Block Size 4×4. (a) Cross, (b) Cross Complement (Crosscom) and (c) ZigZag

The idea partial cost calculation is quite simple. For example, in an ideal case, if a block in the current frame has the exact match to a block pointed by the MV in the reference frame, no matter which ever point you choose and no matter how many point you choose, either all or a few points in a certain pattern, the calculation of SAD for each and every point would be zero and hence the total would be zero. But in real situation, the minimum cost point, which results from full SAD calculation and partial SAD calculation, may or may not be the same. That would affect the accuracy of motion estimation in partial SAD calculation but it is negligible compared to the reduction in computational load.

Fig. 5-16 (a), (b) and (c) show the possible partial SAD calculation patterns for 4×4 block size. The pattern are chosen in such a way that the points insides each and every pattern cover as much area of the block as possible. The cross pattern is considered as the main pattern because of its simplicity in terms of hardware implementation. The others patterns are chosen in order to evaluate the

performance of cross. For that reason, the number of points for the patterns in Fig. 5-16 (b) and (c) are kept the same with cross in order to have the fair comparison. The cross complement pattern in Fig. 5-16 (b) has the points which are not considered in the cross calculation. The algorithm of zigzag pattern is as follow.

```
/*Pseudo Code for Calculating SAD using ZigZag Pattern, Assume MV = 0 */
/* n is block's dimension (assume square block)*/
offset1 = 0;
offset2 = n/2;
i = 1;
for j = 1:n
        sad+=abs((currentBlk(j,(i+offset1))-refBlk(j,(i+offset1))));
        sad+=abs((currentBlk(j,(i+offset2))-refBlk(j,(i+offset2))));

        offset1 = offset1+((n/2) - 1);
        offset2 = offset2+((n/2) - 1);

        if (offset2 > (n-1))
                offset1 = 0;
                offset2 = n/2;
        end
end
```

In Fig. 5-16, the number of points included in each pattern is 8 and so it gives two fold reduction factor in partial SAD calculation compared with full. The reduction factor depends upon the number of points in a pattern and the block size as well. For example, the number of points in the cross pattern is 16 in 8×8 block size but the total number of pixels in this block is 64 giving four time reduction factor in partial SAD calculation.

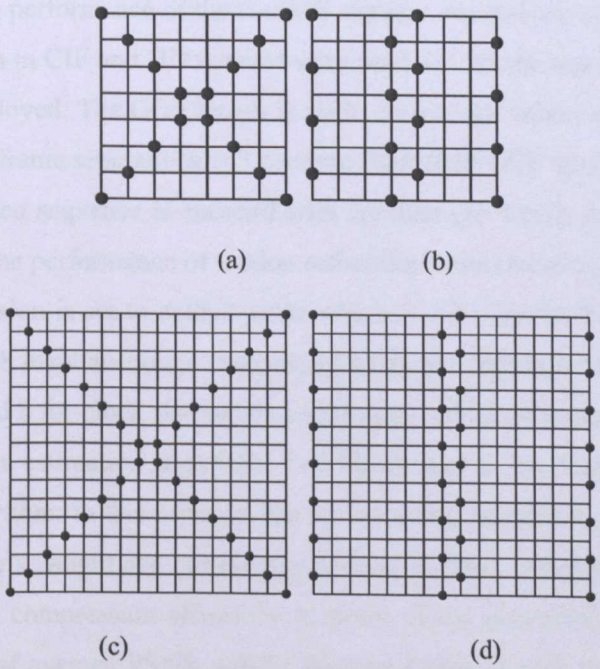


Fig. 5-17 Partial SAD Calculation Patterns (a) Cross 8×8, (b) Zig Zag 8×8, (c) Cross 16×16 and (d) Zig Zag 16×16

Fig. 5-17 (a), (b), (c) and (d) show the cross and zigzag patterns for block sizes 8×8 and 16×16, respectively. The cross complement (Crosscom) was excluded since it requires more points to

implement than cross pattern. As expected, the block size 16×16 gives even larger reduction factor which is 32/256 = 8, with given patterns.

5.8 Results and Discussions

In this part of the chapter, all the results from each section mentioned above (section 5.5 to 5.7) will be presented and discussed separately. First of all, the fast ME strategy's results with the current version of Dirac video encoder (Dirac 0.6) [3] will be compared with the existing one of Dirac. The comparison includes complexity in terms of the required number of SAD calculation per block, motion estimation accuracy in terms of the weight of the motion compensated residual error frame, compression efficiency in terms of the generated file size and objective performance in terms of PSNR. After that the performance of the adopted motion estimation strategies in H.264 JM 11 [22] will be evaluated and the best motion estimation strategy will be used to compare with the fast ME strategy which was implemented in H.264. Again, the comparison covers all the aspects mentioned above. Finally, the result of the partial distortion idea will be discussed. First, the performance of the presented patterns is compared using FS and then the best pattern chosen is implemented in some of the well known BMAs in order to evaluate the performance in terms of motion estimation accuracy and complexity reduction.

5.8.1 Semi-Hierarchical Fast Motion Estimation (Dirac)

In order to evaluate the performance of the fast ME strategy, several test sequences ranges from slow, medium to high motion in CIF and HD formats were used. As for the test platform, Dirac version 0.6 from [3] has been employed. The GOP length is set to 36 (default value) which means the number of L_1 frames is 11 and L_1 frame separations is 3 forming IBBPBBP GOP structure. Rate control function is disabled and the video sequence is encoded with constant QF which is 7 for all test sequences in order to monitor only the performance of motion estimation without having any interference from rate control part. MV precision is set to default value which is 1/4 (Quarter Pixel) accuracy even though Dirac supports up to 1/8 pixel accuracy. Overlapped block parameters are set according to the default values which are 12 and 8 for block size (width and height) and block separation, respectively.

Basically, a motion estimation algorithm can be evaluated by determining the accuracy and complexity of the algorithm. In this research, the accuracy and complexity are represented in terms of the residual error frame's weight and the average number of SAD calculation per block, respectively. In addition to this, the compression efficiency in terms of the generated file size and the objective performance in terms of average PSNR will be discussed even though these parameters are used to determine only the overall encoding performance. Table 5-1 shows the motion estimation results of both Dirac 0.6's ME and the fast ME strategies for different test sequences in CIF format. In Table 5-1, average SAD calculation per block is the average number of SAD calculation required for a block in a particular frame. It can be calculated by dividing the total number of SAD calculation count in a

frame with the number of block in this frame. Its value is again taken as average over entire encoded sequence including *I* frames so the value in Table 5-1, column 3 is actually the average SAD calculation per block per frame and can be calculated as follow.

$$\text{Avg. SAD/block} = \frac{\sum_{i=1}^{K} SAD_i}{K} \tag{5.5}$$

Where, K is the number of blocks in a frame and SAD_i is the number of SAD calculations for a block i. The average SAD calculation per block per frame can be calculated as follow.

$$\text{Avg. SAD/block/frame} = \frac{\sum_{j=1}^{L} (\text{Avg.SAD/block})_j}{L} \tag{5.6}$$

Avg. weight refers to the average residual error frame's weight for the whole sequence and the lower weight reflects the higher accuracy in the corresponding motion estimation algorithm. It is the division of the combination of absolute value of the coefficients in residual error frame by frame dimensions and the number of frames in the sequence, and can be calculated as follow.

$$Weight = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} |a_{i,j}|}{M \times N} \tag{5.7}$$

Where, $a_{i,j}$ is the pixel value at coordinate (i ,j) and M, N are frame width and height. It is for one frame and average weight is its average value over the entire sequence.

$$Avg.\ Weight = \frac{\sum_{k=1}^{L} Weight_k}{L} \tag{5.8}$$

Where, L is the encoded number of frames in a sequence. It is important to note that the average weight calculation in equation (5.8) also considers the weight of *I* frame.

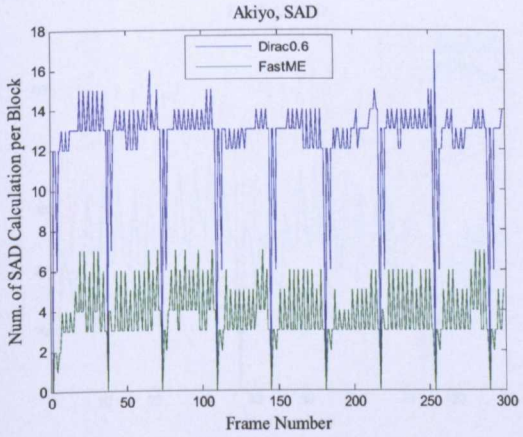
As shown in Table 5-1, the fast ME strategy gives the average weight which is slightly lower than Dirac 0.6 in all sequences except Akiyo. In terms of compression efficiency, again the fast ME strategy gives approximately equal or smaller file size for all test sequences and it is more significant especially in the Bus sequence. Reduction in file size is directly related to the average weight reduction since the weight of *I* frame in both strategies are constant. PSNR for Y component is used to compare the objective quality of the reconstructed frames. As shown in the last column of Table 5-1, all the test sequences give approximately the same value of PSNR for both strategies.

Sequence	ME Strategy	Avg. SAD/ Block	Avg. Weight	File Size (bytes)	Avg. PSNR-Y (dB)
Akiyo	Dirac 0.6	12.48	4.447486	237959	39.97
	Fast ME	3.76	4.448263	238538	39.97
Foreman	Dirac 0.6	32.46	8.240105	614303	34.63
	Fast ME	14.78	8.181149	609187	34.67
Bus	Dirac 0.6	34.33	9.104126	636695	31.50
	Fast ME	15.04	8.694348	590205	31.68
Football	Dirac 0.6	40.39	9.464432	275735	33.03
	Fast ME	23.28	9.430217	274491	32.91

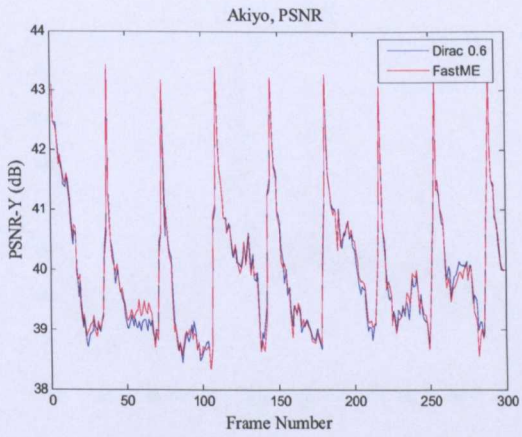
Table 5-1 The Comparison of ME Results for CIF Format, All the Values are Taken as Average Over Entire Encoding Sequence Except File Size

But there is significant improvement in fast ME strategy as far as the speed of the algorithm is concerned. There is a huge saving, at least two folds in average number of SAD calculation per block for all test sequences. Reduction in the number of SAD calculation is much more significant in relatively static sequence (e.g. Akiyo) where the required number of SAD calculation per block in the fast ME strategy is more than one third that of Dirac 0.6. It is simply because of the application of early termination method after the initial search and the use of semi-hierarchical approach in the fast ME strategy. An initial stopping criterion allows the algorithm to skip the refined search stage when the minimum SAD at the initial search is less than the number of coefficients in a block (i.e. initial threshold). In static sequences, the chances of meeting the initial stopping criterion is quite high for most of the blocks since the displacement of both background and foreground objects are not much significant between the adjacent frames. On the other hand, dynamic motions sequences (e.g. Football) require refined search stage since initial search results are not good enough to stop the algorithm for most of the blocks, requiring more number of SAD calculation compared with the less dynamic sequences. Limiting the number of SAD calculation in such sequences with refined stopping criteria gives the noticeable reduction in terms of complexity in the fast ME strategy. Application of semi-hierarchical motion estimation removes the requirement of hierarchical motion estimation in B frame coding giving added advantage in reducing the total computational load.

In Dirac, the number of reference frames to be searched for motion estimation is 2 and so the average number of required SAD calculation for one block per one reference frame is approximately half of the given values in Table 5-1.

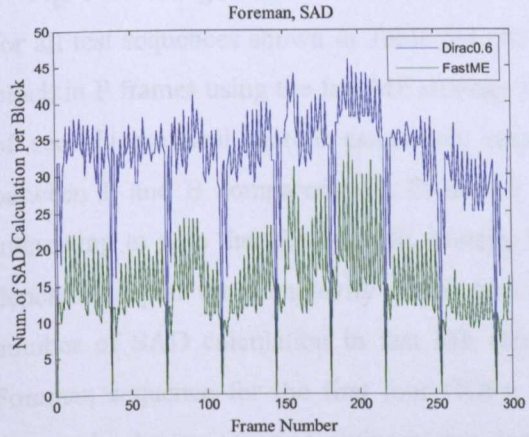


(a) Comparison of Number ofSAD Calculations per Block

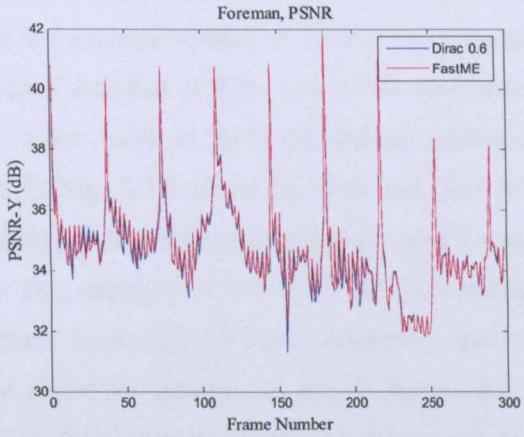


(b) Comparison of PSNR-Y

Fig. 5-18 Comparison of ME results for Dirac 0.6 and the Fast ME, Akiyo in CIF Format

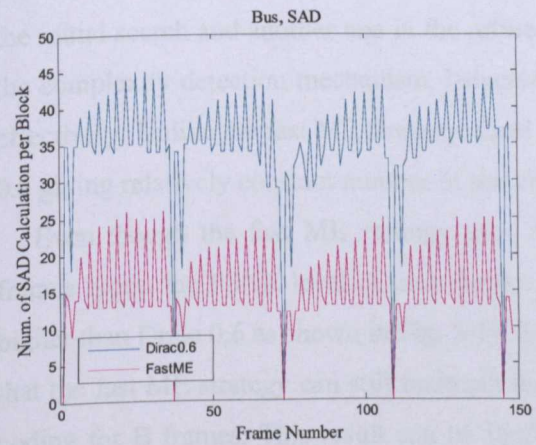


(a) Comparison of Number ofSAD Calculations per Block

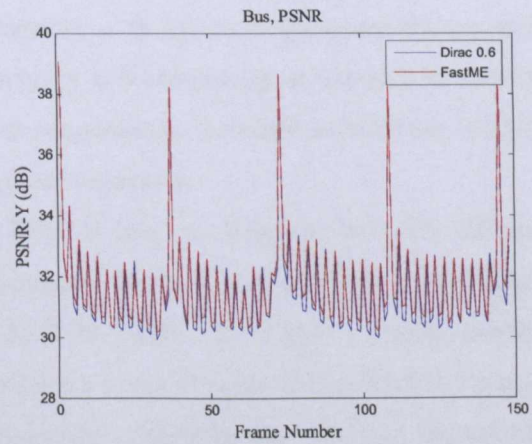


(b) Comparison of PSNR-Y

Fig. 5-19 Comparison of ME results for Dirac 0.6 and the Fast ME, Foreman in CIF Format

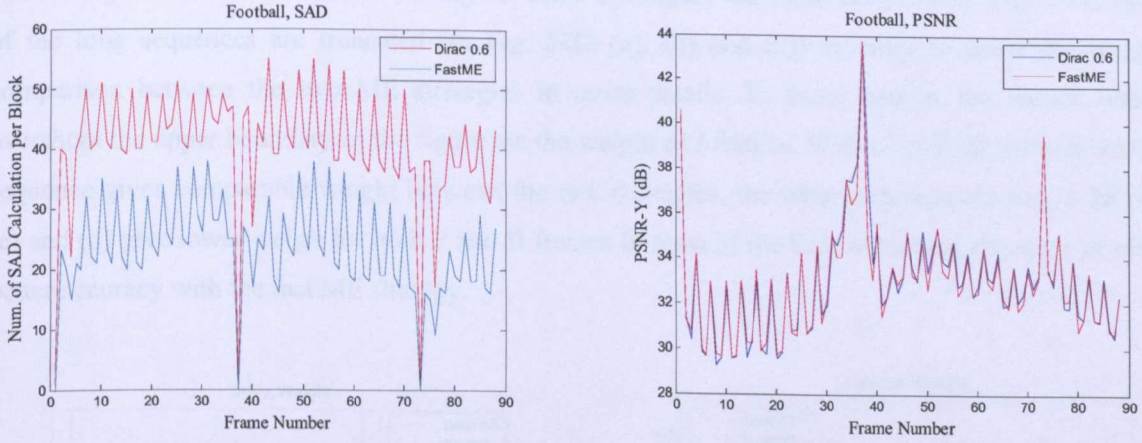


(a) Comparison of Number ofSAD Calculations per Block



(b) Comparison of PSNR-Y

Fig. 5-20 Comparison of ME results for Dirac 0.6 and the Fast ME, Bus in CIF Format



(a) Comparison of Number of SAD Calculations per Block

(b) Comparison of PSNR-Y

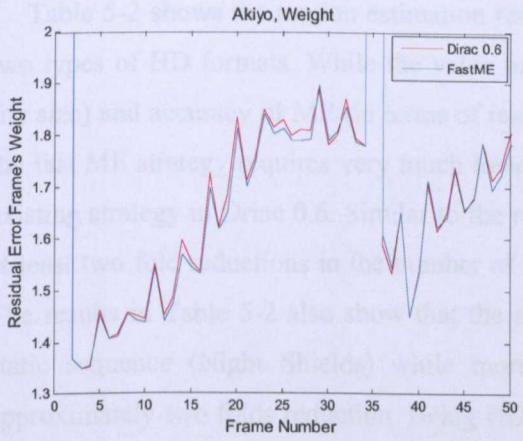
Fig. 5-21 Comparison of ME results for Dirac 0.6 and the Fast ME, Football in CIF Format

Fig. 5-18 to Fig. 5-21 show the number of SAD calculation per block and PSNR-Y of each frame for all test sequences shown in Table 5-1. As expected, the average number of SAD calculation per block in P frames using the fast ME strategy is much higher than that of B because of the application of semi-hierarchical motion estimation, resulting the wider band of SAD calculation difference between P and B compared with Driac 0.6 as shown in Fig. 5-18 (a) to Fig. 5-21 (a). But it is interesting to note that the fast ME strategy has the ability to increase or decrease its search range depending upon the complexity of the test sequence. For example, in Fig. 5-19 (a), the average number of SAD calculation in fast ME strategy is lower while the motion is relatively static in Foreman sequence for the first four GOPs. Then, the algorithm increase its search range once it detects the dynamic motion giving higher number of SAD calculation per block in the fifth and sixth GOP in order to maintain the level of accuracy in motion estimation.

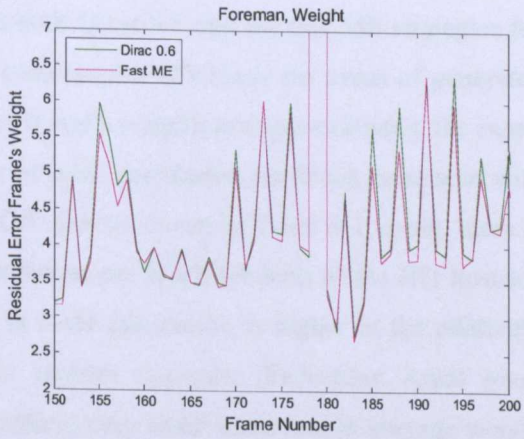
Again, the algorithm reduces its search range for the remaining frames which have less dynamic motions giving lower number of SAD calculation. The application of double thresholds system, one in the initial search and another one in the refined stage together with refined stopping criteria serves as the complexity detection mechanism, balancing the accuracy and complexity of overall ME strategy effectively. Unlike the fast ME strategy, there is no such adaptation in the motion estimation of Dirac 0.6 giving relatively constant number of search in all type of sequences.

Even though the fast ME strategy uses un-equal level of motion estimation between different frames types, the PSNR level of each frames is approximately the same and sometime even slightly higher than Driac 0.6 as shown in Fig. 5-18 (b) to Fig. 5-21 (b). PSNR results in these figures confirm that the fast ME strategy can still maintain the same accuracy even with the non-hierarchical way of coding for B frames. This result can be further confirmed by comparing the weight of the residual error frame for both ME strategies in Fig. 5-22. Generally speaking, weight can be represented as the accuracy of corresponding ME algorithm. So, getting approximately the same weights between the

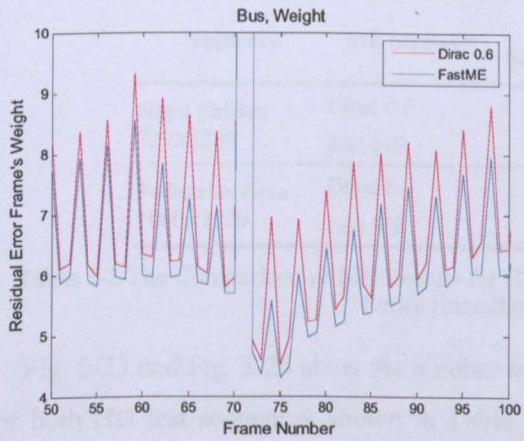
two strategies can be translated as having the same or roughly the same accuracy. In Fig. 5-22, most of the long sequences are truncated (in Fig. 5-22 (a), (b) and (c)) in order to show the weight comparison between the two ME strategies in more details. In these figures, the values which overshoot the upper boundary of the figure are the weight of *I* frames. While Fig. 5-22 (a) with Akiyo sequence gives comparable weight between the two strategies, the other sequences in Fig. 5-22 (b), (c) and (d) give lower weigh for both P and B frames in most of the time achieving the same or even better accuracy with the fast ME strategy.



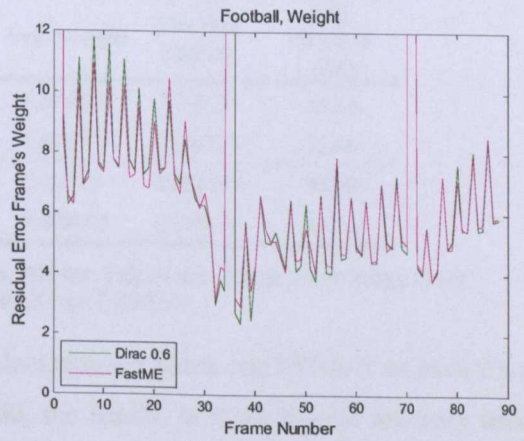
(a) Akiyo, Truncated Between Frame Num. 1 to 50



(b) Foreman, Truncated Between Frame Num. 150 to 200



(c) Bus, Truncated Between Frame Num. 50 to 100



(d) Football

Fig. 5-22 Comparison of Weights for Dirac 0.6 and the Fast ME, in CIF format

The above discussion has already proved the importance of P frame coding in the IBBP GOP structure. As discussed in section 5.5, according to the structure of GOP, the reference frames of P are very much further away compared with B, requiring wider search window or hierarchical way of searching in order to maintain the optimum accuracy.

Getting high level of ME accuracy in P frame coding is crucial since the MV from P frame is used as the temporal predicted MV in adjacent B frame's ME. Because of the application of non-

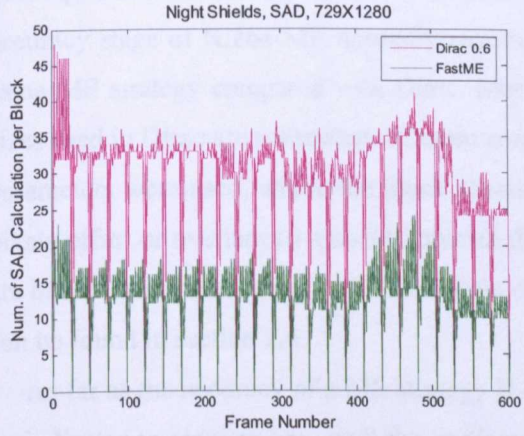
hierarchical way, ME in B frame coding have to rely mostly on the temporal predicted MV. A good reconstructed quality in P frame coding also helps in reducing the weight of the residual error of B frame. In general, higher accuracy of ME in P frame coding yields better picture quality which in turn gives lower B frame weight in motion compensation process. Specifically, the weight of the residual error frame either P or B frame depends upon the motion estimation accuracy and the reconstruction quality of its reference frame. It is because motion estimation is performed on the original uncompressed frames and the motion compensation process is carried out over the reconstructed reference frame.

Table 5-2 shows the motion estimation results from both Dirac 0.6 and the fast ME strategies for two types of HD formats. While the value of PSNR, compression efficiency (in terms of generated file size) and accuracy of ME (in terms of residual error frame's weight) are approximately the same, the fast ME strategy requires very much lower number of SAD calculation per block compared with existing strategy in Driac 0.6. Similar to the results in CIF format shown in Table 5-1, again, there is at least two fold reductions in the number of SAD calculation per block for both of the HD formats. The results in Table 5-2 also show that the reduction in SAD calculation is higher in the relatively static sequence (Night Shields) while more dynamic motion sequence (Pedestrian Area) gives approximately two folds reduction. Being HD data encoding, very small reduction in average weight reflects huge difference in generated compressed file size (around 100 Kbytes in both sequence), which can easily be seen in Table 5-2.

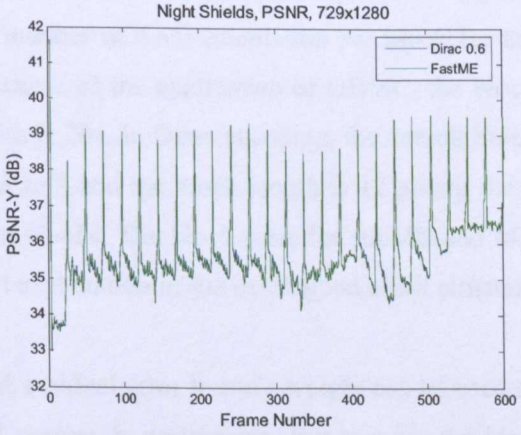
Sequence	ME Strategy	Avg. SAD/Block	Avg. Weight	File Size (bytes)	Avg. PSNR-Y (dB)
Night Shields 729×1280	Dirac 0.6	29.80	5.874248	7473176	35.66
	Fast ME	12.81	5.872379	7346537	35.66
Pedestrian Area 1080×1920	Dirac 0.6	31.09	10.2165	7424520	38.09
	Fast ME	16.98	9.978009	7334674	38.16

Table 5-2 The Comparison of ME Results for HD Formats, All the Values are Taken as Average Over Entire Encoding Sequence Except File Size

Fig. 5-23 and Fig. 5-24 show the number of SAD calculation per block and PSNR-Y of each frame for both HD test sequences shown in Table 5-2. Again, the results in these figures are very much similar to the CIF results shown in Fig. 5-18 to Fig. 5-21. From Fig. 5-23 (a) and Fig. 5-24 (a), it can also be seen clearly that the difference in the number of SAD calculation pre block between P and B frames in the fast ME strategy is much higher and changing more dynamically than existing ME strategy of Dirac 0.6. In these figures, the position of *I* frame can be seen clearly since the number of SAD calculation per block for these frames is zero. But, PSNR curves show less noticeable difference between the two strategies and it is almost identical with static sequence (Night Shield).

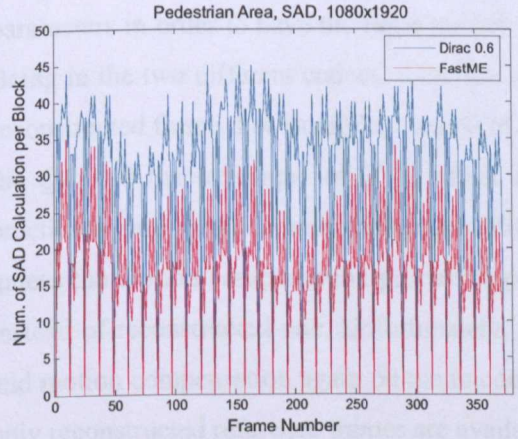


(a) Comparison of Number of SAD Calculations per Block

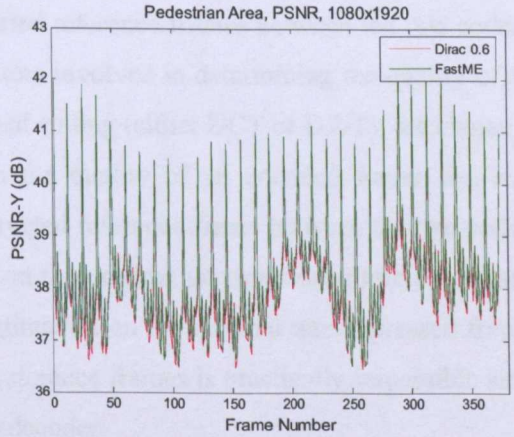


(b) Comparison of PSNR-Y

Fig. 5-23 Comparison of ME results for Dirac 0.6 and the FastME, Night Shields, HD 729×1280



(a) Comparison of Number of SAD Calculations per Block



(b) Comparison of PSNR-Y

Fig. 5-24 Comparison of ME results for Dirac 0.6 and the Fast ME, Pedestrian Area, HD 1080×1920

5.8.2 Semi-Hierarchical Fast Motion Estimation (H.264)

The semi-hierarchical fast ME strategy presented in this chapter is implemented in H.264 JM 11 reference software [22] in order to evaluate its performance against with the existing ME methods adopted in JM11. It is because, not like in rate control algorithm, there is no direct method of comparing the two strategies using the two codecs in a fair way. Available parameters which can be used to compare the two strategies are the number of SAD calculation per block, residual error frame's weight, generated number of bits or file size and PSNR.

Unfortunately, the number of SAD calculation per block between the two strategies using two codecs cannot be compared directly because of the different approach in pixel accuracy ME of the two codecs. As mentioned in section 5.6.5 H.264's pixel accuracy motion estimation involves multiple block sizes starting hierarchically from mode 1 to 7 or from biggest (16×16) to smallest (4×4) block size while Dirac's pixel accuracy ME searches only the block level which is smallest and

goes up to subMB and MB levels in the mode decision stage with subpixel accuracy MVs. So, pixel accuracy stage of H.264 ME normally requires higher number of SAD calculation per block for the same ME strategy compared with Dirac. Moreover, because of the application of OBMC, the block sizes used in Dirac are somewhat different compared with H.264. In Dirac encoding, the default block parameters were used, where the block separation is set to 8 and the block length is 12 giving the 4 pixels offset or overlapped amount between the adjacent blocks. The block sizes for subMB and MB are based upon these values and are multiple of 2. Detail explanation of the overlapped block structure can be found in section 2.4.

As far as the accuracy of a ME strategy is concerned, residual error frame's weight can be used as an indicator to monitor how well the implemented ME strategy is performing. But it is not the ideal indicator in representing the MV's accuracy because residual error frame's weight also depends upon the reconstructed quality of the reference frame as discussed in section 5.8.1. However, there is a way to represent the accuracy of the MV by residual error frame's weight only by adjusting the encoding parameters in order to have the same quality reconstructed reference frames between the two codecs. Being in the two different codecs, there are several factors involved in determining the quality of the reconstructed frame. For example, method of transformed coding (either DCT or DWT), the choice of the *QP* and the technique of VLC affect largely on the quality of an encoded frame. So, it is practically impossible to have the same quality reconstructed reference frame between the two codecs unless the motion compensation process is performed on the original uncompressed reference frame instead of reconstructed one. Unfortunately, motion estimation on the original uncompressed frames and motion compensation again on the uncompressed reference frames is practically impossible since only reconstructed reference frames are available at the decoder.

Parameter Description	Set Value
Profile	Main
Intra Period	12
Number of B frames	2
Inter Search Block Sizes	16×16, 8×8, 4×4
Search Range	16
Number of References	2
BList0References	0 (default) sets number to be equal to NumberReferenceFrames
BList1References	1
MV Precision	1/4 Pixel
Rate Control	Disabled
Use of Fast ME	0= Spiral, 1=UMHexagonS, 2=Simplified UMHexagonS, 3=EPZS, 4= Fast ME

Table 5-3 H.264 Configuration File Parameters

Another two parameters which are the generated number of bits and PSNR determine the compression efficiency and objective quality of a codec and they are useful only in a case where the performances of the two encoders are to be determined. So, there is no other way but implementing

the exactly the same strategy in the reference software of H.264 in order to evaluate its performance against with the existing adopted ME strategies of H.264. Table 5-3 shows the configuration file parameters of H.264 JM11 [22] used in encoding of the test sequences in this section.

The parameters in Table 5-3 are carefully chosen in order to keep the encoding environment of the two codecs as similar as possible even though there is no direct comparison on the results of the two codecs. Intra period is set to 12 and the number of B frames is 2 forming 36 GOP structure with main profile. Inter search block sizes are limited to only three levels removing the rectangular block structure in order to keep the block shape and their levels as close as that of Dirac. As in Dirac encoding, the number of reference frames is set to 2 with quarter pixel accuracy MV. But there are two lists of reference frame where each list can have up to N number of reference frames. N is the number of reference frames to be used and it was set to 2. List0References is set to zero (default value) which means overriding the List 0 reference is disable and so the number of reference frames in list 0 is equal to the number of reference frames which is 2. List1References is set to 1 since 1 reference is usually recommended for normal GOP structures. So, even though the number of reference was set to 2, the total number of reference frames for B is 3; i.e. 2 from list 0 and 1 from list 1 while the number of reference for P frame is still 2. Rate control is disabled and the required ME strategy is chosen from among five possible strategies including the fast ME.

Table 5-4 show results of five different ME strategies where four of them are existing adopted ME strategies of H.264 and the last one is the fast ME using semi-hierarchical approach. Unlike Dirac, the number of SAD calculation per block in H.264 is the total value which is averaged over the result of three block levels and can be calculated as:

$$\frac{\left(\left(\sum_{i=1}^{I} SAD16_i\right)\times 16^2\right)+\left(\left(\sum_{j=1}^{J} SAD8_j\right)\times 8^2\right)+\left(\left(\sum_{k=1}^{K} SAD4_k\right)\times 4^2\right)}{4^2\times K} \tag{5.9}$$

where, SAD16, SAD8 and SAD4 are the number of SAD calculations for a particular block i, j and k of block sizes 16×16, 8×8 and 4×4, respectively. I, J and K are the total number of blocks for the corresponding block sizes in a frame. So the result in equation (5.9) is actually the average number of SAD calculation per level 7 block (i.e. 4×4) over a frame. The result, average SAD calculation per block in Table 5-4, column 3 is the average value of equation (5.9) over the entire sequence including I frame.

From Table 5-4, it is interesting to find that EPZS offers the lowest number of SAD calculation per block from among the four ME strategies already adopted in H.264 while all the other data including residual error frame's weight, generated number of bits and PSNR are approximately remain the same. As expected, Spiral search gives flat number of SAD calculation regardless of the content of the encoding sequence. It is important to note that the average number of SAD calculation per block for all test sequence in spiral search should be the same in Table 5-4 if the test sequence lengths are

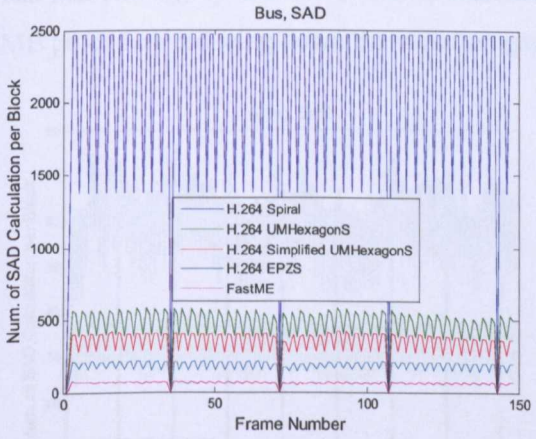
equal. Apart from complexity issue, spiral search gives the best performance in terms of error weight, file size and PSNR from among the four adopted ME strategies. But in overall, EPZS is the best followed by Simplified UMHexagon, UMHexagon and Spiral search. Being the best from among adopted ME strategies, only EPSZ will be used to compare with the fast ME strategy in the later part of the discussions.

Sequence	ME Strategy	Avg. SAD/Block	Avg. Weight	File Size (bytes)	Avg. PSNR-Y (dB)
Akiyo	H.264 Spiral	2058.977	4.556877	126388	39.86989
	H.264 UMHexagonS	142.1074	4.561025	125955	39.86047
	H.264 Simplified UMHexagonS	64.20805	4.567424	125842	39.83498
	H.264 EPZS	59.35906	4.562085	126863	39.86134
	H.264 Fast ME	32.58389	4.560703	126478	39.86305
Foreman	H.264 Spiral	2058.977	8.048367	559580	36.02452
	H.264 UMHexagonS	358.4698	8.145874	568721	35.99397
	H.264 Simplified UMHexagonS	242.3154	8.2119	568605	35.96817
	H.264 EPZS	147.4899	8.136794	569589	36.00326
	H.264 Fast ME	58.30872	7.904992	563270	36.01913
Bus	H.264 Spiral	2050.588	7.647301	725027	34.06482
	H.264 UMHexagonS	472.7297	8.016668	751377	33.99407
	H.264 Simplified UMHexagonS	342.6689	8.135844	755646	33.96008
	H.264 EPZS	182.5473	7.880499	735894	34.03321
	H.264 Fast ME	60.5	7.57017	693342	34.10883
Football	H.264 Spiral	2045.489	8.854028	392249	36.34858
	H.264 UMHexagonS	448.0909	9.257794	401030	36.31117
	H.264 Simplified UMHexagonS	322.8977	9.448333	397754	36.29262
	H.264 EPZS	200.0682	9.103859	395099	36.31627
	H.264 Fast ME	84	8.541634	389285	36.35373

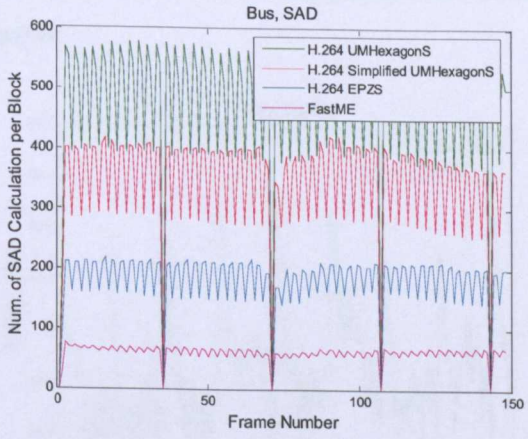
Table 5-4 The Comparison of ME Strategies in H.264 with the Fast ME using CIF Format, All the Values are Taken as Average Over Entire Encoding Sequence Except File Size

As shown in Table 5-4, the Fast ME strategy is even better than EPZS especially in the reduction of computational load. The complexity reduction is up to three folds for Bus sequence and for all test sequences, it offers at least two folds reduction compared with EPZS. But, in terms of compression efficiency, generated number of bits or file size is even smaller than that of EPZS in all test sequences while maintaining the same PSNR level. From Table 5-4, it can be seen clearly that this compression efficiency actually comes from the smaller residual error frame's weight which is mainly because of having better ME accuracy of the fast ME strategy compared with EPZS. Moreover, the performance of the fast ME strategy in terms of error weight, file size and PSNR is even better than spiral search in Bus and Football sequences. It is because spiral search is limited in the fixed size window resulting less accurate MVs for the encoding sequences which have faster motion or larger displacement

objects (e.g. Bus and Football) even though it is based upon FS. For this reason, apart from complexity issue, spiral search gives the best performance from among all listed strategies in Table 5-4 when the encoding test sequence is relatively static (e.g. Akiyo and Foreman).



(a) Comparison of Number of SAD Calculations per Block in Each Frame, Five Different Strategies



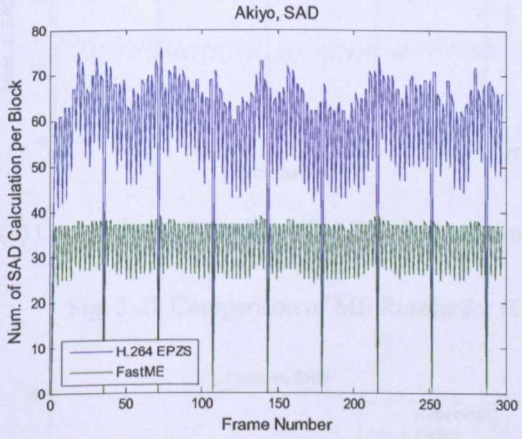
(b) Comparison of Number of SAD Calculations per Block in Each Frame, Without Spiral

Fig. 5-25 Comparison of Number of SAD Calculation per Block in Each Frame for Five Different ME Strategies Including the Fast ME Using H.264, Bus Sequence in CIF Format

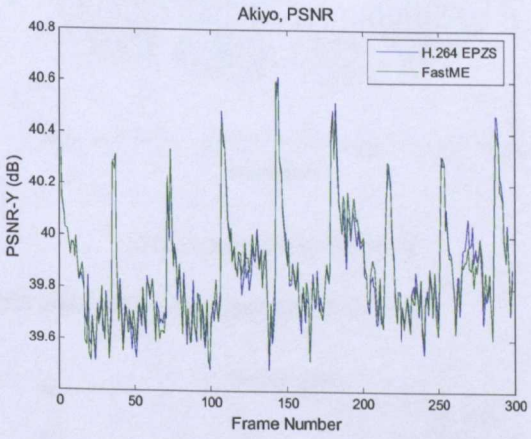
Fig. 5-25 (a) shows the comparison of number of SAD calculation per block in each frame for five different ME strategies including the fast ME using H.264 encoder with Bus sequence in CIF format. It shows the step by step improvement of the adopted ME strategies together with the fast ME in reducing the complexity of the pixel accuracy search. Even though Bus sequence is used here, all other sequences show more or less similar trend which can be seen clearly in Table 5-4 as the average value. As discussed, spiral gives constant number of SAD calculation for P and B frames and being based upon FS, it is the highest among the different strategies listed in the figure. There is a huge gap between spiral and UMHexagonS followed very closely by Simplified UMHexagonS, EPZS and the the fast ME. Spiral Data is removed in Fig. 5-25 (b) in order to be able to monitor the behavior of each strategy more closely. From Fig. 5-25 (a) and (b), it is interesting to see that the difference between the number of SAD calculation between P and B frame is much higher giving wider variation in spiral search and this variation becomes gradually narrow down as the overall complexity becomes lower in the other strategies. The fast ME strategy which gives the lowest complexity has the narrowest variation as shown in Fig. 5-25 (b).

It is because H.264's B frame ME involves three reference frames (i.e. 2 from list 0 and 1 from list 1) while P frame ME has only two, resulting the higher number of SAD calculation per block in B frame ME. The variation is widest in spiral because of the application of fixed window size ME. But in adaptive based ME (i.e. UMHexagonS, Simplified UMHexagonS and EPZS), the variation becomes gradually narrow down as the algorithm inside each ME strategy try to find the global minimum as accurately as possible. As discussed in section 5.5, finding the global minimum in P frame normally requires wider window size because of the structure of GOP requiring more number

of SAD calculations per block while B frame ME requires relatively narrower window since the location of their reference frames are closer to the coding frame. But in the fast ME strategy, because of the application of un-equal level of ME (i.e. semi-hierarchical approach), the algorithm inside the fast ME strategy spends more time in searching the best match in P frame by using fully hierarchical ME procedure reducing down the variation gap even further.

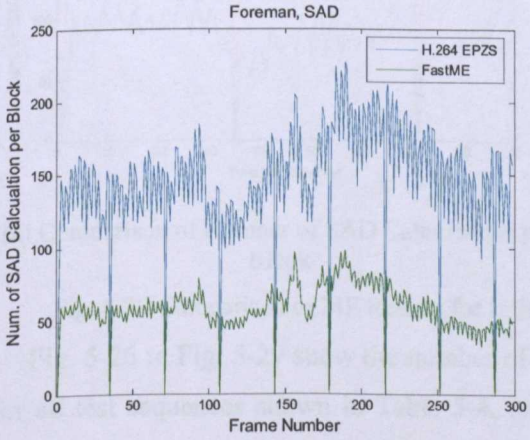


(a) Comparison of Number of SAD Calculations per Block

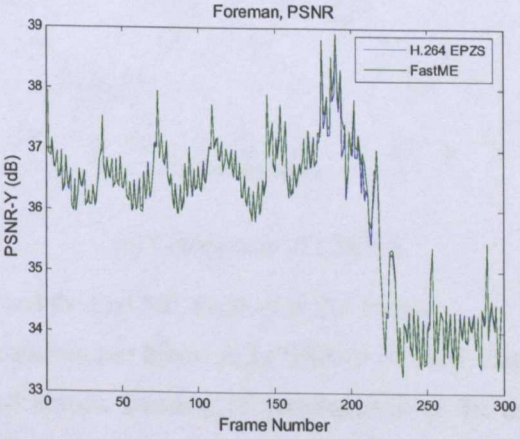


(b) Comparison of PSNR-Y

Fig. 5-26 Comparison of ME Results for H.264's EPZS and the Fast ME, Akiyo in CIF Format

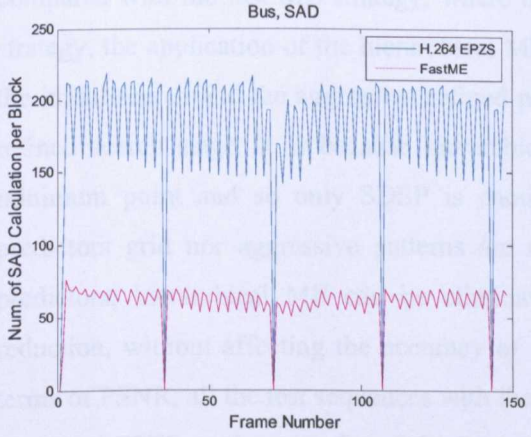


(a) Comparison of Number of SAD Calculations per Block

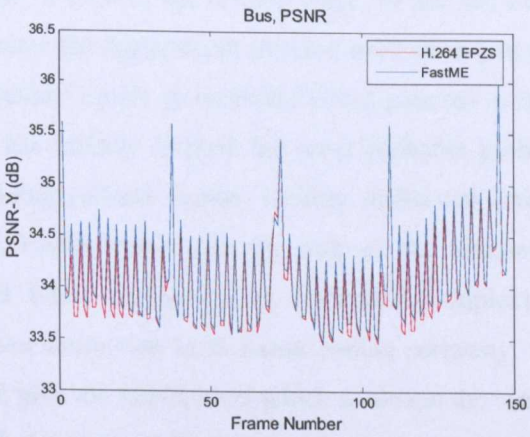


(b) Comparison of PSNR-Y

Fig. 5-27 Comparison of ME Results for H.264's EPZS and the Fast ME, Foreman in CIF Format

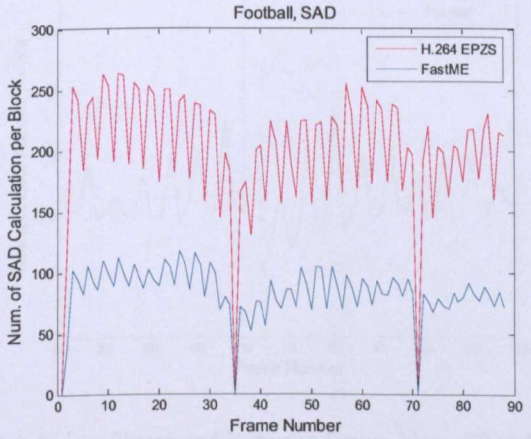


(a) Comparison of Number ofSAD Calculations per Block

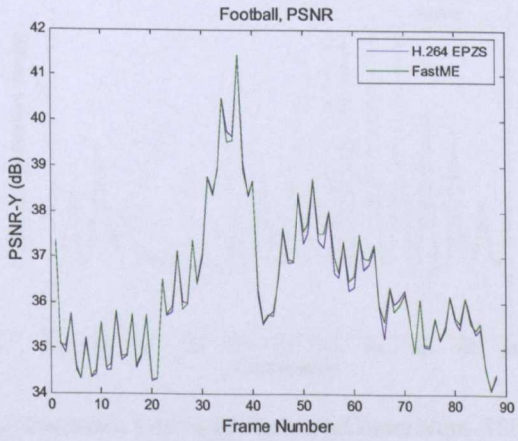


(b) Comparison of PSNR-Y

Fig. 5-28 Comparison of ME Results for H.264's EPZS and the Fast ME, Bus in CIF Format



(a) Comparison of Number of SAD Calculations per Block

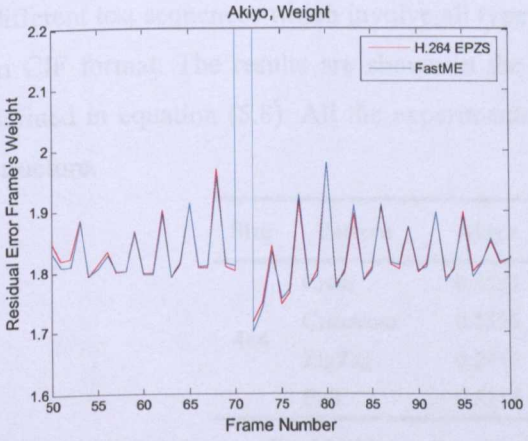


(b) Comparison of PSNR-Y

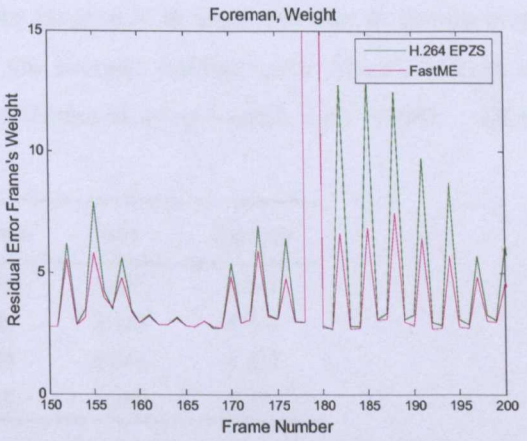
Fig. 5-29 Comparison of ME Results for H.264's EPZS and the Fast ME, Football in CIF Format

Fig. 5-26 to Fig. 5-29 show the number of SAD calculation per block and PSNR-Y of each frame for all test sequences shown in Table 5-4. As discussed before, because of consequence of the un-equal number of reference frames between P and B, unlike Dirac ME, the number of SAD calculation per block in B frames is higher than that of P in EPZS of H.264. However, because of the application of semi-hierarchical motion estimation in the fast ME, the required number of SAD calculation difference between P and B becomes narrower compared with EPZS, which can be seen clearly in Fig. 5-26 (a) to Fig. 5-29 (a). Reduction in the number of SAD calculation actually comes from the application of adaptive search algorithm and semi-hierarchical motion estimation. Even though EPZS uses similar adaptive search algorithm as in the fast ME strategy, addition of fixed predictors grid as shown in Fig. 5-14 causes the algorithm to search more prediction points. In the refined stage as well, EPZS uses square shape and more aggressive pattern called extended EPSZ pattern as shown in Fig. 5-15 in addition to the SDSP giving dramatic increments in the number of SAD calculation per block

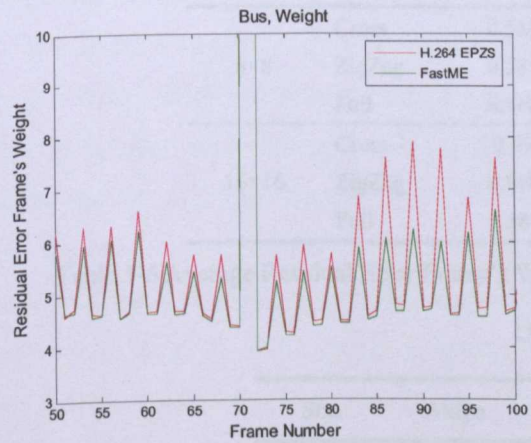
compared with the fast ME strategy, where only SDSP is used in the refined stage. In the fast ME strategy, the application of the hierarchical ME eliminates the requirement of fixed predictors grid in the initial search and the aggressive refined patterns (either square or extended EPSZ pattern) in the refined search stage. It is because hierarchical ME has already located the most probable global minimum point and so only SDSP is enough in doing refined search. Neither additional fixed predictors grid nor aggressive patterns are required. Furthermore, with the help of the temporal predictors, hierarchical ME can be eliminated in B frame coding giving additional complexity reduction, without affecting the accuracy of the motion estimation in B frame coding seriously. In terms of PSNR, all the test sequences with the fast ME give the PSNR level which is almost the same as that of EPZS as shown in Fig. 5-26 (b) to Fig. 5-29 (b) while giving at least two fold reduction in computational load.



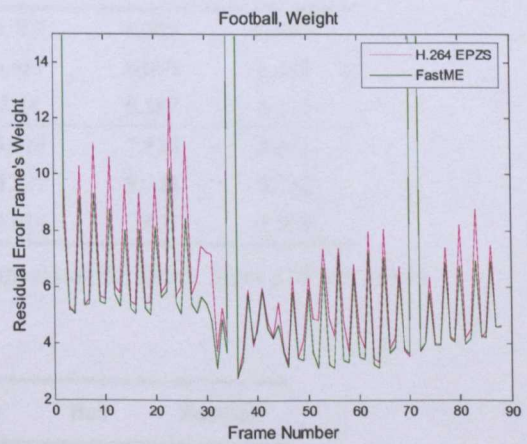
(a) Akiyo, Truncated Between Frame Num. 50 to 100



(b) Foreman, Truncated Between Frame Num. 150 to 200



(c) Bus, Truncated Between Frame Num. 50 to 100



(d) Football

Fig. 5-30 Comparison of Weights for H.264'S EPZS and the Fast ME, in CIF format

Fig. 5-30 shows the comparison of residual error frame's weight between H.264's EPZS and the fast ME for all test sequences in CIF format. As in Fig. 5-22, most of the long sequences are truncated (in Fig. 5-30 (a), (b) and (c)) in order to show the weight comparison between the two ME strategies

in more details. Because of the application of un-equal level of ME, P frame's weight with the fast ME strategy is lower than that of EPZS in most of the test sequences. Weight reduction in P frame can be seen more clearly in dynamic motion sequences as shown in Fig. 5-30 (b), (c) and (d). It is because for a static sequence like in Akiyo, it is not required to increase the search window size for P frame ME because of the static nature of sequence. For static sequence, the algorithm in any type of ME strategy can easily find the best match in the vicinity or exactly the same location of the current block resulting exactly the same or approximately the same level of accuracy for most of the ME strategies.

5.8.3 Partial Cost Function Calculation

In this section, the performance of the partial cost function calculation patterns presented in section 5.7 will be evaluated and their results will be discussed in details. In order to investigate their performance, the patterns shown in Fig. 5-16 were implemented using FS method and tested with different test sequences which involve all type of motions ranging from slow, medium to fast motions in CIF format. The results are shown in the form of the average residual error frame's weight as defined in equation (5.8). All the experiments were implemented using Matlab with IPPPP.... GOP structure.

Size	Pattern	Akiyo	Foreman	Bus	Football
4×4	Cross	0.5523	3.154	5.525	4.631
	Crosscom	0.5536	3.16	5.518	4.636
	ZigZag	0.5414	3.083	5.303	4.529
	Full	0.5114	2.815	4.988	4.371

Table 5-5 Average Residual Error Frame's Weight Comparison for Block Size 4×4, Using FS

Size	Pattern	Akiyo	Foreman	Bus	Football
8×8	Cross	0.5862	3.709	6.585	6.507
	ZigZag	0.5817	3.681	6.699	6.382
	Full	0.5604	3.44	6.167	6.116
16×16	Cross	0.598	4.214	7.834	8.423
	ZigZag	0.5988	4.227	8.176	8.732
	Full	0.5872	4.014	7.471	7.909

Table 5-6 Average Residual Error Frame's Weight Comparison for Block Sizes 8×8 and 16×16, Using FS

Size	Akiyo	Foreman	Bus	Football
4×4	0.03	0.268	0.315	0.158
8×8	0.0213	0.241	0.418	0.266
16×16	0.0108	0.2	0.363	0.514

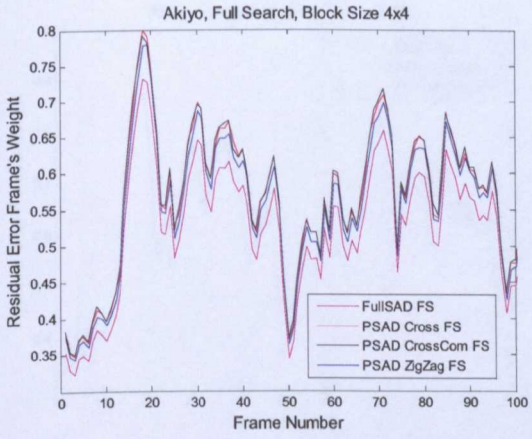
Table 5-7 Weight Deviation of the Partial SAD Calculation from Full SAD for All Block Sizes

Table 5-5 shows the average residual error frame's weight comparison of partial SAD calculation patterns with full SAD for block size 4×4. In most of the test sequences, cross pattern performs slightly better than cross complement but zigzag pattern gives the lowest weight compared with the other two. In the residual error frame's weight versus frame number curve, zigzag curve is just above the FS for 4×4 block size as shown in Fig. 5-31. Apparently, the results from Table 5-5 show the crucial requirement of the distribution of SAD calculation points over the block. The coverage area of cross complement pattern is only at the margin giving higher weight compared with cross in most of the test sequences. The performance comparison of cross and zigzag patterns is extended to the larger block sizes which are 8×8 and 16×16 as shown in Fig. 5-17 (a) to (d).

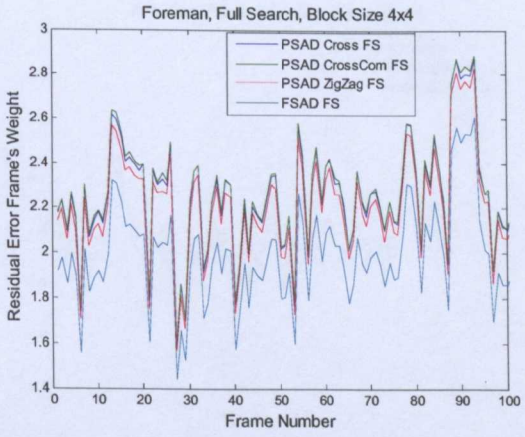
Table 5-6 shows the average residual error frame's weight comparison of partial SAD calculation patterns with full SAD for block size 8×8 and 16×16. As the block size becomes larger, the performance of zigzag becomes comparable with cross and in some case, even lower than cross, e.g. in bus sequence for 8×8 block size. But in block size 16×16, all the results are in the reverse condition and the cross pattern outperforms zigzag in all test sequences. It can also be seen in Fig. 5-32 (a) to (d) where partial SAD curve using cross pattern is just above full SAD for all test sequence. It is because in the zigzag pattern, it is increasingly difficult to locate the calculation points to cover the most area of the block as the block size increase, which results in lower performance compared with cross. But in cross, having specific and more concentrated points at the centre of the block, gives more chance to yield the optimum performance for all sizes of blocks.

Table 5-7 shows the deviation of best error weight from the full SAD calculation for all block sizes and for all test sequences. The best error weight is the minimum weight of a particular calculation pattern and can be either cross, cross complement or zigzag in 4×4 block size and cross or zigzag in other block sizes.

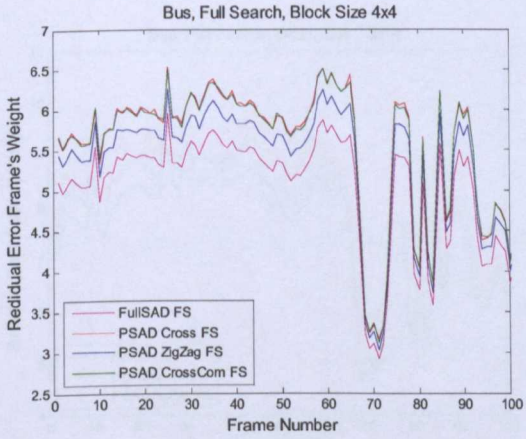
It is interesting to note that the deviation from the full SAD calculation result becomes smaller as the block size increases for Akiyo and Foreman test sequences. It is because as the block size reduces; it is increasingly difficult to get a unique SAD value even with the full SAD calculation because there is less number of pixels involved in the absolute difference summation. So, in the case of partial SAD calculation, it is more likely to miss interpret a minimum SAD point giving higher deviation in smaller block size especially for the relatively static motion sequences. It can be seen clearly on Fig. 5-31 (a), (b) and Fig. 5-32 (a), (b) where partial SAD calculation curves are very much closer to the full SAD curves as the block size larger. This effect is not much significant in complex and faster motion because bigger block size likely to have the higher number of complex motion objects giving more deviation error in the partial SAD calculation. So, generally speaking, accuracy of the partial SAD calculation is more sensitive to the smaller block size and it is especially obvious in static motion sequences.



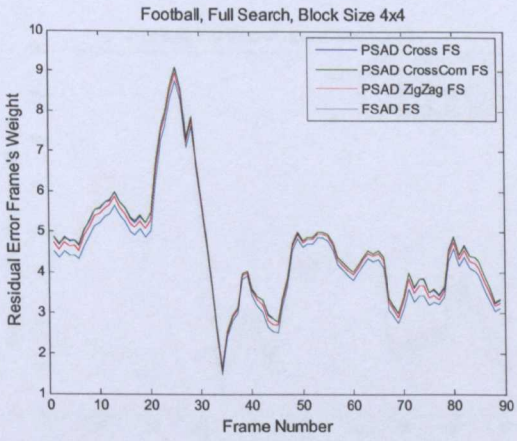
(a) Akiyo, Truncated Between Frame Num. 1 to 100



(b) Foreman, Truncated Between Frame Num. 1 to 100

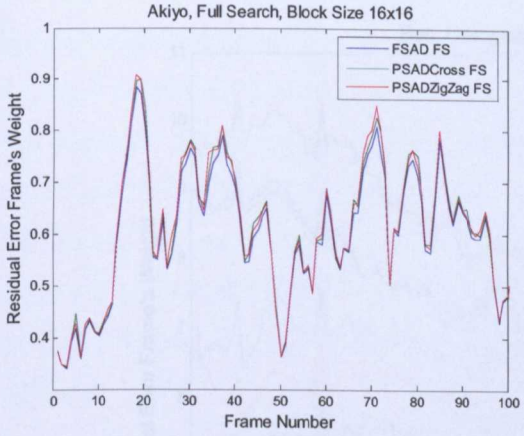


(c) Bus, Truncated Between Frame Num. 1 to 100

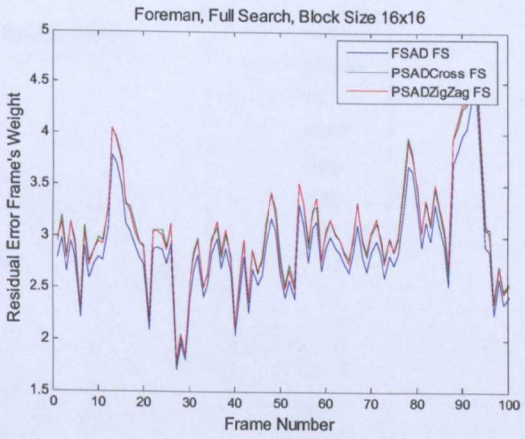


(d) Football

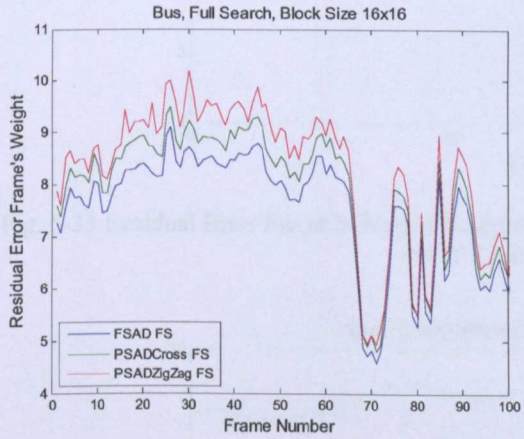
Fig. 5-31 Residual Error Frame's Weight Comparison for Different Partial SAD Calculation Patterns and Full SAD Using FS, CIF Format with Block Size 4×4



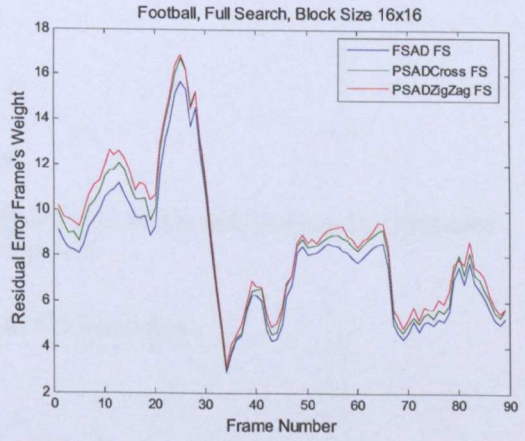
(a) Akiyo, Truncated Between Frame Num. 1 to 100



(b) Foreman, Truncated Between Frame Num. 1 to 100



(c) Bus, Truncated Between Frame Num. 1 to 100



(d) Football

Fig. 5-32 Residual Error Frame's Weight Comparison for Different Partial SAD Calculation Patterns and Full SAD Using FS, CIF Format with Block Size 16×16

In this research, the cross pattern is chosen to find the performance of partial SAD calculation with some of the fast BMAs mentioned in section 5.4 because of its simplicity and giving optimum performance for all block sizes. But, the modification of zigzag pattern by locating the points systematically in order to maximize the coverage over the block area can still achieve the better performance than cross for all block sizes with the expense of simplicity.

In order to evaluate the performance of the chosen partial cost function calculation pattern over fast BMAs, it is required to select some of the best BMAs mentioned in section 5.4. The selection is based upon their performance in terms of the required number of cost calculation per block and the accuracy which is measured by comparing the residual error frame's weight.

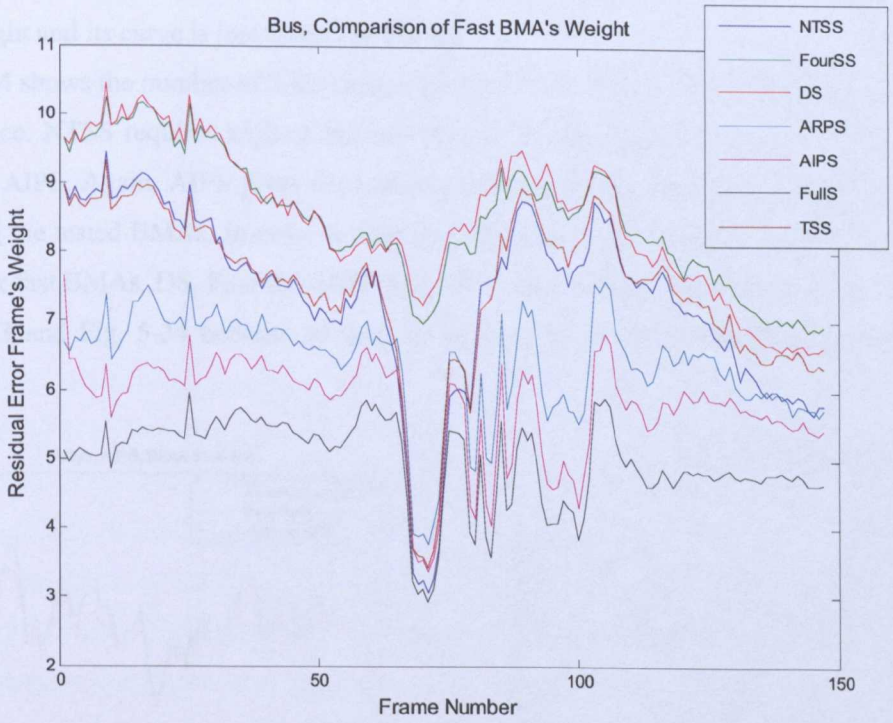


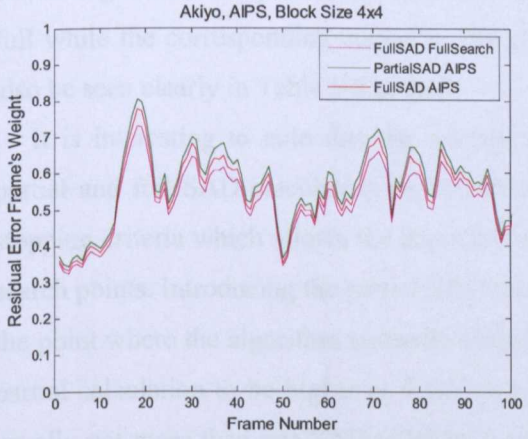
Fig. 5-33 Residual Error Frame's Weight Comparison of Different Fast BMAs and FS using Bus Sequence in CIF Format, Block Size 4×4

Fig. 5-34 Comparison of the Number of SAD Computation per Block of Different Fast BMAs using Bus Sequence in CIF Format, Block Size 4×4

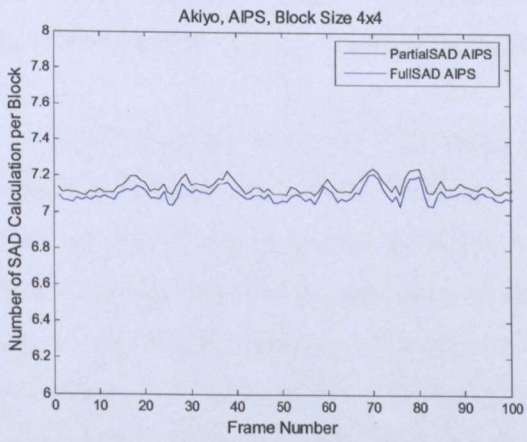
Fig. 5-33 shows the residual error frame's weight comparison of six different fast BMAs and FS using bus sequence in CIF format. The block size used was 4×4. The figure shows highest weights for

DS and FourSS followed by TSS, NTSS, ARPS and AIPS. It is interesting to note that AIPS gives lowest weight and its curve is just above the FS one.

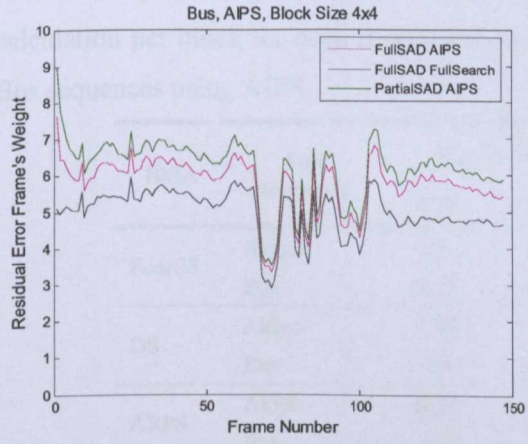
Fig. 5-34 shows the number of SAD computation of all the BMAs shown in Fig. 5-33 for the same test sequence. NTSS requires highest number of SAD computation followed by TSS, FourSS, DS, ARPS and AIPS. Again, AIPS gives the lowest number of SAD computation and considered as the best among the tested BMAs. In order to find the performance of the chosen partial SAD calculation pattern over fast BMAs, DS, FourSS, ARPS and AIPS were chosen from among the six BMAs shown in Fig. 5-33 and Fig. 5-34 because of their lower number of SAD computation and the optimum weight.



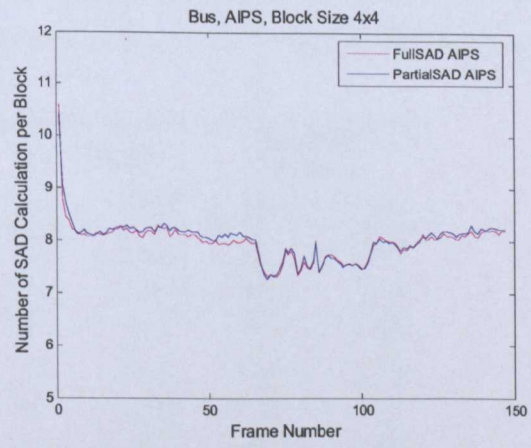
(a) Akiyo, Comparison of Weight, Truncated Between Frame Num. 1 to 100



(b) Akiyo, Comparison of Number of SAD Calculation, Truncated Between Frame Num. 1 to 100



(c) Bus, Comparison of Weight



(d) Bus, Comparison of Number of SAD Calculation

Fig. 5-35 Performance Comparison of Partial and Full SAD Calculation using AIPS

Chosen partial cost calculation pattern (cross) is again implemented in four selected fast BMAs in order to investigate the effects of partial SAD calculation over the performance of BMA. It is implemented and tested only for the 4×4 block size since the objective is to find the accuracy and complexity of the BMA using partial SAD calculation. The accuracy and complexity of a BMA can

be represented in terms of residual error frame's weight and average number of operation (i.e. a pair of subtraction and addition) per block, respectively.

Fig. 5-35 shows the performance comparison in terms of the error weight and the number of SAD calculation for partial and full SAD calculation using AIPS. The result of full SAD calculation using FS is also mentioned in both sequences in order to have the better comparison idea relative to FS. As expected, the error weight of the partial SAD calculation is slightly higher than full in both sequences; Akiyo and Bus as shown in Fig. 5-35 (a) and (c). The amount of weight increment depends upon speed of the motion and complexity of the test sequence. The partial SAD calculation of the complex motion sequence tends to give higher weight increment than the less complex one. As the results, the error weight curve for partial SAD calculation in Fig. 5-35 (c) for Bus sequence is slightly higher than full while the corresponding curves in Fig. 5-35 (a) for Akiyo sequence remain comparable. It can also be seen clearly in Table 5-8 as well.

It is interesting to note that the average numbers of SAD calculation points for each block in partial and full SAD calculation are not exactly the same. It is because most of the fast BMA use stopping criteria which allows the algorithm to stop in the half way in order to reduce the number of search points. Introducing the partial SAD calculation idea causes the algorithm to stop before or after the point where the algorithm normally would stop. It causes the average number of SAD points with partial calculation to be higher or lower than the full calculation. But, this amount is negligible and usually not more than one SAD point in average. However, there are reduced numbers of pixel (half in 4×4 block size) in each SAD calculation with the presented patterns giving lower computational load to the processing unit on the hardware. The increment or decrement of the number of SAD calculation per block for each frames can be seen clearly in the Fig. 5-35 (b) and (d) for Akiyo and Bus sequences using AIPS.

BMA	Test Sequence	Avg. Weight		Avg. SAD/Bock		Avg. Num. of Operation	
		Full	Partial	Full	Partial	Full	Partial
FourSS	Akiyo	0.54	0.57	16.91	16.99	270.56	135.92
	Bus	8.51	9.32	21.55	21.45	344.8	171.6
DS	Akiyo	0.53	0.56	13.1	13.21	209.6	105.68
	Bus	8.5	9.48	19.93	19.47	318.88	155.76
ARPS	Akiyo	0.53	0.55	5.38	5.47	86.08	43.76
	Bus	6.4	7.34	10.51	10.49	168.16	83.92
AIPS	Akiyo	0.53	0.55	7.09	7.14	113.44	57.12
	Bus	5.78	6.2	7.98	8.03	127.68	64.24

Table 5-8 Performance Comparison of Partial and Full SAD Calculation for Different Types of BMAs, Block Size 4×4

Table 5-8 shows the average error weight and SAD calculation for the whole sequence using different types of fast BMAs for Akiyo and Bus. The average SAD calculation per block is the number of SAD computation per block defined in equation (5.7) which is averaged over entire

encoding sequence. The number of operation is the number of subtraction and addition pairs in a SAD calculation and hence the average number of operation in the last column of Table 5-8 is the multiplication of average SAD calculation per block with 16 for full SAD and 8 for partial SAD for 4×4 block size.

The average weight using partial SAD calculation is slightly higher than full calculation in both sequences for all types of algorithms. Even though the weight increment in partial SAD calculation compared with full in Bus sequence is more than that of Akiyo, it is not more than one unit in all search algorithms. As discussed before, the average number of SAD calculation points for the whole sequence using partial calculation is slightly higher or lower than that of full calculation in all cases. But, the increment or decrement is not more than one SAD point in average and the numbers of SAD calculation points are comparable in all cases using different sequences and different algorithms. Nevertheless, the reduced numbers of pixel (half in 4×4 block size) in every SAD calculation with the presented pattern, is giving lower computational load to the processing unit on the hardware. The computational load reduction can be seen clearly in the last column of Table 5-8 in terms of the average number of operation per SAD calculation.

As an overall, introducing the idea of partial SAD calculation reduces the computational load significantly and the load reduction factor is 2, 4 or 8 times of full SAD calculation for block sizes 4×4, 8×8 and 16×16, respectively. But, in order to achieve this, there is a little trade off in the form of motion estimation accuracy which is slightly less than full SAD calculation result and usually it is negligible in static motion sequences. Even though there are reduced number of the sum of the absolute different pixels in each block during the partial SAD calculation, the average number of SAD calculation per block for the whole sequence are remain almost the same as shown in Table 5-8.

5.9 Chapter Summary

In this chapter, a fast ME strategy which is the combination of modified adaptive search plus semi-hierarchical way of motion estimation is presented. The strategy gives approximately the same motion estimation accuracy, compression efficiency and PSNR performance compared with existing ME strategy of Dirac 0.6. But there is a significant improvement as far as the speed of the algorithm is concerned. There is a huge saving, at least two folds in average number of SAD calculation per block for all test sequences in CIF format and up to three folds in relatively static sequences. Reduction in the number of SAD calculation is much more significant in static sequence than dynamic one. Similar to the result in CIF format, there is also at least two folds reduction in the number of SAD calculation per block for both of the HD formats. The results with HD sequence encoding also confirm that the reduction in SAD calculation is higher in the relatively static sequence (Night Shields) while more dynamic motion sequence (Pedestrian Area) gives approximately two folds reduction.

Exactly the same ME strategy (i.e. fast ME) was implemented in H.264 JM 11 reference software in order to evaluate its performance against with the existing ME methods adopted in JM11. It was found that in overall, EPZS is the best followed by Simplified UMHexagon, UMHexagon and Spiral search. But, it is interesting to find that the fast ME strategy is even better than EPZS especially in the reduction of computational load. The complexity reduction is up to three folds for Bus sequence and for all test sequences, it offers at least two folds reduction compared with EPZS.

As far as the compression efficiency is concerned, it was found that the generated number of bits or file size is even smaller than that of EPZS in all test sequences while maintaining the same PSNR level. The achieved compression efficiency actually comes from the smaller residual error frame's weight which is mainly because of having better ME accuracy of the fast ME strategy compared with EPZS. Moreover, it was interesting to find that the performance of the fast ME strategy in terms of error weight, file size and PSNR is even better than spiral search in dynamic motion sequences (e.g. Bus and Football).

Finally, performance of the partial cost function calculation patterns is evaluated using FS method with different block sizes and test sequences. It was interesting to find that the deviation from the full SAD calculation result becomes smaller as the block size increases for relatively static sequences (e.g. Akiyo and Foreman). So, generally speaking, accuracy of the partial SAD calculation is more sensitive to the smaller block size and it is especially obvious in static motion sequences.

The cross pattern was chosen to find the performance of partial SAD calculation with some of the fast BMAs in the later stages because of its simplicity and giving optimum performance for all block sizes. In order to implement the chosen partial SAD calculation pattern on the fast BMAs, DS, FourSS, ARPS and AIPS were chosen from among the others BMAs. The selection is based upon their complexity and the accuracy. As expected, it was found that the average weight using partial SAD calculation is slightly higher than full calculation in both tested sequences for all types of algorithms. It was also found that the amount of weight increment depends upon speed of the motion

and complexity of the test sequence. The partial SAD calculation of the complex motion sequence tends to give higher weight increment than the less complex one.

Because of the application of stopping criteria in most of the BMAs, it was interesting to find that the average numbers of SAD calculation points for each block in partial and full SAD calculation are not exactly the same but, the increment or decrement is not more than one SAD point in average. Mostly, the numbers of SAD calculation points are comparable in all cases using different sequences and different algorithms. But, the reduced numbers of pixel (half in 4×4 block size) in every SAD calculation of the proposed method is giving lower computational load to the processing unit on the hardware.

Finally, it is obvious to see that the partial cost function calculation method is very promising idea in achieving the fast motion estimation by further reducing the computational load in cost function calculation. Moreover, it can be incorporated in any type of search strategies and in any type of fast motion estimation algorithms.

Chapter 6

6 Conclusion and Further Recommendation

In this part of the chapter, all the research works presented in the previous chapters will be summarized and concluded along with their overall achievement to the aim and objective of the research. After that, all the possible modification which could improve the performance of the presented research methodologies will be discussed in details as the future work.

6.1 Overall Achievement

6.1.1 Error-Resilient Coding Scheme

In this part of the research, a combined source and channel coding scheme which provides the error-resilient transmission of the compressed video bitstream of Dirac video encoder over the packet erasure wired network, is presented. As discussed before, in the current alpha releases of Dirac, the encoder has only been optimized for storage purposes and still there is no error-resilient encoding mechanism in order to be able to use in real time transmission. The main objective of this research is to propose a simple, low complexity and patent free error-resilient coding scheme since one of the design criteria of the Driac video codec is to be simple and the developers do not want to include any patented algorithm in their codec architecture.

According to the experimental results, for the same type of channel coding which is RCPC, the maximum source coding gains are found to be 10 dB, 8 dB and 4 dB in 99P, 33P and 6-3P partitioned formats at 8 % packet loss rate. On the other hand, the channel coding gains for the rate 1/2, 1/3 and 1/4 using RCPC are around 4 dB, 17 dB and 20 dB, respectively at the 10% packet loss. From the results, it can be concluded that for the combined source and RCPC type of channel coding, encoder rates 1/3 and 1/4 are able to be used to protect the header plus MV or layer 1 for less than 6% packet loss and rate 1/2 is optimum for the protection of data or layer 2.

But the simulation results also highlighted that with the use of less powerful FEC e.g. RCPC, Dirac can only operate in a less congested network with the packet loss not more than 10%. Instead, using a more powerful FEC e.g. Turbo codes, Dirac could operate in a congested network up to 30% packet loss. In terms of the subjective quality, it was found that the application of the error concealment becomes easier with the presented scheme. Even the simple error concealment method, in which the corrupted coefficients are replaced with the average values of the coefficients of the previous and next partitions, is proved to be very effective in concealing the error since the errors become localized in one partition. The data loss can be estimated either from the adjacent partitions (spatially) or the adjacent frames (temporally) or using more sophisticated error concealment algorithms available in the literature.

Finally, it was found that because of the arrangement of wavelet transformed data, source coding (data partitioning) only in Dirac is not enough for the error-resilient transmission of wavelet compressed bitstream and hence the requirement of channel coding becomes mandatory in order to protect the sensitive part of the coefficients in the wavelet transform subbands.

From the description of the presented scheme and the results, it can be seen clearly that the coefficient partitioning process itself does not introduce much complexity to the encoder, and the use of low complex channel coding (i.e. RCPC) as a whole offer a simple and effective error-resilient scheme for Dirac video encoder. On the other hand, with the use of more complex channel coding (i.e. Turbo), the combined scheme can be a powerful mechanism to combat the harsh packet-erasure channel/network. So, the scheme is found to be not only simple and efficient but also offering flexible solution on the different level of network congestion, achieving the main objective of the research.

6.1.2 Rate Control Algorithm

As the second phase of the research, a rate control algorithm which is efficient, simple and easy to integrate to the Dirac encoder was presented in chapter 4. The algorithm is based upon the R-QF model since *QF* which is an integral parameter of Dirac encoder plays an important role in controlling the quality of the encoded video sequence or the number of bit generated. The current Dirac architecture is controlling constant quality rather than bitrate by using a single *QF* as quality indicator to maintain the desired quality. The algorithm presented in this research exploits this idea by considering *QF* as a varying parameter in order to achieve average bitrate which is constant over each GOP.

As mentioned in chapter 4, the main objective of this research is to implement a simple and efficient rate control algorithm for the Dirac video encoder in order to be able to be used in real-time video broadcasting together with the presented error-resilient coding scheme discussed in chapter 3.

According to the experimental results, it was found that the presented algorithm performs very well both in intra frame-only and inter frame coding mode. But, the algorithm works more accurately in intra frame-only coding giving the deviation error which is well within 1% of the target bitrate. For inter frame coding, the precision is around 1% of target bitrates.

In comparison with H.264, it was found that the maximum deviation error of the rate control technique [41] in JM11 reference software of H.264 is higher than that of the R-QF model based algorithm with Dirac in most of the time especially in the larger frame size (e.g. CIF and HD) and smaller frame size with higher target bit rate (e.g. QCIF with 128 Kbps). But in the measure of average deviation error, PSNR-Y and SSIM-Y, encoded video sequences with Dirac show lower performance especially with the lower target bitrate (32, 64 Kbps) in QCIF frame coding. But the average deviation error and SSIM performance becomes comparable with that of H.264 as the target bitrate increases (512, 1024 Kbps) in CIF and HD format video sequences while PSNR performance of Dirac is lower in all cases.

In general, it was found that the performance of the Dirac encoder with the presented rate control algorithm increases both with the video frame size and the target bitrate showing better bitrate regulation and comparable SSIM index with H.264 especially in CIF with higher target bitrate and HD encoding.

Being able to regulate the bitrate according to the target bitrate is crucial in real-time multimedia data streaming in preventing buffer overflow or underflow. Moreover, unlike rate control in H.264, the algorithm is based only on the GOP level and frame level rate control reducing the complexity dramatically since it does not require MB or basic unit level bit allocation. In addition to this, it is a complete one pass process and does not require iterating the calculation for finding the optimum *QF* value. The calculation of *QF* is based upon the simple mathematical equation and it does not even need to calculate the *QF* for each frame in inter frame coding mode. The algorithm is also capable of controlling the large range of bitrates from a few to several thousand Kbps and so it is practically applicable for all types of video frame sizes (QCIF to HD) supported from Dirac. It is also applicable to the different coding modes supported from Dirac, which are intra frame-only and inter frame coding. So, it can be concluded that the presented R-QF model based rate control algorithm achieves both accuracy and simplicity at the same time with the ability to support all type of coding modes supported from Dirac while capable of controlling the larger range of bitrates.

6.1.3 Motion Estimation Strategy

As for the final phase of the research, a fast ME strategy which is the combination of modified adaptive search plus semi-hierarchical way of motion estimation was presented in chapter 5. The same strategy is again implemented in H.264 to evaluate its performance against with the existing ME strategies currently adopted by H.264. Finally, the new idea of partial SAD operation in order to reduce the computational load in the cost function calculation was presented as the last part of the chapter. It can be used with any type of ME strategy and BMAs currently available in the literature.

The main objective of this research is to propose the fast and efficient motion estimation strategy since the current release of the Dirac encoder [1] employs fully hierarchical motion estimation for all types of inter frames consuming almost 80% of the total encoding time in motion estimation stage only.

According to the experimental results, the fast ME strategy gives approximately the same motion estimation accuracy, compression efficiency and PSNR performance compared with existing ME strategy of both Dirac 0.6 and H.264 JM 11. But there is a significant improvement as far as the speed of the algorithm is concerned. There is a huge saving, at least two folds in average number of SAD calculation per block for all test sequences and up to three folds in relatively static sequences. Reduction in the number of SAD calculation is much more significant in static sequence than dynamic one. The complexity of the fast ME strategy can be further reduced down by implementing the partial SAD calculation. The additional computational load reduction can be at least two folds using only

with 4×4 block size by just trading with a small portion of the motion estimation accuracy. The reduction in the computational load can be up to 8 times compared with full SAD calculation when using 16×16 block size calculation. It was not implemented in the fast ME algorithm and left for the future work since the main objective of the research is to reduce down the real number of SAD calculation. As an overall, the presented fast ME strategy achieves the main objective by giving at least two fold reduction in the complexity of ME and found that it is even much better than the best ME strategy currently employed in H.264 which is considered to be the state-of-the-art video encoder.

6.2 Future Recommendation

In this section of future recommendation, all possible techniques which could improve the performance of the presented methods mentioned in chapter 3, 4 and 5 will be discussed in detail. The ideas given here are for guideline only and hence further adjustments or modifications might be required in order to achieve the required performance.

6.2.1 Error-Resilient Scheme

6.2.1.1 Un-Equal Error Protection

In the presented scheme, according to the existing bitstream syntax, the bitstream of Dirac is divided into two layers; namely layer 1 and layer 2 for header plus MV and data. Currently, in the presented method, the same level of protection (i.e. protection using channel coding) is applied to the coefficients data of each subband. But, the wavelet transformed coefficient data from subband 13 to 10 (i.e. four lowest resolution subbands for four levels transform as shown in Fig. 6-1) are relatively quite important for the good reconstruction of the video frame. So, it would be interesting to find the performance of existing scheme by giving better protection to these subband coefficient data. The new idea can be implemented by introducing another layer, layer 3 with the different levels of protection using different encoding rate as shown in Fig. 6-2.

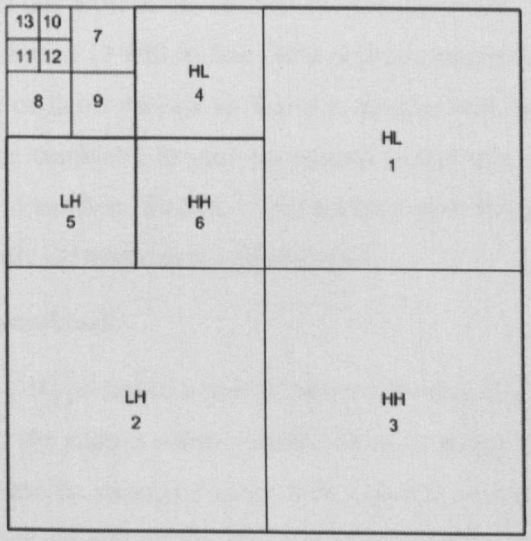


Fig. 6-1 Subband Numbers for Four Level Wavelet Transformed Frame

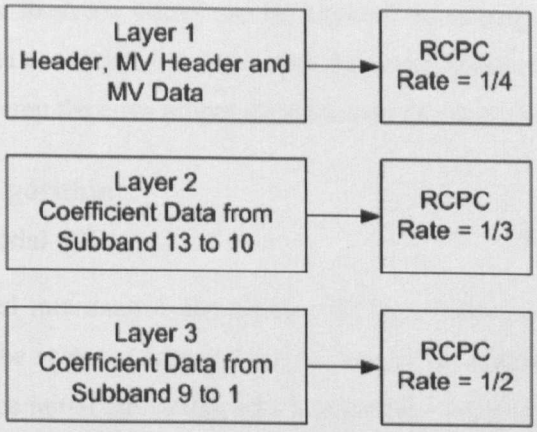


Fig. 6-2 The idea of Three Layers Protection

6.2.1.2 Variable Length Packetization

Another enhancement is concerning with the packetization. Currently, fixed length packetization is applied in which the packet length is fixed to a certain value and extra bits were chopped off regardless of the location of the information bits in the subband. Even though the existing approach is quite simple to implement, there are some factors which could affect the quality of the reconstructed video. For example, if the data inside a packet falls between the two subbands, the loss of this packet means the loss of two subband data i.e. later part of the first subband and the entire part of the successive subband data.

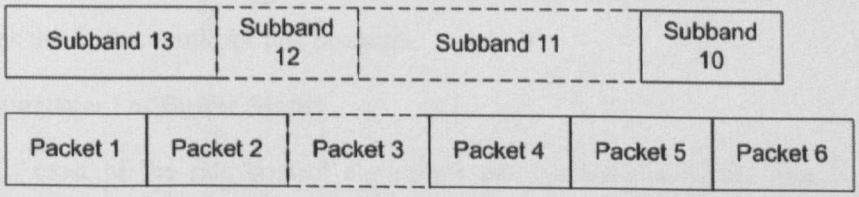


Fig. 6-3 The Problem with Fixed Length Packetization

The problem with fixed length packetization can be easily seen in Fig. 6-3. When the packet 3 has an error and can not correct this error at the channel decoder, both data from later part of the subband 12 and entire part of the subband 11 will be lost. This problem can be solved by mapping all the data from each subband to one or more packets so that the encoder will have the series of packets with each of their lengths (or the combined length) correspond to that of a subband and the length of the packet may vary from one to another. So that, if one packet is lost, this means losing only a portion of a subband or at most an entire corresponding subband data.

6.2.1.3 Periodic Intra Macroblock

J. Zheng and L. P. Chau in [84] proposed a new scheme to alleviate the effect of error propagation for the H.264 encoded video. By adding some periodic MBs in every fifth inter-frames, the average PSNR of a video sequence can be improved about 3 dB with 5% increase in bit rate when the MB loss rate is 15%. The selection of periodic MB is based on the distortion expectation of each MB and the

number of periodic MBs in every frame can be adjusted according to the available transmission bandwidth. The same idea can be used together with the presented error-resilient scheme (as a further enhancement) in order to stop the error propagation because of the packet error in the reference frame.

6.2.2 Rate Control Algorithm

6.2.2.1 Estimation of Initial *QF*

In the R-QF model based rate control algorithm with intra frame-only coding, either PSNR deep fading or fluctuation can be occurred especially in the low bitrate coding (i.e. 256 kbps target bit rate) because of the result of the initial *QF* setting which is too high for that particular target bitrate. In the result and discussion section of chapter 4, it was suggested that this problem can be solved either by inserting a certain number of frames (about 30 to 40 frames) at the beginning as a training sequence in order to achieve stability while encoding training sequence or setting the proper initial *QF*. In the former suggestion, the frames for the training sequence can be the exact replica of the first 30 frames of the video sequence being encoded. Even though it is a simple solution, addition of training sequence could generate unnecessary delay time in encoding and it will not work very well with the sequence where scene change is taking place at the beginning of the sequence. In the latter solution, a proper initial *QF* can be approximated from the coding mode whether using Intra frame-only or inter frame coding, target bitrate and frame rate. Building a simple mathematical model which can approximate the optimum initial *QF* from the available encoding parameters is required and can be considered as the further work for this research.

6.2.2.2 Requirement of Buffer Model

Even though most of the rate control algorithms can control the bitrate with a certain level of accuracy, the bitrate of the compressed bitstream can vary significantly because of the random nature of the contents of the video sequences (e.g. more textured regions or faster motion). Since the compressed digital video is often transmitted through the bandwidth limited channels, the bitrate variations need to be smoothed using buffering mechanisms at the encoder and decoder.

In video coding standards, a compliant bit stream must be decoded by a hypothetical decoder that is conceptually connected to the output of an encoder and consists of a decoder buffer, a decoder, and a display unit. This virtual decoder is known as the Hypothetical Reference Decoder (HRD). The encoder must create a bit stream so that the hypothetical decoder buffer does not overflow or underflow. This requirement can be strictly satisfied by the rate control algorithm implemented in the encoder.

Unfortunately, current Dirac's standard does not specify any HRD requirement. But, it is still worth to consider the implementation of buffer model in order to work together with the presented rate control algorithm so that the decoder buffer does not suffer under flow or over flow.

6.2.2.3 Recovery from the Bitrate Explosion Because of Scene Change

In the current version of Dirac, the encoder detects the number of intra MBs after motion estimation stage and inserts intra frame if the number of intra MBs is beyond the certain threshold. This is called scene change detection and insertion of intra frame normally takes place only when there is an abrupt scene change in the encoding video sequence. But, in the presented rate control algorithm, scene change scenario is not considered and so when the intra frame is inserted because of scene change, resulting bitrate may overshoot well beyond the target bitrate called bitrate explosion. In order to avoid this, the algorithm can be modified by resetting all its parameters once the scene change is occurred and recalculate parameters from the point of intra frame insertion.

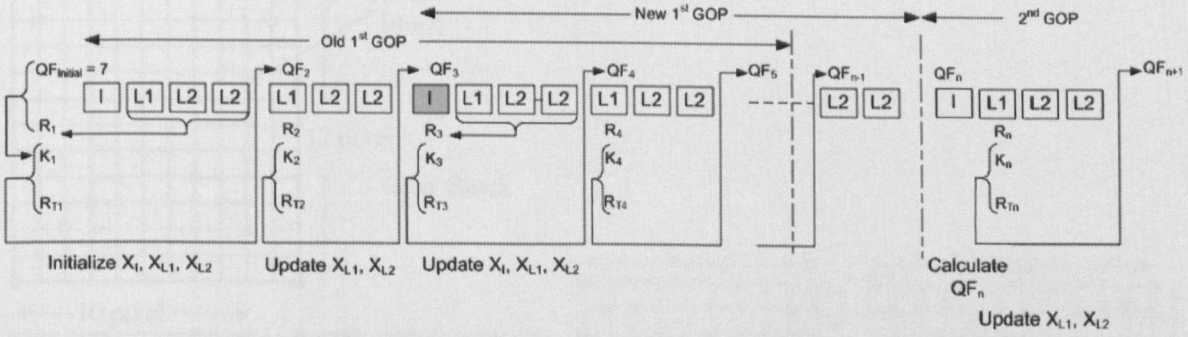


Fig. 6-4 Modification to the Presented Algorithm for Intra Frame Insertion because of Abrupt Scene Change

Fig. 6-4 shows modification to the presented rate control algorithm for the abrupt scene change condition. In figure, inserted *I* frame is shown with gray color. Once the *I* frame is inserted by the encoder because of abrupt scene change, the rate control algorithm needs to update the complexities of each frame type, replace the allocated number of bits left for the current GOP with total number of bits allocated for a GOP, reset the number of frames for each frame type used in a GOP and calculate R_{T3} and QF_4 (refers Fig. 6-4). After that the same procedure as mentioned in chapter 4 continues until the end of the sequence. It is important to note that new GOP starts from the inserted *I* frame, not from the end of the previous *I* frame.

6.2.2.4 Implementation of the Presented Rate Control Algorithm in H.264

Even though the algorithm is designed mainly for Dirac, it can also be used in other types of video codec, e.g. H.264 by incorporating a parameter which controls the quality of the encoded video sequence. Like in Dirac, H.264 also employs RDO principle and hence building a mathematical model which relates between a quality control parameter, QF and the Lagrangian multiplier, λ could lead to a simple and effective rate control mechanism for H.264.

6.2.3 Motion Estimation Strategy

6.2.3.1 Implementation of Partial SAD Calculation

As for the future work, presented partial cost function calculation can be implemented in the fast ME strategy to further reduce down the computational load. Without modifying to the algorithm, the cross partial cost function calculation pattern can be swapped directly with the existing full SAD calculation function. But, unfortunately, because of the application of overlapped block structure, there would be difficulty in implementing the cross pattern on irregular block shapes especially when the block is located at the margins.

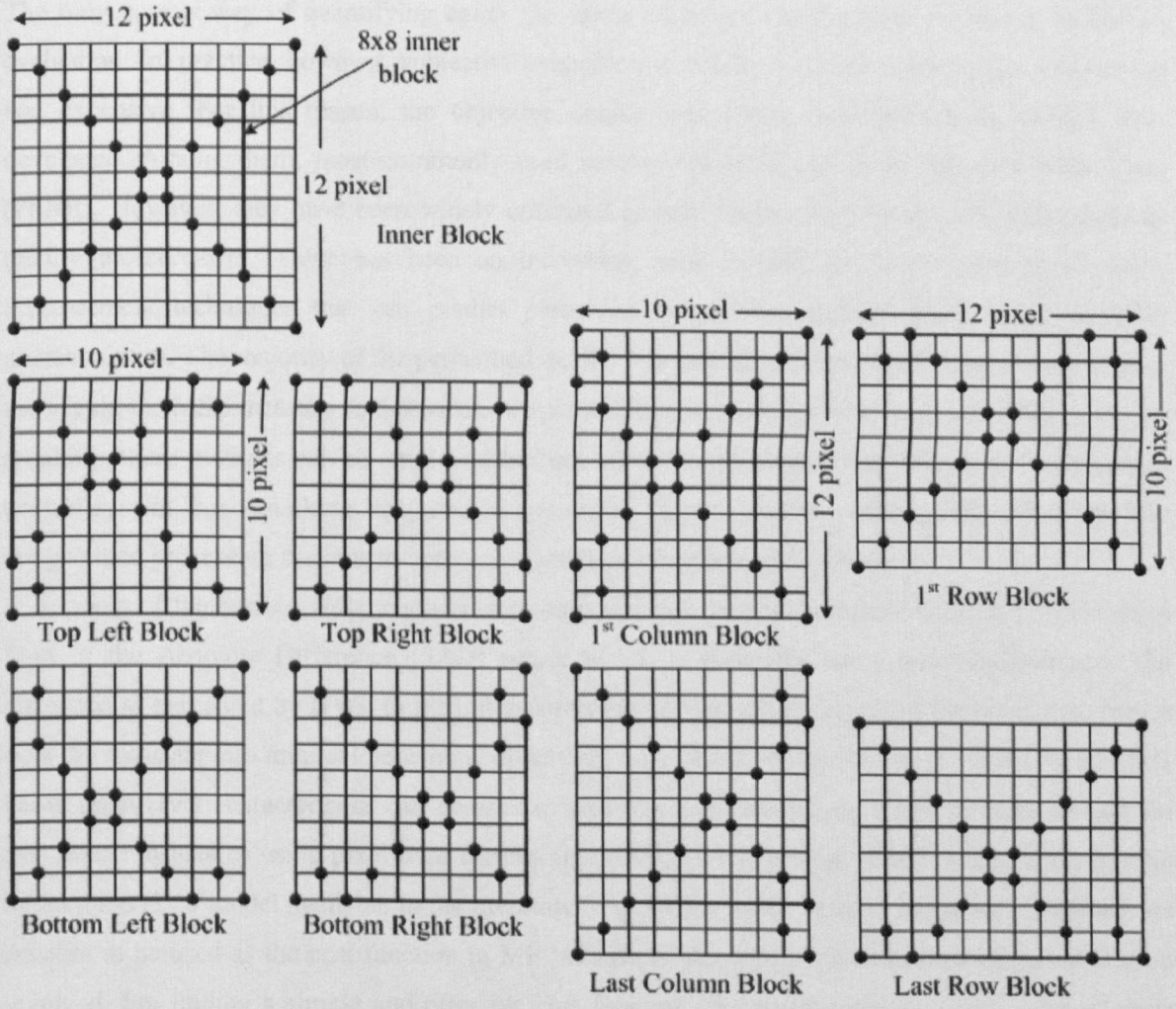


Fig. 6-5 Cross Shape Partial SAD Calculation Pattern for Dirac

Fig. 6-5 shows possible implementation of cross patterns on 12×12 block size with the overlapping block parameters; *xblen* = 12, *yblen* = 12, *xbsep* = 8 and *ybsep* = 8 which can be set independently by user. The detail overlapping block structure of Dirac for a MB is shown in Fig. 2-7 where the areas shown with stronger solid outline boundary are the inner blocks, which are believed to be non overlapping area if the offset or overlapping amount is reduced down to zero. The inner block has the dimension equal to x and y block separation and it is 8 for a particular block parameters

used in Fig. 6-5. Generally speaking, cross pattern in Fig. 6-5 requires only 24 points out of $12 \times 12 = 144$ and the reduction factor in term of the number of point to calculate SAD is $144/24 = 6$. The cross pattern is mainly based upon the inner block and so there are some irregularities upon the pattern especially for the marginal blocks.

But, further research may still be required in order to find the best pattern in partial SAD calculation for the overlapping block structure. But, it is important to note that there is no such problem with H.264 because of the non-overlapping nature of block.

6.2.3.2 Application of Perceptual Quality Based Cost Function in the ME of Dirac

The only correct way of quantifying either the visual image or video quality is through subjective evaluation. In, practice, however, subjective evaluation is usually too inconvenient, time-consuming and expensive. For this reason, the objective quality assessment distortion/quality metrics were developed. Among them, most commonly used metrics are MSE and Peak Signal-to-Noise Ratio (PSNR). However, they have been widely criticized as well for not correlating well with perceived quality measurement. There has been an increasing need recently to develop objective quality measurement techniques that can predict perceived image/video quality based upon the HVS characteristics. The majority of the perceptual quality assessment metrics have followed a strategy of modifying the MSE measure so that errors are penalized in accordance with their visibility. Generally speaking, these methods can be employed in three ways: monitoring image/video quality for quality control system, benchmarking image/video processing systems and algorithms, and embedding into image/video processing systems to optimize algorithms and parameters settings.

In most of the current video encoder, motion estimation process typically find the best match in Sum of the Absolute Difference (SAD) sense, which is generally not a good indication of the distortion as perceived by HVS. In [85], it was recognized that minimization of the pixel wise metric does not translate into minimal perceptual distortion. So, it would be interesting if the metric which is based upon HVS characteristics can be used in cost function calculation of ME in order to find the best match instead of using pixel wise metrics (i.e. SAD, MAD). Unfortunately, most of the metrics based upon HVS model available in the literature (e.g. SSIM, Video Quality Metrics (VQM)) are not suitable to be used as the cost function in ME because of the sophisticated mathematical calculation involved. So, finding a simple and effective cost function (distortion metrics) which is based upon HVS model could lead to the new era of ME. There will be a great deal of challenging involved in this research since it is creating a new dimension in ME area and nobody has ever thought about this before.

Reference:

[1] BBC Research & Development., Dirac Video Codec. *http://diracvideo.org/.* [Online]

[2] Schrödinger Release Download. *http://diracvideo.org/download/.* [Online]

[3] Dirac Software Releases Download. *http://sourceforge.net/project/showfiles.php?group_id=102564&package_id=112141.* [Online]

[4] BBC Research and Development., "Dirac Fundamentals." *http://dirac.sourceforge.net/documentation/algorithm/algorithm/toc.htm.* [Online]

[5] Mohammed Ghanbari., *Video Coding an introduction to standard codecs.* s.l. : The Institute of Electrical Engineers, 1999. ISBN 0852967624.

[6] BBC Research and Development., "http://diracvideo.org/download/specification/dirac-spec-latest.pdf." *Dirac Video Compression.* [Online]

[7] T. Borer, and T. Davies,., *Dirac video compression using open standards.* Sept. 2005. BBC R&D White Paper. WHP 117,.

[8] J. Zheng, and L. P. Chau., "Multiple Description Coding using Multiple Reference frame for robust Video Transmission." IEEE Int. Symp. Circuits and Systems, ISCAS 2005. Vol. 4, pp. 4006 – 4009, May 2005.

[9] C. M. Chen, Y.C. Chen, and C. M. Chen., "Multiple Description Motion Compensation Video Coding for MPEG-4 FGS over Lossy Packet Networks." IEEE Int. Conf. Multimedia and Expo, ICME2004. Vol. 2, pp. 1143 – 1146, Jun. 2004.

[10] M. Schaar, and D. S. Turaga., "Multiple Description Scalable Coding using Wavelet-Based Motion Compensated Temporal Filtering." IEEE Int. Conf. Image Processing, ICIP2003. Vol. 2, pp. 489-92, Sept. 2003.

[11] Leou, L. W. Kang and J. J., "An Error Resilient Coding Scheme for H.264 Video Transmission based on Data Embedding." Proceedings on Acoustics, Speech, and Signal Processing, 2004 (ICASSP '04). Vol. 3, pp. 257-260, May 2004.

[12] J. Kim, R. M. Mersereau, and Y. Altunbasak., "Error-Resilient Image and Video Transmission Over the Internet Using Unequal Error Protection." IEEE Transactions on Image Processing. Vol. 12, pp. 121-131, February 2003.

[13] C. D. Creusere., "A new method of robust image compression based on the embedded zerotree wavelet algorithm." IEEE Trans. Image Processing. Vol. 6, pp. 1436-1442, Oct. 1997.

[14] W. Tan, and A. Zakhor., "Resilient Compression of Video for Transmission over the Internet." 32nd Asilomar Conf. Signals, Systems & Computers. Vol. 1, pp. 243 - 247, Nov. 1998.

[15] P. G. Sherwood, and K. Zeger., "Progressive image coding for noisy channels." IEEE Signal Processing Lett. Vol. 4, pp. 189-191, July 1997.

[16] W. R. Heinzelman, and M. B. R. Talluri., "Unequal Error Protection of MPEG-4 Compressed Video." Int. Conf. Image Processing, ICIP99. Vol. 2, pp. 530-534, Oct. 1999.

[17] N. V. Boulgouris, N. Thomos, and M. G. Strintzis., "Transmission of Images Over Noisy Channels Using Error-Resilient Wavelet Coding and Forward Error Correction." Issue 12, Vol. 13, pp. 1170-1181, Dec. 2003.

[18] S. Cho, and W. A. Pearlman., "A Full-Featured, Error-Resilient, Scalable Wavelet Video Codec Based on the Set Partitioning in Hierarchical Trees (SPIHT) Algorithm." IEEE Trans. Circuits Syst. Video Technology, Vol. 12, pp. 157-171, Mar. 2002.

[19] B.-J. Kim, Z. Xiong, W. A. Pearlman, and Y. S. Kim., "Progressive video coding for noisy channels." J. Vis. Commun. Image Repres. Vol. 10, pp. 173-185, 1999.

[20] Z. Xiong, B.-J. Kim, and W. A. Pearlman., "Progressive video coding for noisy channel." Proc. IEEE Int. Conf. Image Processing (ICIP '98). Vol. 1, pp. 334-337, Oct. 1998.

[21] J. Zheng, and L. P. Chau., "Error-Resilient Coding of H.264 Based on Periodic Macroblock." IEEE Transactions on Broadcasting, Issue 2, Vol. 52, pp. 223-229, Jun. 2006.

[22] H264/AVC JM11Reference Software. *http://iphome.hhi.de/suehring/tml/.* [Online]

[23] H.264 Video Coding License Fees. *http://www.vialicensing.com/products/AVCH264VC/license.terms.html.* [Online]

[24] J. Hagenauer., "Rate-Compatible Punctured Convolutional Codes (RCPC Codes) and their Applications." IEEE Trans. Commun. Vol. 36, pp. 389-400, Apr. 1988.

[25] C. Berrou, A. Glavieux, and P. Thitimasjshima,., "Near Shannon limit error correcting coding and decoding: Turbo codes." Proc. of IEEE Int. Conf. on Commun. Geneva, Switzerland. pp. 1064-1070, May 1993.

[26] J. M. Shapiro., "Embedded image coding using zerotrees of wavelets coefficients." IEEE Trans. Signal Processing, Vol. 41, pp. 3445-3462, Dec. 1993.

[27] A. Said and W. Pearlman., "A new, fast and efficient image codec based on set partitioning in hierarchical trees." IEEE Trans. Circuits Syst. Video Technol., Vol. 6, pp. 243-250, June 1996.

[28] Q. Wang and M. Ghanbari., "Scalable coding of very high resolution video using the virtual zero-tree." IEEE Trans. Circuits Syst. Video Technol., Special Issue on Multimedia, pp. 719-729, Oct. 1997.

[29] S. A. Martucci, I. Sodagar, T. Chiang,., "A Zerotree Wavelet Video Coder." IEEE Trans. Circuits and Syst. for Video Technol., Issue 1, Vol. 7, pp. 109-118, Feb. 1997.

[30] G. D. Forney, Jr.,., "The Viterbi algorithm." Proc. IEEE. Vol. 61, pp. 169-176, Jan. 1994.

[31] G. Ungerboeck., "Channel Coding with Multilevel/Phase Signals." IEEE Transactions on Information Theory, Issue 1, Vols. 11-28, pp. 55-67, Jan. 1982.

[32] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv,., "Optimal decoding of linear codes for minimizing symbol error rate." IEEE Trans. Inf. Theory, pp. 284-287, Mar. 1974.

[33] W. E. Ryan., "Concatenated Convolutional Codes and Iterative Decoding." *Wiley Encyclopedia of Telecommunication.* s.l. : John Wiley and Sons, August 2003.

[34] T. Davies., "A Modified Rate-Distortion Optimization Strategy for Hybrid Wavelet Video Coding." IEEE International Conference on Acoustics, Speech, and Signal Processing, 2006 (ICASSP '06). Vol. 2, pp. 909 - 912, May 2006.

[35] C. Neeb, M.J. Thul, N. Wehn,., "Network-on-chip-centric approach to interleaving in high throughput channel decoders." IEEE Int. Sym. on Circuits and Systems (ISCAS'05), pp. 1766-1769, May. 2005.

[36] A. Giulietti, B. Bougard, V. Derudder, S. Dupont, J.-W. Weijers, L. Van der Perre,., "A 80 Mb/s low-power scalable turbo codec core." Proc. IEEE Custom Integrated Circuits Conference, pp. 389- 392, 2002.

[37] MPEG-2 Test Model 5., Apr. 1993. Doc. ISO/IEC JTC1/SC29 WG11/93-400.

[38] J. Corbera and S. Lei., *Rate Control for Low-Delay Video Communication.* ITU Study Group 16, Video Coding Experts Group, Portland. 1997. Documents Q15-A-20.

[39] T. Chiang and Y. Q. Zhang., "A new rate control scheme using quadratic rate distortion model." IEEE Trans. Circuits Syst. Video Technol.,, Issue 1, Vol. 7, pp. 246–250, Feb. 1997.

[40] Z.G. Li, F. Pan, K.P. Lim, G.N. Feng, X. Lin and S. Rahardaj,., *Adaptive basic unit layer rate control for JVT.* s.l. : 7th meeting, Pattaya II, Thailand, 7- 14, March, 2003. JVT-G012.

[41] Z. G. Li, F. Pan, K. P. Lim, and S. Rahardja,., "Adaptive rate control for H.264." International Conference on Image Processing, 2004. ICIP '04. pp. 745–748, Oct. 2004.

[42] Jianfeng Xu, and Yun He,., "A novel rate control for H.264." Proceedings of the International Symposium on Circuits and Systems, ISCAS '04. Vol. 3, pp. 809-812, May 2004.

[43] H. Xiong, J. Sun, S. Yu, J. Zhou, and C. Chen,., "Rate Control for Real-Time Video Network Transmission on End-To-End Rate-Distortion and Application-Oriented QoS." Issue 1, Vol. 51, pp. 122 – 132, March 2005.

[44] Y. Zhang, W. Yuan and S. Lin,., "A New Rate Control Scheme for H.264/AVC." International Conference on Digital Telecommunications, 2006. ICDT '06. pp. 13-18, 2006.

[45] D. K. Kwon, M. Y. Shen, and C.C. J. Kuo,., "Rate Control for H.264 Video With Enhanced Rate and Distortion Models." IEEE Transactions on Circuits and Systems for Video Technology, Issue 5, Vol. 17, pp. 517-529, May 2007.

[46] S. Hong, S. Yoo, S. Lee, H. Kang, and S. Hong,., "Rate Control of MPEG Video for Consistent Picture Quality." IEEE Trans. on Broadcasting, Issue 1, Vol. 49, pp. 1 – 13, March 2003.

[47] Zhihai He and Mitra, S.K.,., "Optimum bit allocation and accurate rate control for video coding via ρ-domain source modeling." IEEE Transactions on Circuits and Systems for Video Technology, Issue 10, Vol. 12, pp. 840 – 849, Oct. 2002.

[48] Zhihai He, Yong Kwan Kim and Mitra, S.K.,., "ρ-domain Source Modeling and Rate Control for Video Coding and Transmission." IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001 (ICASSP '01). Vol. 3, pp. 1773 – 1776, 7-11 May 2001.

[49] Yong Kwan Kim, Zhihai He and Mitra, S.K.,., "A novel linear source model and a unified rate control algorithm for H.263/MPEG-2/MPEG-4." IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001 (ICASSP '01). Vol. 3, pp. 1777 - 1780, 7-11 May 2001.

[50] Seonki Kim and Yo-Sung Ho., "Rate control algorithm for H.264/AVC video coding standard based on rate-quantization model." IEEE International Conference on Multimedia and Expo, 2004. ICME '04. Vol. 1, pp. 165 – 168, 27-30 June 2004.

[51] Seonki Kim, Seung-Hwan Kim and Yo-Sung Ho,., "Adaptive model-based quantization for H.264 video rate control." IEEE Region 10 Conference TENCON 2004. Vol. 1, pp. 331 – 334, 21-24 Nov. 2004.

[52] S. W. Ma, W. Gao, Y. Lu, and H. Lu,., *Proposed draft description of rate control on JVT standard.* s.l. : in Joint Video Team (JVT) of ISO/IECMPEG & ITU-T VCEG, JVT-F086, 6th Meeting , Awaji, Japan, December 2002.

[53] S. Ma, W. Gao and Y. Lu,., "Rate Distortion Analysis for H.264/AVC Video Coding and its Application to Rate Control." IEEE Transactions on Circuit and Systems for Video Technology, Issue 12, Vol. 15, December 2005.

[54] H. Wang, and S. Kwong,., "Rate-Distortion Optimization of Rate Control for H.264 With Adaptive Initial Quantization Parameter Determination." IEEE Transactions on Circuits and Systems for Video Technology, Issue 1, Vol. 18, pp. 140-144, January 2008.

[55] D. Lazar and A. Averbuch., "Wavelet-Based Video Coder via Bit Allocation." IEEE Transactions on Circuit and Systems for Video Technology, Issue 7, Vol. 11, pp. 815-832, July 2001.

[56] T. Zgaljic, N. Sprljan, E. Izquierdo,., "Bit-stream allocation methods for scalable video coding supporting wireless communications." Elsevier Journal of Signal Processing Image Communication, Vol. 22, pp. 298 - 316, 2007.

[57] K. P. Lim, G. Sullivan, T. Wiegand,., *Text Description of Joint Model Reference Encoding Methods and Decoding Concealment Methods.* Busan, Korea : s.n., April 2005. JVT-O079.

[58] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli,., "Image Quality Assessment: From Error Visibility to Structural Similarity." IEEE Transactions on Image Processing, Issue 4, Vol. 13, pp. 600-612, April 2004.

[59] T. Davies., "A Modified Rate-Distortion Optimization Strategy for Hybrid Wavelet Video Coding." IEEE International Conference on Acoustics, Speech, and Signal Processing, 2006 (ICASSP '06). Vol. 2, pp. 909 - 912, 7-11 May 2006.

[60] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro,., "Motion compensated interframe coding for video conferencing." In Proc. Nat. Telecommun. Conf., New Orleans, LA. pp. G.5.3.1-G.5.3.5, 1981.

[61] R. Li, B. Zeng, and M. L. Liou,., "A new three-step search algorithm for block motion estimation." IEEE Transactions on Circuits and Systems for Video Technology, Issue 4, Vol. 4, pp. 438-442, 1994.

[62] L. M. Po, and W. C. Ma,., "A novel four-step search algorithm for fast block motion estimation." IEEE Transactions on Circuits and Systems for Video Technology, Issue 3, Vol. 6, pp. 313-317, 1996.

[63] Shan Zhu, and Kai-Kuang Ma,., "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation." IEEE Transactions on Image Processing, Issue 2, Vol. 9, pp. 287-290, February 2000.

[64] Yao Nie, and Kai-Kuang Ma,., "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation." IEEE Transactions on Image Processing, Issue 12, Vol. 11, pp. 1442-1448, December 2002.

[65] Yao Nie, and Kai-Kuang Ma,., "Adaptive Irregular Pattern Search with Matching Prejudgment for Fast Block-Matching Motion Estimation." IEEE Transactions on Circuits and Systems for Video Technology, Issue 6, Vol. 15, pp. 789-794, June 2005.

[66] H. Nam, S. Lim,., "A New Motion Estimation Scheme Using a Fast and Robust Block Matching Algorithm." WSEAS Trans. on Information Science & Applications, Issue 11, Vol. 3, pp. 2292-2299, November 2006.

[67] Y. Shi, C. Yi and Z. Cai,., "Multi-Direction Cross-Hexagonal Search Algorithms for Fast Block Motion Estimation." WSEAS Trans. on Computers, Issue 6, Vol. 6, pp. 959-963, June 2007.

[68] C. L. Lin, J. J. Leou,., "An Adaptive Fast Search Motion Estimation Algorithm for H.264." WSEAS Trans. on Communications, Issue 7, Vol. 4, pp. 396-406, July 2005.

[69] M. Bierling., "Displacement estimation by Hierarchical block matching." Procs. Visual Communications and Image Processing, SPIE. Vol. 1001, pp. 942-951, 1988.

[70] Y. Q. Zhang and S. Zafar., "Motion-compensated wavelet transform coding for color video compression." IEEE Trans. Circuits Syst. Video Technology, Issue 3, Vol. 2, pp. 285–296, Sep. 1992.

[71] S. Zafar, Y. Q. Zhang and B. J. Jabbari,., "Multiscale video representation using multiresolution motion compensation and wavelet decomposition." IEEE J. Sel. Areas Commun. Vol. 1, pp. 24–35, Jan. 1993. 1.

[72] J. Zan, M. O. Ahmad and M. N. S. Swamy,., "Comparison of Wavelets for Multiresolution Motion Estimation." IEEE Transactions on Circuits and Systems for Video Technology, Issue 3, Vol. 16, pp. 439-446, March 2006.

[73] J. Zan, M.O. Ahmad and M.N.S. Swamy,., "A multiresolution motion estimation technique with indexing." IEEE Transactions on Circuits and Systems for Video Technology, Issue 2, Vol. 16, pp. 157-165, Feb. 2006.

[74] F. Lopes and M. Ghanbari., "Improved motion estimation by spatial transformations and overlap." IEEE International Conference on Image Processing, ICIP'98. October 1998.

[75] Y. Nakaya and H. Harashima., "Motion Compnesation based on spatial transformations." IEEE Transactions on Circuits and Systems for Video Technology, Issue 3, Vol. 4, pp. 339-356, June 1994.

[76] G. J. Sullivan and R. L. Baker., "Motion compensation for video compression using control grid interpolation." International Conference on Acoustics, Speech and Signal Processing. pp. 2713-2720, 1991.

[77] F. Lopes and M. Ghanbari., "Hierarchical motion estimation with spatial transforms." International Conference on Image Processing, 2000. Vol. 2, pp. 558-561, Sept. 2000.

[78] C. K. Cheung and L. M. Po., "Normalized Partial Distortion Search Algorithm for Block Motion Estimation." IEEE Transactions on Circuits and Systems for Video Technology , Issue 3, Vol. 10, pp. 417-422, APRIL 2000.

[79] Z. Chen, P. Zhou and Y. He,., *Fast Integer Pel and Fractional Pel Motion Estimation for JVT.* 5-13 December, 2002. JVT-F017.

[80] J. Xu, Z. Chen and Y. He,., "Efficient Fast ME Predictions and Early-termination Strategy Based on H.264 Statistical Characters." Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Vol. 1, pp. 218-222, 15-18 Dec. 2003.

[81] H.-Y.C.Tourapis and A.M. Tourapis., "Fast motion estimation within the H.264 codec." Proceedings. 2003 International Conference on Multimedia and Expo, 2003. ICME '03. Vol. 3, pp. 517-520, 6-9 July 2003.

[82] B. Liu and A.Zaccarin., "New fast algorithm for the estimation ofblock motion vectors." IEEE Trans. Circuits Video Technol., Vol. 3, pp. 148–157, Apr. 1993.

[83] C. H. Cheung and L. M. Po., "Adjustable Partial Distortion Search Algorithm for Fast Block Motion Estimation." IEEE Trans. Circuits Syst. Video Technol., Vol. 13, pp. 100–110, Jan. 2003.

[84] J. Zheng and L. P. Chau., "Error-Resilient Coding of H.264 Based on Periodic Macroblock." IEEE Transactions on Broadcasting, Issue 2, Vol. 52, pp. 223-229, Jun. 2006.

[85] S. Winkler., "A perceptual distortion metric for digital colour video." Proc. SPIE. Vol. 3644, pp. 175-184, 1999.

[86] H. Eeckhaut, B. Schrauwen, M. Christaens, and J.V. Campenhout,., "Speeding up Dirac's Entropy Coder." Proc. 5th WSEAS Int. Conf. on Multimedia, Internet and Video Technologies, Greece,. pp. 120-125, Aug. 2005.

Appendix A

Dirac's Bitstream Syntax

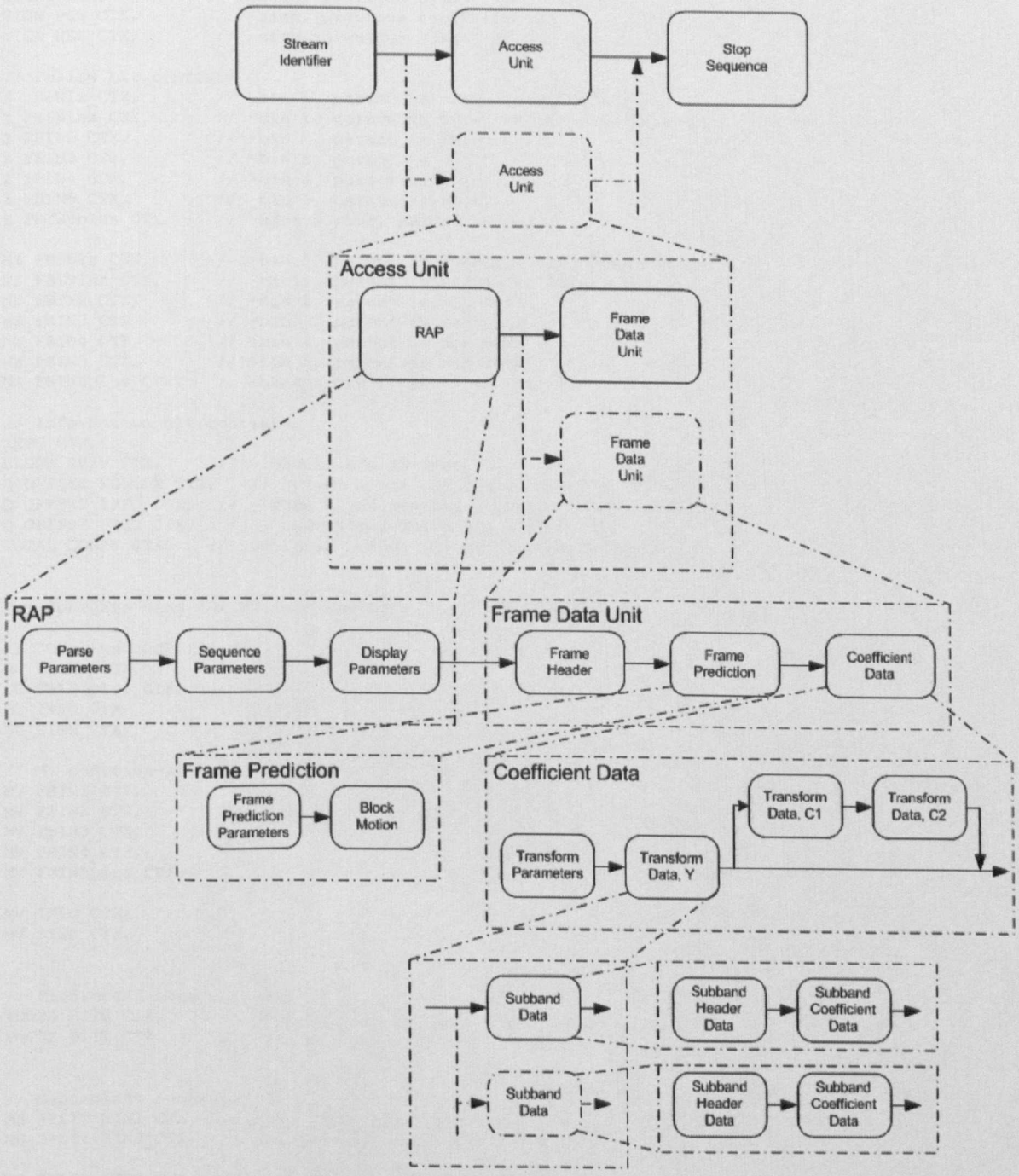


Fig. A1 Block Diagram of Dirac Encoder's Bitstream Syntax

Appendix B

Contexts Used in Dirac

```
//used for residual coding
SIGN0_CTX,              // -sign, previous symbol is 0
SIGN_POS_CTX,           // -sign, previous symbol is +ve
SIGN_NEG_CTX,           // -sign, previous symbol is -ve

// Follow bit contexts
Z_FBIN1z_CTX,           // -bin 1, parent is zero, neighbours zero
Z_FBIN1nz_CTX,          // -bin 1, parent is zero, neighbours non-zero
Z_FBIN2_CTX,            // -bin 2, parent is zero
Z_FBIN3_CTX,            // -bin 3, parent is zero
Z_FBIN4_CTX,            // -bin 4, parent is zero
Z_FBIN5_CTX,            // -bin 5, parent is zero
Z_FBIN6plus_CTX,        // -bins 6 plus, parent is zero

NZ_FBIN1z_CTX,          // -bin 1, parent is non-zero, neighbours zero
NZ_FBIN1nz_CTX,         // -bin 1, parent is non-zero, neighbours non-zero
NZ_FBIN2_CTX,           // -bin 2, parent is non-zero
NZ_FBIN3_CTX,           // -bin 3, parent is non-zero
NZ_FBIN4_CTX,           // -bin 4, parent is non-zero
NZ_FBIN5_CTX,           // -bin 5, parent is non-zero
NZ_FBIN6plus_CTX,       // -bins 6 plus, parent is non-zero

// Information bit contexts
INFO_CTX,
BLOCK_SKIP_CTX,         // - blocks are skipped
Q_OFFSET_FOLLOW_CTX,    // - code block quantiser offset magnitude
Q_OFFSET_INFO_CTX,      // - code block quantiser offset info context
Q_OFFSET_SIGN_CTX,      // - code block quantiser offset sign
TOTAL_COEFF_CTXS        // The total number of coefficient contexts


//! Contexts used for MV data coding

// DC value contexts //
DC_FBIN1_CTX,
DC_FBIN2plus_CTX,
DC_INFO_CTX,
DC_SIGN_CTX,

// MV contexts //
MV_FBIN1_CTX,
MV_FBIN2_CTX,
MV_FBIN3_CTX,
MV_FBIN4_CTX,
MV_FBIN5plus_CTX,

MV_INFO_CTX,
MV_SIGN_CTX,


// Prediction mode contexts
PMODE_BIT0_CTX,         // -bit 0, prediction mode value
PMODE_BIT1_CTX,         // -bin 1, prediction mode value


// Macroblock contexts
MB_SPLIT_BIN1_CTX,      // bin 1, MB split mode vals
MB_SPLIT_BIN2_CTX,      // bin 2, MB split mode vals. Bin 3 not required

MB_SPLIT_INFO_CTX,      // info context for MB split mode
TOTAL_MV_CTXS           // The total number of MV contexts
```

Appendix C

Dirac's Bitstream Syntax after Source Coding

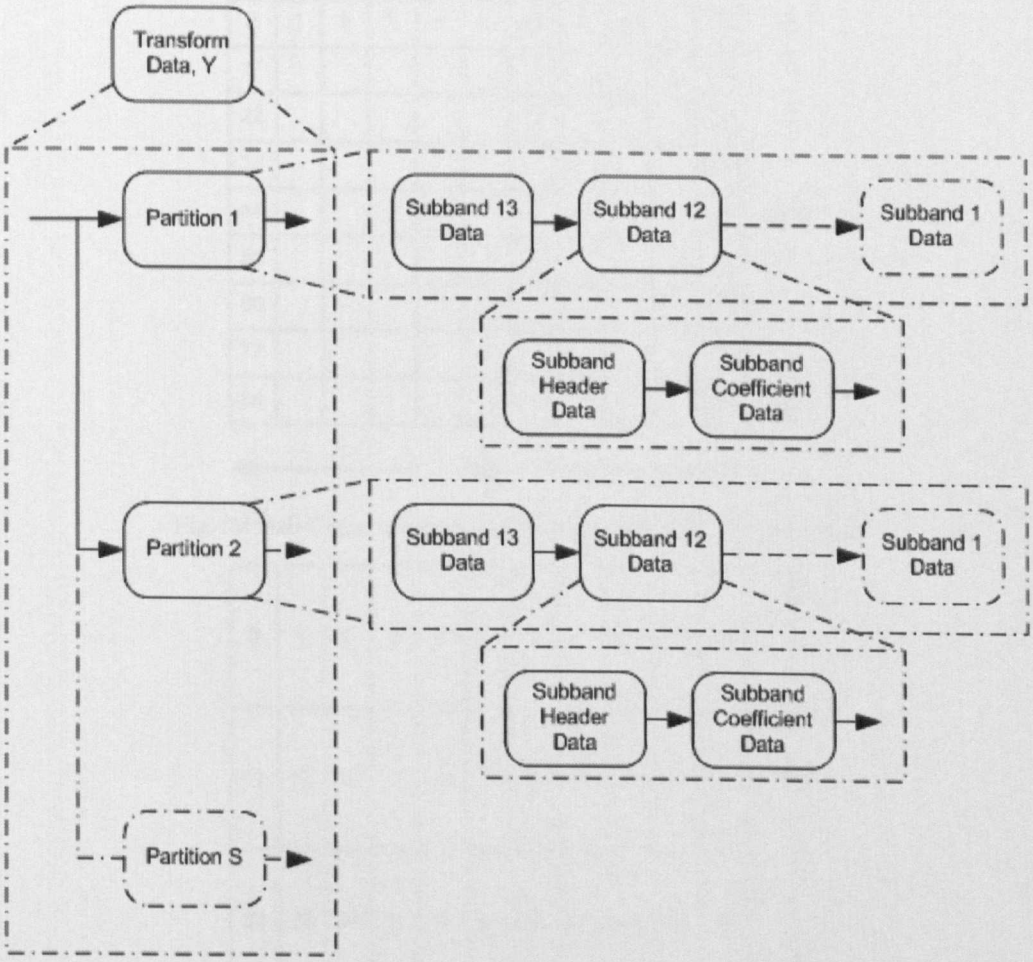


Fig. C1 The block diagram of the Bitstream Syntax of Dirac with Error Resilient Coding

Appendix D

Different Frame Partitioning Formats

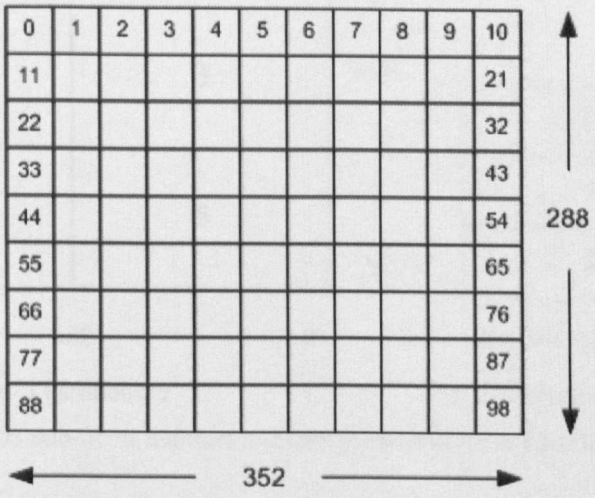


Fig. D1 Sub-frame numbers and their location (CIF, 99 partitions)

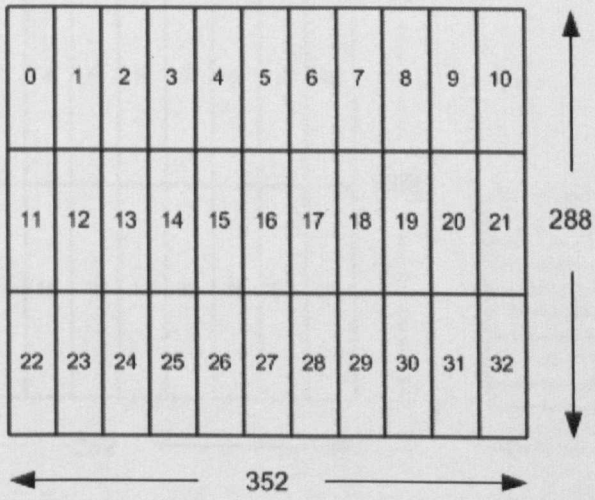
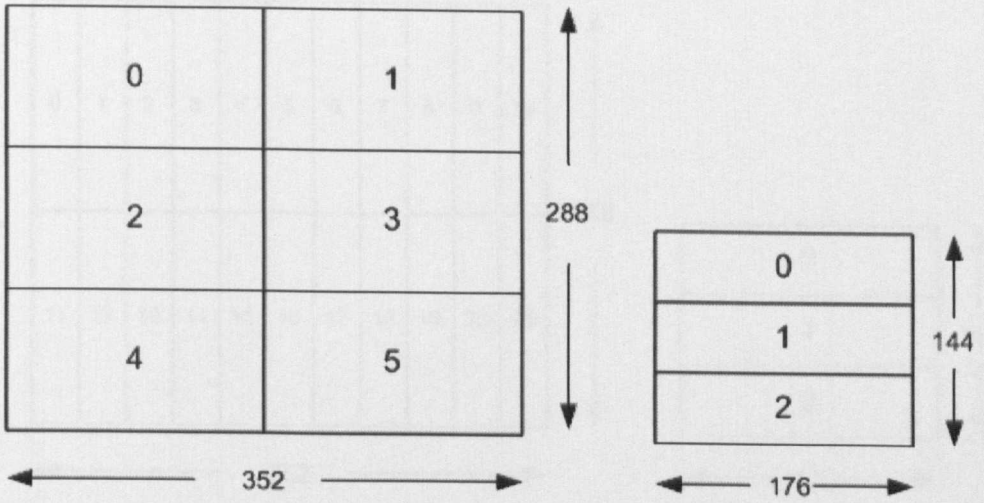
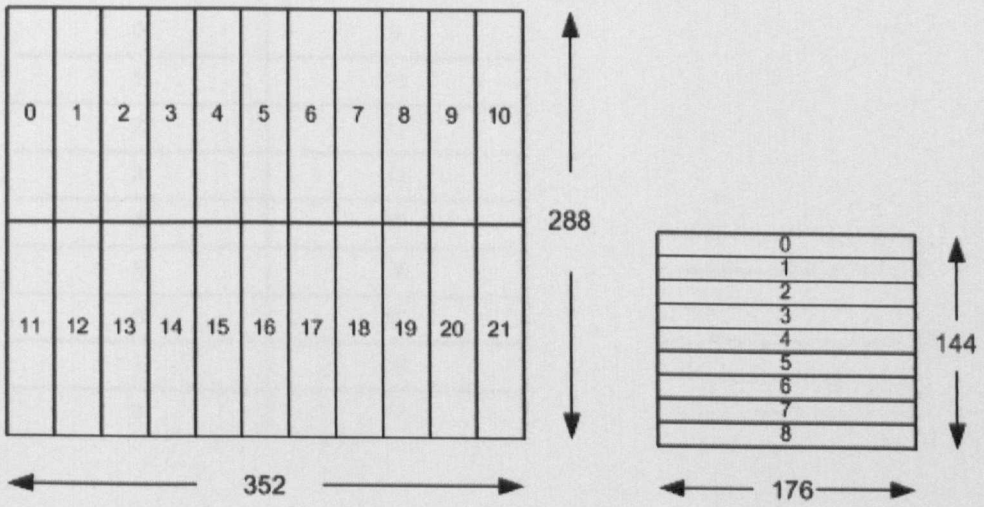


Fig. D2 Sub-frame numbers and their location (CIF, 33 partitions)



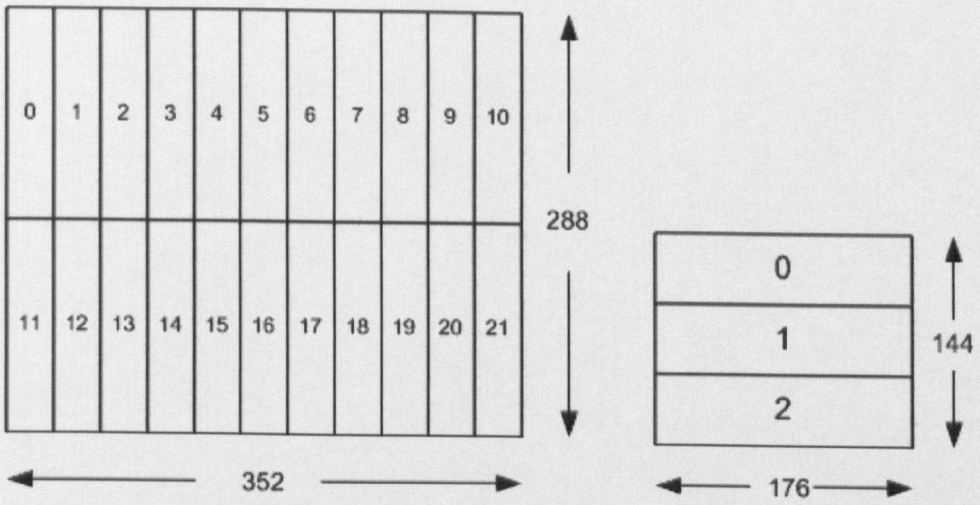
6_3 Partitions, Y 6_3 Partitions, U,V

Fig. D3 Sub-frame numbers and their location (CIF, 6_3 partitions)



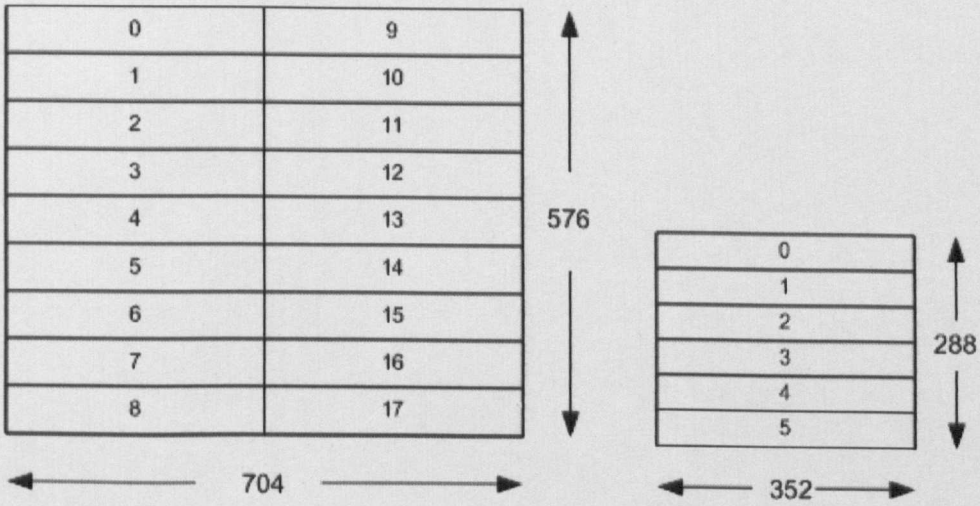
22_9 Partitions, Y 22_9 Partitions, U,V

Fig. D4 Sub-frame numbers and their location (CIF, 22_9 partitions)



22_3 Partitions, Y 22_3 Partitions, U,V

Fig. D5 Sub-frame numbers and their location (CIF, 22_3 partitions)



18_6 Partitions, Y 18_6 Partitions, U,V

Fig. D6 Sub-frame numbers and their location (SD576, 18_6 partitions)