

**A VIDEO COMPRESSION ALGORITHM USING
ZEROTREE WAVELET AND HIERARCHICAL
FINITE STATE VECTOR QUANTIZATION**

138898

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of
Dokuz Eylül University
In Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy in Electrical and Electronics Engineering, Electrical
and Electronics Program**

**by
İlker KILIÇ**

**October, 2003
İZMİR**

138898
İZMİR ÜNİVERSİTESİ
DOĞRU YÖNÜNDEN KONTROL

Ph.D. THESIS EXAMINATION RESULT FORM

We certify that we have read the thesis, entitled “A VIDEO COMPRESSION ALGORITHM USING ZEROTREE WAVELET AND HIERARCHICAL FINITE STATE VECTOR QUANTIZATION” completed by İLKER KILIÇ under supervision of Assist.Prof.Dr. Reyat YILMAZ and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.



Assist.Prof.Dr. Reyat YILMAZ

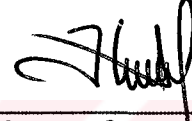
Supervisor



Assist. Prof. Dr. Adil ALPKOÇAK

Jury Member

(Thesis Committee Member)



Assist.Prof. Dr. Haldun SARNEL

Jury Member

(Thesis Committee Member)



Prof. Dr. Mete SEVERCAN

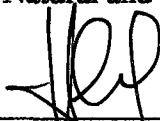
Jury Member



Prof. Dr. Mustafa GÜNDÜZALP

Jury Member

Approved by the
Graduate School of Natural and Applied Sciences



Prof.Dr.Cahit Helvacı

Director

ACKNOWLEDGMENTS

I would like to thank my supervisor, Assist. Prof. Dr. Reyat YILMAZ, for the support, encouragement, and time which he has given me, for the ideas he has shared with me, and for the opportunities which made this work possible.

I would also like to thank the Ph.D. thesis committee members, Assist. Prof. Dr. Haldun SARNEL, Assist. Prof. Dr. Adil ALPKOÇAK, for the ideas they have shared with me and for their questions and suggestions which have improved the quality of this thesis.

Thanks to the Dokuz Eylül University, and the Department of Electrical and Electronics Engineering staff for their supports during my thesis work.

İlker KILIÇ

ABSTRACT

In this thesis a video compression technique with comparable low bit rates is presented. The proposed encoder is similar to other motion compensated, block based discrete cosine transform (DCT) video coding standards such as MPEG-1/2, H.261/3 and MPEG-4/Verification Model (VM). The novel feature of this coding scheme is the combination of Zerotree Wavelet (ZTW) structure with the Hierarchical Finite State Vector Quantization (HFSVQ).

In this study, performances of the popular block motion estimation algorithms are compared. The simulation results show that the New Diamond Search (NDS) performs better than the compared algorithms. Because of that as a block motion estimation algorithm, the NDS is used to track the local motion. The motion vectors of the macroblocks are Huffman encoded. The overlapping block motion compensation (OBMC) technique which is an advance scheme in the H.263 standard is used after NDS in order to reduce the block artifacts. A three level discrete wavelet transform (DWT) is used instead of DCT to remove the spatio-temporal correlation. It is seen that the DWT coefficients are Gaussian shape distributed in each level except the lowest frequency subband. Therefore the proposed encoder uses a linear quantizer for the lowest frequency subband and the nonlinear Lloyd-Max quantizer for the rest of the subbands. Since the ZTW structure consists of different size of wavelet blocks that contain similar information, then it is very suitable for these blocks to be coded by HFSVQ.

The computer simulation of whole system is done by using the Object Oriented Borland C++ Builder software. The simulation results of the proposed encoder was obtained using three types of standard video sequences; QCIF Akiyo which contains a few movements at the rate of 10 kb/s and 5 fr/s, QCIF Carphone which contains relatively more movements than Akiyo at the rate of 30 kb/s and 10 fr/s and QCIF

Coastguard which contains more movements than the others at the rate of 48 kb/s and 7.5 fr/s. Average PSNR over the entire coded sequences show that the proposed ZTW-HFSVQ video compression technique achieves comparable performance over MPEG-4/VM, H.263, ZTE and EZW for both I and P frames.

Keywords: Video compression, Zerotree Wavelet, Vector Quantization



ÖZET

Bu tezde çok düşük bit hızları için tasarlanmış bir video kodlayıcısı sunulmaktadır. Önerilen video kodlama tekniği, hareket kompanzasyonlu, blok tabanlı, ayırık kosinüs dönüşümü kullanan MPEG-1/2, H.261/3 ve MPEG-4/VM standard video kodlama tekniklerine benzemektedir. Kodlayıcının en önemli ve yeni özelliği ise Sıfır-Ağaç Dalgacık Yapının (SAD), Hiyerarşik Sonlu Durum Vektör Nicemleyici (HSDVN) ile birleştirilmesidir.

Bu çalışmada, güncel hareket kestirim algoritmalarının başarımları karşılaştırılmış ve yapılan simülasyonlar sonucunda Yeni Mücevher Araştırma (YMA) algoritmasının karşılaştırılan diğer algoritmalarından üstün olduğu görülmüştür. Bu nedenle blok hareket kestirim algoritması olarak YMA seçilmiştir. Macrobloklara ait hareket vektörleri Huffman algoritması ile kodlanmıştır. YMA algoritmasından sonra oluşan blok sınır süreksizliklerini azaltmak için H.263 standardında kullanılan Örtüşmeli Blok Hareket Kompanzasyon yöntemi (ÖBHK), Ayırık Kosinüs Dönüşümü yerine ise üç kademeli ayırık dalgacık dönüşümü (ADD) kullanılmıştır. En düşük frekans katmanı haricindeki diğer tüm ADD katmanlarına ait dalgacık katsayıları Gaus dağılımına sahiptirler. Bu nedenle önerilen kodlayıcıda, en düşük frekans katmanı için doğrusal, diğer tüm katmanlar için ise doğrusal olmayan Lloyd-Max nicemleyici kullanılmaktadır. SAD yapısı, benzer bilgiler içeren değişik büyüklüklerdeki dalgacık bloklarından oluşmaktadır. Bu yüzden bu bloklar HSDVN tekniği ile kodlanmaya uygun bir yapıdadır.

Tüm sistemin bilgisayar simülasyonu nesne tabanlı Borland C++ Builder yazılımı ile gerçekleştirilmiştir. Önerilen yeni kodlayıcının simülasyonunda üç farklı hareket hızına sahip standard QCIF video serileri kullanılmıştır. Bunlar, az hareket içeren 5 çerçeve/sn ve 10 kb/sn'lik Akiyo, Akiyo serisinden daha fazla hareket içeren 10

çerçeve/sn ve 30 kb/sn'lik Carphone ve diğerlerinden çok daha fazla hareket içeren 7.5 çerçeve/sn ve 48 kb/sn'lik Coastguard serileridir. Bütün kodlanmış serilerin ortalama PSNR değerleri göstermiştir ki, önerdiğimiz SAD-HDSVN video kodlama tekniği hem çerçeveler içi hem de çerçeveler arası kodlamada, MPEG-4/VM, H.263, ZTE ve EZW tekniklerine göre daha iyi bir başarımla sağlamıştır.

Anahtar Sözcükler: Video sıkıştırma, Sıfır Ağaç Dalgacık Yapı, Vektör Nicemleme



CONTENTS

	Page
Contents	viii
List of Tables	xi
List of Figures	xiii

Chapter One

INTRODUCTION

1. Introduction	1
1.1 Motivation of the Project	1
1.2 Overview of the Video Compression Standards	2
1.3 Research Objectives	3
1.4 An Outline of the Thesis	4

Chapter Two

DIGITAL VIDEO CHARACTERISTICS

2. Digital Video Characteristics	7
2.1 RGB Color Coordinate	7
2.2 YUV Color Coordinate	8
2.3 YIQ Color Coordinate	8
2.4 YCbCr Color Coordinate	9
2.5 Common Intermediate Format (CIF)	11
2.6 Source Input Format (SIF)	13
2.7 Performance Measures	14

Chapter Three

BLOCK MOTION ESTIMATION AND COMPENSATION

	Page
3. Block Motion Estimation and Compensation	16
3.1 Introduction	16
3.2 Full Search (FS)	17
3.3 Two Dimensional Logarithmic Search (TDL)	17
3.4 One at a Time Search (OTS)	19
3.5 Three Step Search (3SS)	19
3.6 An Improved Three Step Search (I3SS)	20
3.7 A New Three Step Search (N3SS)	23
3.8 A Novel Four-Step Search (N4SS)	25
3.9 Simple and Efficient Search (SES)	27
3.10 New Diamond Search (NDS)	29
3.11 Computer Simulation of Block Motion Estimation Algorithms and the Performance Comparison	32
3.12 The Performance Results of Block Motion Estimation Algorithms with Overlapping Block Motion Compensation	42
3.13 Conclusion	49

Chapter Four

DISCRETE WAVELET TRANSFORM

4. Discrete Wavelet Transform	50
4.1 Introduction	50
4.2 The Wavelet Characteristics	50
4.3 The Wavelet Expansion	52
4.4 The Scaling Function	53
4.5 Multiresolution Analysis	54
4.6 The Wavelet Functions Used by the Introduced Encoder	55
4.7 Conclusion	63

Chapter Five

QUANTIZATION

	Page
5. Quantization	64
5.1 Scalar Quantization	64
5.2 Lloyd-Max Quantization	67
5.3 Vector Quantization	72
5.3.1 Codebook design by LBG algorithm	73
5.4 Hierarchical Finite State Vector Quantization	76
5.4.1 Structure map construction	76

Chapter Six

A VIDEO COMPRESSION TECHNIQUE USING ZEROTREE WAVELET AND HIERARCHICAL FINITE STATE VECTOR QUANTIZATION

6. A Video Compression Technique Using Zerotree Wavelet and HFSVQ	79
6.1 Introduction	79
6.2 Motion Estimation and Compensation	81
6.3 Discrete Wavelet Transform of QCIF Video Frames	85
6.4 Lloyd-Max Quantization of Wavelet Coefficients	88
6.5 Zerotree Structure	91
6.6 HFSVQ and Rate Control	94
6.7 Simulation Results	96

Chapter Seven

CONCLUSION

7. Conclusion	105
References	109

LIST OF TABLES

	Page
Table 2.1 Digital video formats specified by ITU-R 601, JPEG(J), H.261(H) And MPEG-1(M) (I: interlaced, P: progressive)	9
Table 2.2 Parameters of CIF and TV systems	13
Table 2.3. Conversion of two source formats to SIF	13
Table 3.1 Average MSE values for 100 Football frames	33
Table 3.2 Average SPN values for 100 Football frames	34
Table 3.3 Average MSE values for 100 Miss America frames	34
Table 3.4 Average SPN values for 100 Miss America frames	35
Table 3.5 MSE results for 75 th and 76 th frames of Football sequence without OBMC	47
Table 3.6 MSE results for 75 th and 76 th frames of Football sequence with OBMC	47
Table 3.7 Decrease in MSE for 75 th and 76 th frames of Football sequence by OBMC	47
Table 3.8 MSE results for 27 th and 28 th frames of Miss America without OBMC	47
Table 3.9 MSE results for 27 th and 28 th frames of Miss America with OBMC	48
Table 3.10 Decrease in MSE for 27 th and 28 th frames of Miss America by OBMC	48
Table 4.1 Haar, Daubechies and Biorthogonal wavelet and scaling function coefficients	59
Table 6.1 Probabilities of motion vector differences and corresponding Huffman codes for reconstructed 48 th frame of Akiyo	84
Table 6.2 H.261 VLC Table of motion vector differences	84
Table 6.3 The maximum and minimum Biorthogonal 9.3 DWT coefficient values for layers of 256x256 LENA image	88

	Page
Table 6.4 For a 176x144 QCIF frame, possible block sizes and number of zerotree blocks in each level of DWT decomposition	94
Table 6.5 QCIF first I-Frame Coding PSNR Results in dB	97
Table 6.6 P Frame PSNR results for AKIYO at the rate of 5 fr/s and 10 kb/s	99
Table 6.7 Carphone QCIF sequence at the rate of 10 fr/s and 30 kb/s	100
Table 6.8 Coastguard QCIF video sequence at the rate of 10 fr/s and 48 kb/s ...	103
Table 6.9 The average results of H.263, MPEG-4/VM and ZTW-HFSVQ in dB	104



LIST OF FIGURES

	Page
Figure 2.1 The RGB color cube	7
Figure 2.2 Positioning of YCbCr pixels for a) 4:4:4 b) 4:2:2 c) 4:1:1 d) 4:2:0 ...	10
Figure 2.3 Information exchange via the common intermediate format (CIF)	12
Figure 2.4 CIF and QCIF resolutions	12
Figure 2.5 Pre and post-processing at the encoder and decoder	14
Figure 2.6 Decimation and interpolation filters for conversion between ITU-R 601 and SIF	14
Figure 3.1 The block motion estimation structure	17
Figure 3.2 An example of TDL algorithm	18
Figure 3.3 One at a Time Search	19
Figure 3.4 Three Step Search Algorithm	20
Figure 3.5 Search patterns of the I3SS a) First step centered on centre pixel b) Second step centred on a corner pixel c) Second step centred on a middle pixel d) Third step	21
Figure 3.6 Two examples of I3SS search paths	22
Figure 3.7 The block diagram of the N3SS algorithm	24
Figure 3.8 Three examples of N3SS search paths	24
Figure 3.9 Search patterns of the N4SS. a) First step b) second / third step c) second / third step d) fourth step	26
Figure 3.10 Two different search paths of N4SS	27
Figure 3.11 Search pattern in the first step of SES	28
Figure 3.12 Search pattern corresponding to each selected quadrant a) quadrant I, b) quadrant II, c) quadrant III, d) quadrant IV e) (numbers in the circles show the first and the second phases) ..	28
Figure 3.13 An example for the SES algorithm	29
Figure 3.14 Two search patterns of the NDS a) LDSP b) SDSP	30

Figure 3.15 Three cases of checking – point overlapping in LDSP when the MSE point found in the previous search step (number 1) is located at a) one of the corner points, b) one of the edge points, and c) the center point. The points of number 2 are the new checking points	31
Figure 3.16 A search path example of NDS	32
Figure 3.17 The Football sequence MSE values at 4x4 macroblock size for (a) FS, 3SS, NDS, N3SS, SES (b) FS, TDL, OTS, N4SS	35
Figure 3.18 The Miss America MSE values at 4x4-macroblock size for (a) FS, N3SS, TDL, NDS (b) FS, OTS, 3SS, SES, N4SS	36
Figure 3.19 The Football sequence average SPN values at different macroblock Sizes for (a) N4SS, N3SS, 3SS, OTS (b) SES, TDL, NDS, I3SS ...	37
Figure 3.20 The Miss America sequence average SPN values at different Macroblock sizes for (a) SES, TDL, N4SS, I3SS (b) NDS, 3SS, N3SS, OTS	39
Figure 3.21 The Football sequence average MSE values at different macroblock Sizes for (a) FS, N3SS, NDS, 3SS (b) FS, OTS, TDL, N4SS, SES	40
Figure 3.22 The Miss America sequence average MSE values at different macroblock sizes for (a) FS, N3SS, OTS, TDL (b) FS, 3SS, NDS, N4SS, SES	41
Figure 3.23 OBMC for upper left half of the Macroblock	43
Figure 3.24 OBMC matrixes for a 16x16 macroblock. Each small luminance block size is 8x8 (a) matrix for motion vector of current luminance block (b) matrix for motion vector of top / bottom luminance block (c) matrix for motion vector of left / right luminance block	43
Figure 3.25 OBMC matrixes for a 16x8 macroblock. Each small luminance block size is 8x4 (a) matrix for motion vector of current luminance block (b) matrix for motion vector of top / bottom luminance block (c) matrix for motion vector of left / right luminance block	44

Figure 3.26 OBMC matrixes for a 8x16 macroblock. Each small luminance block size is 4x8 (a) matrix for motion vector of current luminance block (b) matrix for motion vector of top / bottom luminance block (c) matrix for motion vector of left / right luminance block	45
Figure 3.27 OBMC matrixes for a 8x8 macroblock. Each small luminance block size is 4x4 (a) matrix for motion vector of current luminance block (b) matrix for motion vector of top / bottom luminance block (c) matrix for motion vector of left / right luminance block	45
Figure 3.28 OBMC matrixes for a 4x4 macroblock. Each small luminance block size is 2x2 (a) matrix for motion vector of current luminance block (b) matrix for motion vector of top / bottom luminance block (c) matrix for motion vector of left / right luminance block	46
Figure 3.29 a) NDS result at 8x8 macroblock size for 76 th frame of Football b) The result of the OBMC algorithm	48
Figure 4.1 Nested vector spaces spanned by the scaling functions	55
Figure 4.2 Scaling function and wavelet vector spaces	57
Figure 4.3 Functions of a) Haar scaling b) Haar wavelet c) Daubechies(4) Scaling d) Daubechies(4) wavelet e) Biorthogonal (9.7) scaling f) Biorthogonal (9.7) wavelet	60
Figure 4.4 Two-stage two-band analysis tree	61
Figure 4.5 Two-stage two-band synthesis tree	62
Figure 5.1 Transfer function of a uniform 8-level midriser quantizer	65
Figure 5.2 20-step Lloyd - Max quantizer. 10 steps on the positive side are shown, negative side is symmetric	69

Figure 5.3 The refining algorithm for 3 stages of Lloyd-Max quantizer (a) Refine 1 interval $[x_7, x_{10})$ and take x_9 as the refining threshold in the first stage (b) Take x_5 and x_8 as the refining thresholds and refine the intervals $[x_3, x_7)$ and $[x_7, x_9)$ in the second stage (c) Take x_2, x_4 and x_6 as the refining thresholds and refine the intervals $[x_1, x_3)$, $[x_3, x_5)$ and $[x_5, x_7)$ in the thirs stage	71
Figure 5.4 Voronoi regions	73
Figure 5.5 Block diagram of simple vector quantizer	75
Figure 5.6 Structure Tree	78
Figure 5.7 Structure Map	78
Figure 6.1 Block diagram of the proposed video encoder	80
Figure 6.2 The encoder results of NDS, OBMC and error reconstruction by using OCIF Akiyo sequence at the rate of 5 fr/s and 10 kb/s (a) Reconstructed 42 nd frame with MSE = 16.1 (b) Original 48 th frame (c) Reconstructed 48 th frame by NDS with MSE = 30.22 (d) Reconstructed 48 th frame by OBMC after the NDS with MSE = 24.43 (e) Reconstructed 48 th error frame after OBMC (f) Reconstructed 48 th error frame after error coding with MSE=14.6	83
Figure 6.3 DWT layer structures for a QCIF video frame	85
Figure 6.4 The first layer of the DWT decomposition of a Carphone video frame	86
Figure 6.5 Histogram for three levels DWT of a QCIF video frame	88
Figure 6.6 Deadzone and Lloyd - Max quantization locations in a wavelet transform layer. Horizontal scale is the wavelet coefficient amplitudes, vertical scale is the number of wavelet coefficients	89
Figure 6.7 Lloyd-Max illustration of a video frame for two levels DWT	90
Figure 6.8 Parent-child relationship for wavelet coefficients	92
Figure 6.9 Reorganization of a wavelet tree into a ZeroTree wavelet block	93
Figure 6.10 The HFSVQ block sizes and coding tree that are used by the encoder	95

	Page
Figure 6.11 Intra coding comparison between H.263 and ZTW-HFSVQ	98
Figure 6.12 a) Luminance PSNR results for Akiyo at the rate of 5 fr/s and 10 kb/s b) Chrominance PSNR results for Akiyo at the rate of 5 fr/s and 10 kb/s	99
Figure 6.13 PSNR luminance results for Carphone at the rate of 10 f/s and 30 kb/s	101
Figure 6.14 Visual comparison of 72 nd luminance Carphone frame a) original frame b) H.263 result with PSNR=31.8 dB and c) ZTW - HFSVQ result with PSNR= 33.2 dB	101
Figure 6.15 Luminance PSNR results for Coastguard at the rate of 10 f/s and 48 kb/s	102
Figure 6.16 Visual comparison of 41 st luminance Coastguard frame a) original frame b) MPEG – 4 / VM result with PSNR = 29.5 dB and c) ZTW-HFSVQ result with PSNR=31.1 dB	103

CHAPTER ONE

INTRODUCTION

1.1 Motivation of the Project

The number of multimedia applications are increasing rapidly with the growth of the internet and the recent advances in both hardware and software development. Digital video is the fundamental media type that has wide range of applications such as video conferencing, digital video broadcasting, videophones, video on demand and multimedia products. The digital representation of video offers some advantages over the analog video, including : 1) variable-rate transmission, 2) easy software conversion amongst the standards, 3) editing capabilities such as cut, paste and zooming, 4) an open architecture where video may exists at different temporal, spatial and quality resolutions, 5) robustness to channel noise 6) interactivity.

The efficient digital representation of image and video signals is an important operation in video processing. The raw data rate required to transmit an HDTV (High definition TV) color video signal, at a resolution of 1920 pixels x 1080 lines at 8 bpp for each color at the rate of 30 frames/sec is 1.5 Gbits/sec (Gigabits per second). This is a huge channel capacity requirement. On the other hand the memory requirements for 75 minutes of uncompressed digital video at the NTSC (National Television Standards Committee) resolution of 720 pixels x 480 lines, at 8 bpp for each color and at a rate of 30 fr/s is approximately 140 GB(GigaBytes). This enormous memory requirement exceeds the capacities of single hard disk drives that has a storage of 1 to 40 GB typically, as well as CD-ROM technologies where the CD's are capable of holding only 650 MB, and also more recent DVD (Digital Versatile Disk) drives that hold approximately 5 GB. Therefore, video compression is needed for achieving the necessary bitrate reductions for both storage and transmission.

1.2 Overview of the Video Compression Standards

The Moving Picture Experts Group (MPEG) was formed to develop appropriate video coding standards in 1988. MPEG-1 was the first standard completed in 1992 (Gall D. J., 1992). It compresses the CIF video and audio data at bit rates up to 1.5 Mb/s.

The MPEG-2 was developed as an extension to MPEG-1. The MPEG-2 was standardized in 1994 (Chiariglione L., 1995). The performance of MPEG-2 is better than the MPEG-1 at bit rates up to 3 Mb/s.

The H.261 and H.263 are the block based DCT video compression standards (Rao & Hwang, 1996) like MPEG-1 and MPEG-2. They compress the QCIF video and audio data at lower bit and frame rates than MPEG. In addition to this the H.263 standard uses Overlapped Block Motion Compensation technique (Auyeung C. et al., 1992) in order to reduce the block artifacts.

The MPEG-4 Verification Model (MPEG-4/VM) is a block based DCT video coding standard that was developed in 1996 (Sikora T., 1997). The MPEG-4/VM separates the objects and the background in a QCIF video frame. Then the encoder processes each object using block based DCT. The bit rates for MPEG-4/VM are 5-64 kb/s for low bit rate videotelephone standard and up to 5 Mb/s for TV/film applications.

Beside these standards mentioned above, the video compression subject has been studied by several authors (Al-Shaykh O.K. et al., 1999) (Chang, P.C. & Lu, T.T., 1999) (Kim, B.J., 2000). The EZW and ZTE techniques (Martucci S.A. et al., 1997) (Yin, C.Y., 2000) are two of the developed DCT based video encoders which have comparable results to the MPEG-4/VM.

1.3 Research Objectives

In this thesis the search for improving the video compression techniques is addressed. The main concern is to design a video compression technique with comparable low bit rates. In order to achieve this purpose first of all, the performances of the popular block motion estimation algorithms are compared. These algorithms are Two Dimensional Logarithmic Search (TDL), One at a Time Search (OTS), Three Step Search(3SS), New Three Step Search (N3SS), Improved Three Step Search (I3SS), Novel Four Step Search (N4SS), Simple and Efficient Search (SES), New Diamond Search (NDS). This work which is the part of this thesis is published in (Kılıç, İ. & Yılmaz, R., 2001). The simulation results show that the New Diamond Search performs better than the compared algorithms when the Search Point Number and Mean Square Error criterion are considered both. Because of that as a block motion estimation algorithm, the New Diamond Search is selected to track the local motion. The motion vectors of the macroblocks are Huffman encoded. The OBMC technique which is an advance scheme in the H.263 standard is used after NDS to reduce the block artifacts.

In order to remove the spatial correlation a three level of discrete wavelet transform (DWT) is used instead of DCT. The encoder uses different filter bank at each level of the decomposition. Since the longer filters provide good frequency localization and shorter filters cause less ringing artifacts, the Biorthogonal 9.3 filter bank is used at the first level of the DWT decomposition followed by the Haar filter for the remaining two levels. It is seen that the DWT coefficients are Gaussian shape distributed in each level except the lowest frequency subband. Therefore the proposed encoder uses a linear quantizer for the lowest frequency subband and the nonlinear Lloyd-Max quantizer (Lloyd, S.P., 1982) for the rest of the subbands.

The proposed algorithm is similar to the ZTE scheme introduced in (Martucci, S.A., 1997). The main difference is combining the zerotree wavelet structure with HFSVQ instead of arithmetic coding. It is important to note that, the zerotree structure consists of different size of wavelet blocks that contain similar

informations. Because of that it is very suitable for these blocks to be coded by HFSVQ algorithm. ZTW structure itself has the ability to control the bit rate. When the HFSVQ is applied to any ZTW structure, then each type of wavelet block will be represented by different size of codeword. That will give us the second opportunity of controlling the bit rate.

The encoder is computer simulated by using the Object Oriented Borland C++ Builder software. The simulation results of the proposed encoder was taken using three types of standard video sequences; QCIF Akiyo which contains a few movements at the rate of 10 kb/s and 5 fr/s, QCIF Carphone which contains relatively more movements than Akiyo at the rate of 30 kb/s and 10 fr/s and QCIF Coastguard which contains more movements than the others at the rate of 48 kb/s and 7.5 fr/s.

1.4 An Outline of the Thesis

In Chapter 2, some of the digital video characteristics are presented. These are the RGB, YUV, YIQ and YCbCr color coordinate systems, and video source formats, Common Intermediate Format (CIF) and source input format (SIF) (Rao, K.R. & Hwang, J.J., 1996). The performance measure criterions used in this thesis are also described in this chapter.

The popular block motion estimation algorithms are described in detail in Chapter 3. These block motion estimation algorithms are Full Search (FS), Two Dimensional Logarithmic Search (TDL), One at a Time Search (OTS), Three Step Search (3SS) (Rao, K.R. & Hwang, J.J., 1996), An Improved Three Step Search (I3SS) (Xu, D. & Bailey C., 1998), A New Three Step Search (N3SS) (Li R. et al., 1994), A Novel Four Step Search (N4SS) (Po, L.M. & Ma, W.C., 1996), Simple and Efficient Search (SES) (Lu, J. & Liou, M.L., 1997) and New Diamond Search (NDS) (Zhu, S. & Ma, K.K., 2000). The performance comparison of all those search algorithms according to the average search point number and mean square error criterion are done at different macroblock sizes using 100 frames of Football and Miss America video sequence.

The overlapping block motion compensation (OBMC) (Auyeung et al., 1992) technique which is an advance scheme in the H.263 standard is described and its performance results are also presented in this chapter. The standard OBMC filters are 8x8 pixel size for the 16x16 macroblocks. Since the performance comparison of the block motion estimation algorithms mentioned above are done using different size of macroblocks then additional new OBMC filters are designed for 8x8, 16x8, 8x16, 4x4 macroblock sizes. The OBMC filters are used after the block motion algorithm to reduce the block artifacts.

The Chapter 4 describes some useful features of the discrete wavelet transform. These are the definition of discrete wavelet transform, wavelet and scaling function expansion, multiresolution analysis and matrix representations. The scaling and the wavelet function coefficients of the Daubechies 4, Daubechies 6, Biorthogonal 9.7, Biorthogonal 9.3 and the Haar wavelet (Rao, R.M. & Bopardikar, A.S., 1998) are presented. The graphical representation of the Biorthogonal 9.3 and the Haar scaling and wavelet functions, which are used by the proposed encoder are also given in this chapter.

In Chapter 5, the quantization types used by the proposed encoder are defined. These are scalar quantization, Lloyd-Max quantization (Yin, C.Y., 2000), HFSVQ (Yu, P. & Venetsanopoulos, A. N., 1994). Scalar quantization is used at the lowest subband of the DWT. Since the other high frequency subbands of the DWT layers contains wavelet coefficient distributions like Gaussian shape, then Lloyd-Max quantization is selected as the quantizer in those subbands. HFSVQ is used together with the zerotree structure to encode the wavelet coefficients.

The proposed video encoder called ZTW-HFSVQ is described in Chapter 6. The proposed encoder consists of NDS, Huffman coding, OBMC, multiresolution of DWT, ZTW structure, Lloyd-Max quantizer, HFSVQ and rate control structure. In this chapter, the combination of ZTW structure with the HFSVQ, which is the novel feature of this new introduced encoder is described in details. The performance results of the proposed encoder are also given in this chapter using three types of

standard video sequences; QCIF Akiyo which contains a few movements at the rate of 10 kb/s and 5 fr/s, QCIF Carphone which contains relatively more movements than Akiyo at the rate of 30 kb/s and 10 fr/s and QCIF Coastguard which contains more movements than the others at the rate of 48 kb/s and 7.5 fr/s.

The Chapter 7 consists of a brief conclusion.

The performance comparison of the popular block motion estimation algorithms which is the first part of this thesis has been presented at the SIU 2001 conference. The new video compression algorithm called ZTW-HFSVQ which is introduced in this thesis is accepted as a paper to be presented at ISPA 2003 (3rd International Symposium on Image and Signal Processing and Analysis) on September 18-20, 2003 in ITALY.



CHAPTER TWO

DIGITAL VIDEO CHARACTERISTICS

2.1 RGB Color Coordinate

The RGB (red, green and blue) color space is the basic choice for computer graphics and digital video processing. Because color cathode ray tubes use red, green and blue phosphors to create the desired color. Red, green and blue are the three primary additive colors. Individual components are added together to form a specific color. The equivalent addition of all components produces white shown in Figure 2.1

However, RGB is not very efficient for representing real world images, since equal bandwidths are required to describe all three color components. The equal bandwidths results in the same pixel depth and display resolution for each color component. The human eye is more sensitive to luminance component than chrominance. Therefore many image and video coding standards use luminance and color difference signals. These are YUV, YIQ and YCbCr color formats

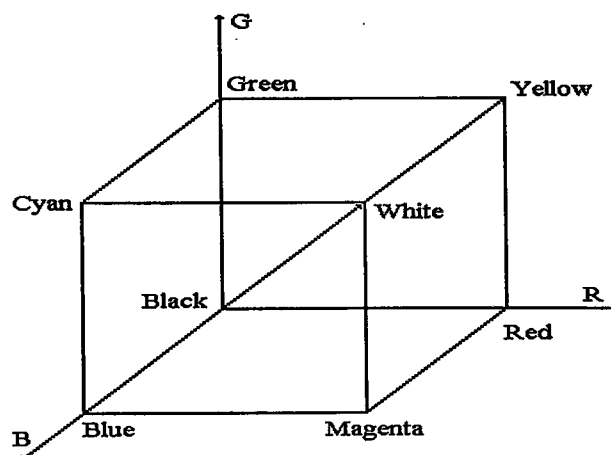


Figure 2.1 The RGB color cube.

2.2 YUV Color Coordinate

The YUV color coordinate is the basic color format used by the NTSC, PAL and SECAM composite color TV standards. Y represents the black-and-white component and color information (U and V) is added to display a color picture. YUV signals are derived by using the RGB signals as

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ U &= -0.147R - 0.289G + 0.436B = 0.492(B - Y) \\ V &= 0.615R - 0.515G - 0.100B = 0.877(R - Y) \end{aligned} \quad (2.1)$$

In a digital television system the YUV signal is obtained by an RGB-to-YUV matrix. Usually color difference signals (U and V) are subsampled by a factor of two to four in the spatial dimensions, because these are much less sensitive to the human visual system than the luminance Y. This is helpful in reducing the bit rate in video compression techniques.

2.3 YIQ Color Coordinate

The YIQ color coordinate is derived from the YUV format and is optionally used by the NTSC composite TV standard. The I stands for in-phase and Q for quadrature-phase, which are the modulation methods separated by a phase angle of 90 degrees. The basic equations are

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ I &= 0.596R - 0.275G - 0.321B = 0.736(R - Y) - 0.268(B - Y) \\ Q &= 0.212R - 0.523G + 0.311B = 0.478(R - Y) + 0.413(B - Y) \end{aligned} \quad (2.2)$$

The color signals I and Q can also be obtained by phase shifting values of U and V such as,

$$\begin{aligned}
 I &= -U\sin(33^\circ) + V\cos(33^\circ) \\
 Q &= U\cos(33^\circ) + V\sin(33^\circ)
 \end{aligned}
 \tag{2.3}$$

2.4 YCbCr Color Coordinate

The YCbCr color coordinate was developed as part of ITU-R BT.601 during the establishment of a worldwide digital video component standard. The YCbCr signals are scaled and offset versions of the YUV format. Y is defined to have a nominal range of 16 to 235; Cb and Cr are defined to have a range of 16 to 240, with zero signal corresponding to level 128.

There are several YCbCr sampling formats, such as 4:4:4, 4:2:2 and 4:1:1 (4:2:0). The basic features of the worldwide standard for digital television studios, ITU-R BT.601, are shown in Table 2.1.

Table 2.1 Digital video formats specified by ITU-R 601, JPEG(J), H.261(H) and MPEG-1(M) (I: interlaced, P: progressive)

Image Format	Resolution	YCbCr	Scan Type	Mbyte/s	Use
ITU-R 601 (NTSC)	720x480	4:2:2	I	20.7	J
ITU-R 601 (PAL)	720x576	4:2:2	I	20.7	J
Square pixel (NTSC)	640x480	4:2:2	I	18.4	J
Square pixel (PAL)	768x576	4:2:2	I	22.1	J
ITU-R SIF (NTSC)	352x240	4:2:0	P	3.8	M, J
ITU-R SIF (PAL)	352x288	4:2:0	P	3.8	M, J
Square pixel SIF (NTSC)	320x240	4:2:0	P	3.5	M, J
Square pixel SIF (PAL)	384x288	4:2:0	P	4.1	M, J
CIF (NTSC & PAL)	352x288	4:2:0	P	4.6	H
QCIF (NTSC & PAL)	176x144	4:2:0	P	1.1	H

The sampling format 4:2:2 implies that the sampling rates of Cb and Cr are one-half that of Y. Also there are alternate Y samples only sandwiched between cosited

Y, Cb and Cr samples. In a consecutive run of four samples there are 4 Y samples and 2 samples each of Cb and Cr. This sampling pattern is signified by 4:2:2.

The positioning of YCbCr samples or pixels for the 4:4:4, 4:2:2, 4:1:1 and 4:2:0 formats are illustrated in Figure 2.2. Each component is typically quantized with 8 bit resolution. Each sample therefore requires 24 bits for the 4:4:4 format and 16 bits for the 4:2:2 format. Four samples require 6 times 8 bits for the 4:2:0 format.

In the 4:2:0 format, the Cb and Cr samples are offset. MPEG-1/2 and H.261 video standards use this format (Table 1).

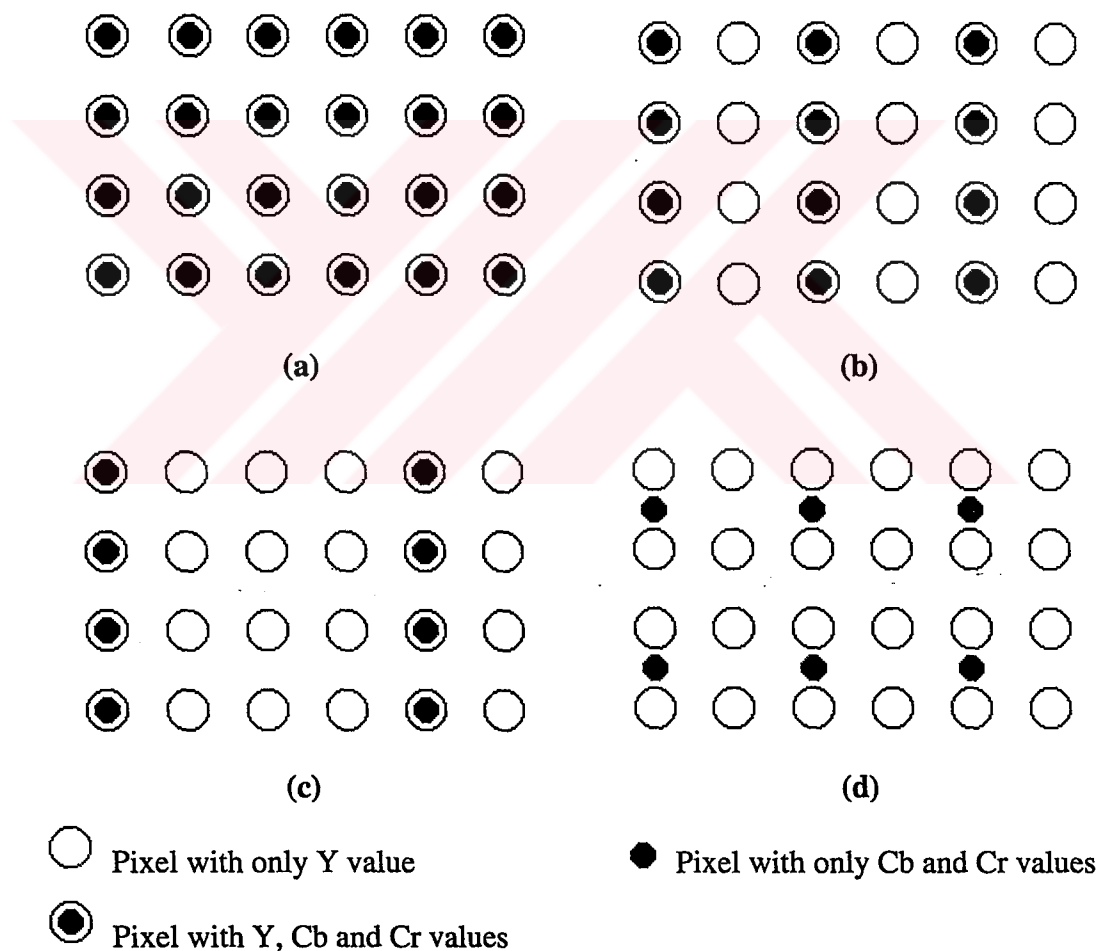


Figure 2.2 Positioning of YCbCr pixels for a) 4:4:4 b) 4:2:2 c) 4:1:1 d) 4:2:0

2.5 Common Intermediate Format (CIF)

The video format for video communication services has to be based on existing standards: NTSC, PAL and SECAM. On the other hand visual telephony, which usually has slow moving objects (head and shoulders configuration), does not, in general, need a large bandwidth. Also, the viewer can tolerate the visual quality, which can be less than that of the TV broadcasting standard. Moreover, resolution reduction is required for a wide range of applications. Therefore, spatial resolution was reduced by CCITT SCXV Specialist Group and the format was called Common Intermediate Format (CIF) and Quarter CIF (QCIF). The information exchange via the common intermediate format shown in Figure 2.3. The resolutions for the two formats (CIF and QCIF) are shown in Figure 2.4.

For luminance component Y, 360 pixels per line were defined, just a 2:1 decimation from the 720 active pixels per line used in TV systems (ITU-R 601). The number of CIF frame line were defined half of the PAL/SECAM system which has 576 active lines (twice the CIF resolution). CIF frame rate of 29.97 is based on the NTSC system.

The processing of frames in H.261, H.263, MPEG-1/2, MPEG4-VM(Verification Model Version) video standards is block based. Because motion estimation is based on 16x16 luminance Y blocks, vertical stripes of 4 pixels on either side of the Y-component, are cropped out, resulting in 352 pixels/line and 288 lines/frame resolution. This is called the significant pixel area (SPA). The frame resolution of the SPA is now integer divisible by the 16x16 block. The corresponding SPA for the luminance CIF/QCIF frames and the chrominance CIF/QCIF frames are shown in Figure 2.4. In Table 2.2, CIF/QCIF and the two systems (NTSC and PAL) are compared. Note that parameters of the two system are based on ITU-R 601.

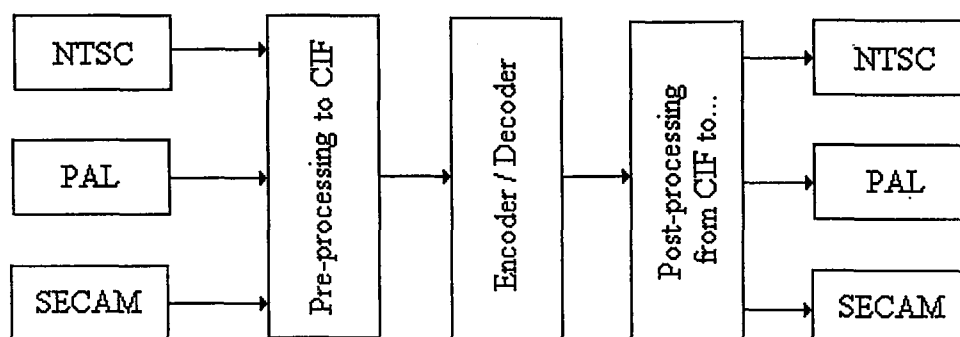


Figure 2.3 Information exchange via the common intermediate format (CIF)

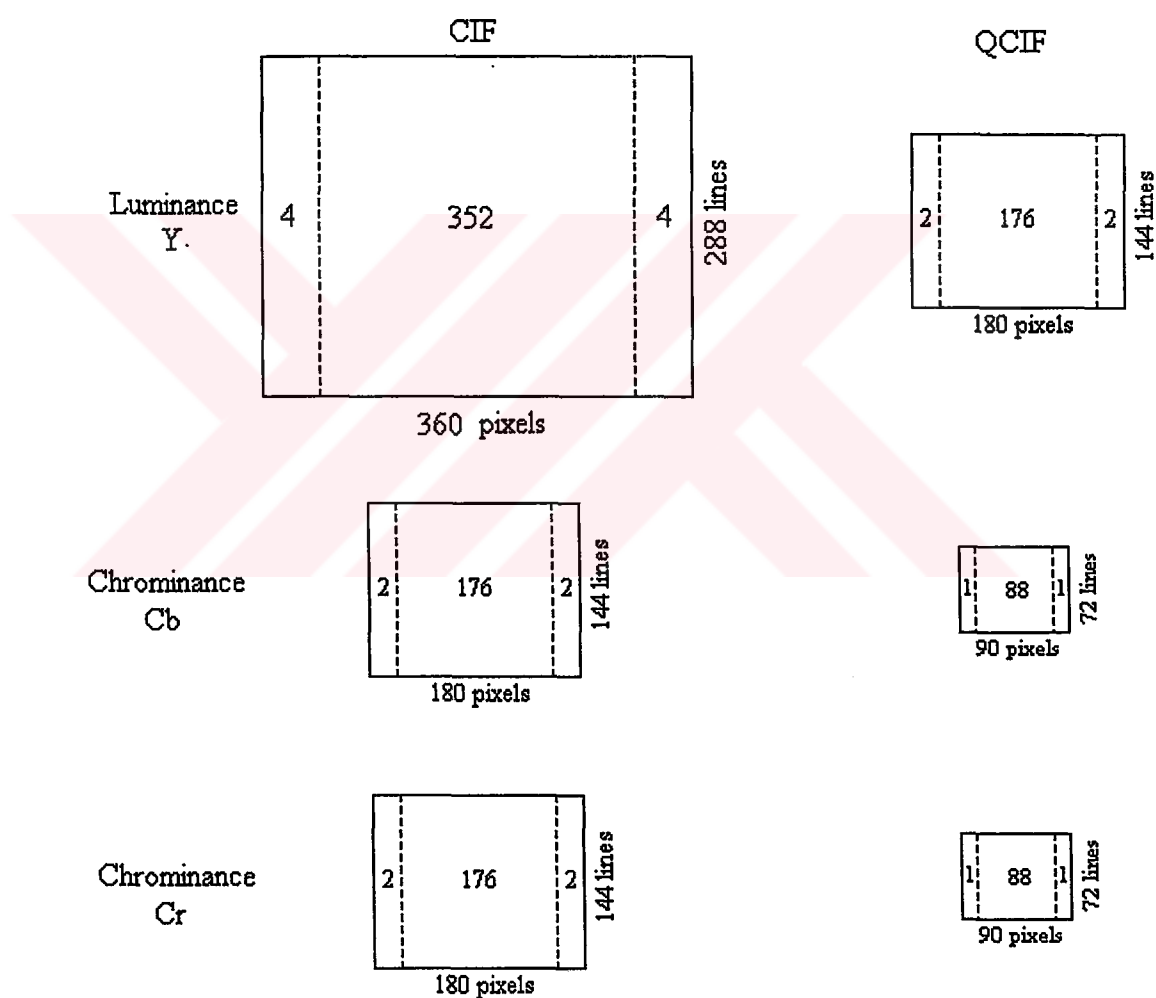


Figure 2.4 CIF and QCIF resolutions

Table 2.2 Parameters of CIF and TV systems

Parameters	Y / Cb, Cr	CIF	QCIF	NTSC (525/30)	PAL (625/25)
Active pixels/line	Y	360	180	720	720
Active pixels/line	Cb, Cr	180	90	360	360
Active lines/picture	Y	288	144	480	576
Active lines/picture	Cb, Cr	144	72	480	576
Pictures / Second	-	29.97	29.97	29.97	25

2.6 Source Input Format (SIF)

The video source information may exist with various formats in different resolutions. The source input format is generally used by MPEG 1-2. After the conversion for NTSC and PAL standards, SIF has two optimum resolutions; first one SIF-525, which has 360x240 pixels at a 30 Hz frame rate, the second one is SIF-625, which has 360x288 pixels at 25 Hz. The different SIF resolutions and the SPA values are shown in Table 2.3.

Table 2.3. Conversion of two source formats to SIF

Parameters	Format	NTSC(525/30)	PAL(625/25)
Luminance (Y)	ITU-R	720x480	720x576
	SIF	360x240	360x288
	SPA	352x240	352x288
Chrominance (Cb, Cr)	ITU-R	360x480	360x576
	SIF	180x120	180x144
	SPA	176x120	176x144
Frame Rate (Hz)	-	30	25

Since the human eyes are more sensitive to luminance frame than chrominance frame, the SIF chrominance frames (Cb, Cr) are subsampled by 2:1 in both vertical and horizontal directions. In order to change the video source resolution the decimation and interpolation filters are used. Decimation filters are used for the

conversion of PAL, SECAM, NTSC into SIF, whereas interpolation filters are used for the reversion (postprocessing) in the decoder as shown in Figure 2.5 and Figure 2.6.

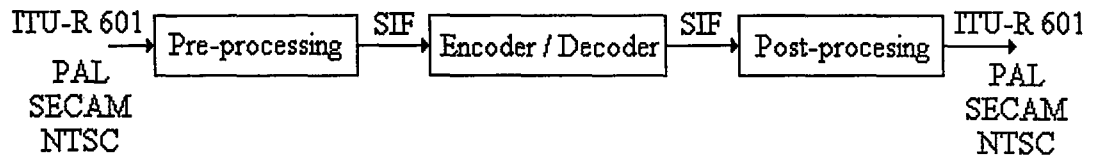


Figure 2.5 Pre and post-processing at the encoder and decoder

-29	0	88	138	88	0	-29	/ 256
-----	---	----	-----	----	---	-----	-------

Luminance Decimation Filter

-12	0	140	256	140	0	-12	/ 256
-----	---	-----	-----	-----	---	-----	-------

Luminance Interpolation Filter

1	3	3	1	/ 8
---	---	---	---	-----

Chrominance Decimation Filter

1	3	3	1	/ 8
---	---	---	---	-----

Chrominance Interpolation Filter

Figure 2.6 Decimation and interpolation filters for conversion between ITU-R 601 and SIF

2.7 Performance Measures

For image or video data, a pixel is the basic element. Therefore bits per sample is also referred to as bits per pixel(bpp). In the literature, the term compression ratio, denoted as C_r , is also used instead of bit rate to explain the capability of the compression system. The definition of C_r is,

$$C_r = \frac{\text{Source coder input size}}{\text{Source coder output size}} \quad (2.4)$$

specific compression method that is used. For a still image, size could refer to the bits needed to represent the entire image. For a video, size could refer to the bits needed to represent one frame of video. Many compression algorithms for video, however, do not process each frame of video the same way. Therefore more commonly used notation for size is the bits needed to represent one second of video.

For n -bit frames, or equivalently, 2^n gray scale frames, peak signal to noise ratio (PSNR) in decibels (dB) is defined as,

$$PSNR = \log_{10} \frac{(2^n - 1)^2}{MSE} \quad (2.5)$$

$$MSE = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N [X(i, j) - \hat{X}(i, j)]^2 \quad (2.6)$$

Here, M and N are the width and the height of the frame respectively, X and \hat{X} are the original frame and the reconstructed frame respectively.

CHAPTER THREE

BLOCK MOTION ESTIMATION AND COMPENSATION

3.1 Introduction

Motion estimation has played an important role in video compression. One popular technique for motion estimation, as recommended in both MPEG and H.261 standards, is the block matching algorithm. The main idea of block matching algorithm is to divide frames into blocks and then find the best matching block in the previous frame for each block in the current frame for a given matching criterion. Block matching is the most time consuming part of the encoding process. During block matching, each target block of the current frame is compared with a past frame in order to find a matching block. The most of the motion estimation algorithms takes a block size of 16×16 pixels and maximum search range of ± 7 pixels in both horizontal and vertical directions. MSE and PSNR are generally good matching criteria formulations for the block matching algorithms. The block motion estimation approach is shown in Figure 3.1. In this figure, the block that belongs to current video frame is searched for in the previous video frame in a search window. The best match is represented by a motion vector (mv).

There are many block motion estimation algorithms in the literature. Some of them are Full Search (FS), Two Dimensional Logarithmic Search (TDL), One at a Time Search (OTS), Three Step Search(3SS), New Three Step Search (N3SS), Improved Three Step Search (I3SS), Novel Four Step Search (N4SS), Simple and Efficient Search (SES) and New Diamond Search (NDS). These block motion estimation algorithms are studied and to derive the performance comparison, all

these algorithms are computer simulated. This is made for different macroblock sizes by using standard Football and Miss America in the following sections.

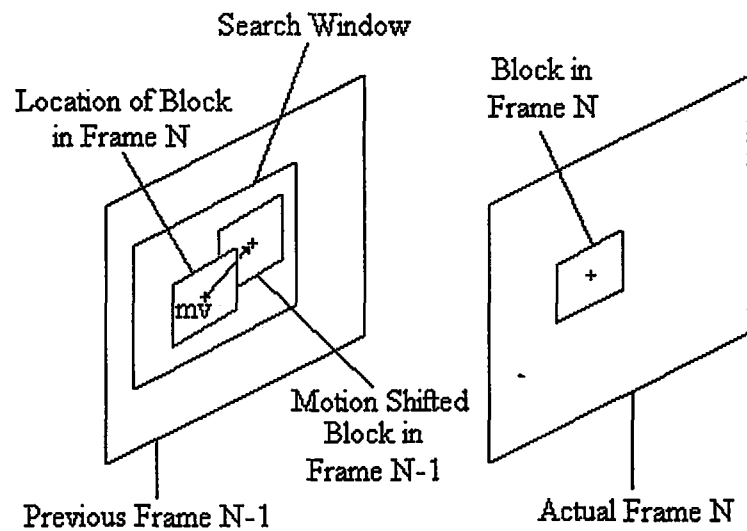


Figure 3.1 The block motion estimation structure.

3.2 Full Search (FS)

The Full Search algorithm(FS) searches all possible displacements within the search range. Although the full search algorithm gives the optimal result by searching through all possible blocks in the searching range, its high computational cost limits its practical use. Therefore all other block motion estimation algorithms were developed to decrease the search point number of the FS at reasonable error rates.

3.3 Two Dimensional Logarithmic Search (TDL)

The Two Dimensional Logarithmic Search (TDL) (Rao & Hwang, 1996) is the first simplified block motion estimation algorithm in the literature which has been used since 1981. The TDL search tracks the motion along the direction of minimum distortion. The algorithm begins with the four search points at X, -X, Y, -Y directions with "s" pixels distance to the centre of the original block. TDL has several stages.

Stage 1: The block at the centre of the search area and the four candidate blocks at a distance “s” from the centre of the x and y axes are compared to the target block to determine which is the best match. Thus if the centre of the search area is at position [0,0] then the candidate blocks at positions [0,0], [0,s], [0,-s], [s,0] and [-s,0] are examined.

Stage 2 : If the position of the best match is at the centre [cx, cy] then the step size s is halved. If the best match is in one of the four outer positions, then that position becomes the centre point of the next stage. That is [cx,cy]=[a,b] where [a,b] is the position of the best match.

Stage 3: If the step size s is equal to one then all nine blocks around the centre position are examined and the best match of these is determined to be the best match for the target block. That is [cx-1,cy-1], [cx-1,cy], [cx-1,cy+1], [cx,cy-1], [cx,cy], [cx,cy+1], [cx+1,cy-1], [cx+1,cy] and [cx+1,cy+1] are examined, the algorithm halts. Otherwise (step size greater than one) the candidate blocks at positions [cx,cy], [cx+s,cy], [cx-s,cy], [cx,cy+s] and [cx,cy-s] are compared. The algorithm goes to stage 2. An example of TDL search path is given in Figure 3.2.

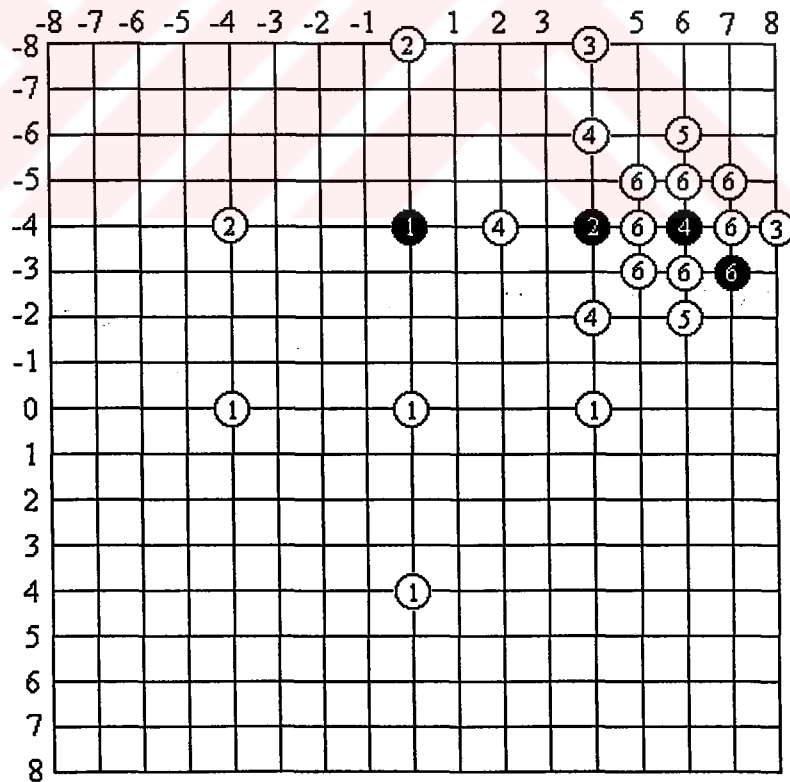


Figure 3.2 An example of TDL algorithm

3.4 One at a Time Search (OTS)

One at a Time Search motion estimation algorithm was developed in 1994 (Rao & Hwang, 1996). This algorithm follows the least MSE point along the first X, and then Y direction.

In the first step the OTS searches the points along the X axis of the search area and finds the least distortion point. In the second step OTS searches the minimum distorted point in the Y direction along the search area. These two steps repeat one more time but the searched point number is the half of the search area size. Then the minimum MSE point gives us the motion vector direction. This algorithm is shown in Figure 3.3.

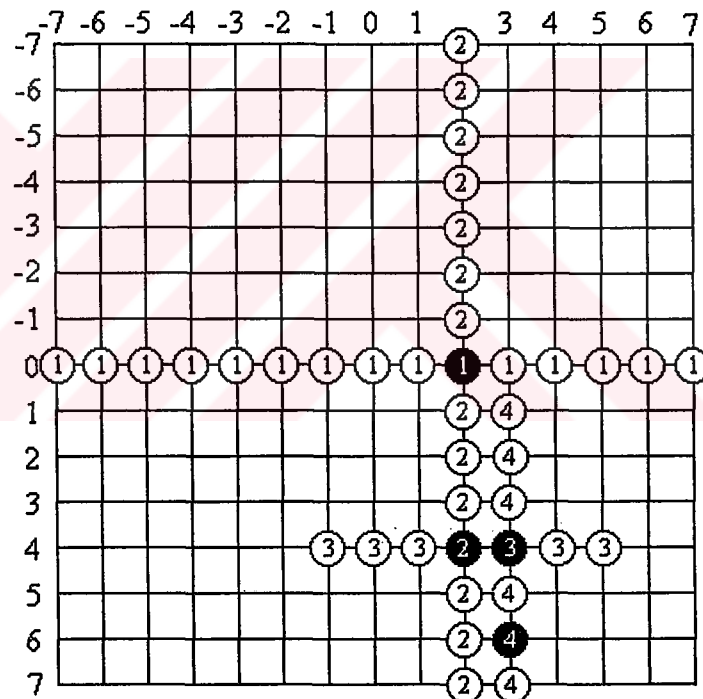


Figure 3.3 An example of One at a Time Search

3.5 Three Step Search (3SS)

This algorithm was introduced in 1981. The 3SS algorithm has a fine-coarse search mechanism that is seen in Figure 3.4.

The first step(circle 1), the search is based on 4 pixels / 4 lines resolution at the nine locations, with the center point corresponding to zero motion vector. The second step , the search is based on 2 pixels / 2 lines resolution around the location determined by the first step (Rao & Hwang, 1996). This is repeated in the third step(circle 3) with 1 pixel / 1 line resolution. The last step yields the motion vector. The motion vector search range is fixed 7 pixels at the directions of X, -X, Y, -Y. Therefore, the number of search point number is 25.

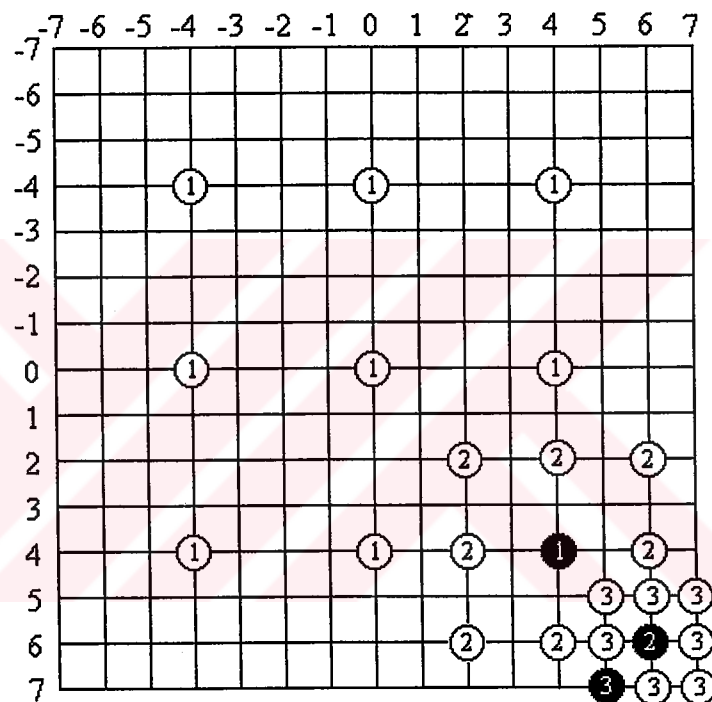


Figure 3.4 An example of Three Step Search Algorithm

3.6 An Improved Three Step Search (I3SS)

This algorithm was designed in 1998 (Xu et al., 1998). I3SS is the improved version of the well-known 3SS method. The I3SS algorithm is specifically aiming towards low bit-rate video coding applications. I3SS needs less search points and has much better performance and faster speed than the 3SS algorithm. The three search steps of I3SS can be summarized as follows:

Step 1: The minimum MSE point is found from a nine checking points pattern on a 5x5 window located at the centre of the 15x15 searching area as shown in Figure 3.5(a). If the minimum MSE point is found to coincide with the centre of the search window, then go to step 3 else go to step 2.

Step 2: The search window size is maintained at 5x5 with a search pattern chosen after considering the two alternatives:

a) If the previous minimum MSE point is located at one of the corners of the previous search window, then five additional checking points are considered. An example is shown in Figure 3.5(b) where black circles and white circles represent additional and previously evaluated pixels respectively.

b) If the previous minimum MSE point is located at the middle of any horizontal or vertical edge of the previous search window, then three additional checking points are considered. An example is shown in Figure 3.5(c) where black circles and white circles represent additional and previously evaluated pixels respectively.

Step 3: The search window is reduced to 3x3 around the minimum MSE point in step 2 as shown in Figure 3.5(d) and the direction of the overall motion vector is taken to be the minimum MSE point among these nine search points, of which eight are new.

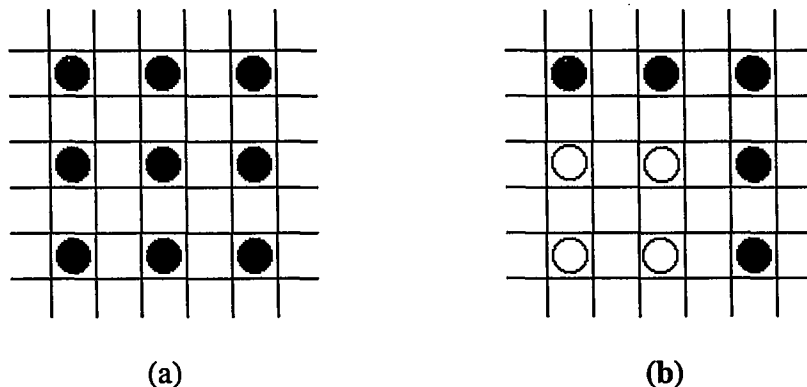


Figure 3.5. Search patterns of the I3SS. a) First step centered on centre pixel b) Second step centred on a corner pixel

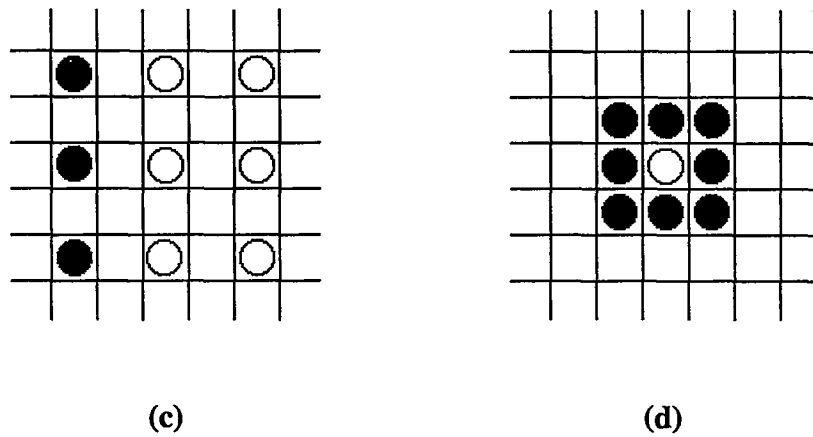


Figure 3.5. Search patterns of the I3SS. a) First step centered on centre pixel b) Second step centred on a corner pixel c) Second step centred on a middle pixel d) Third step

Since there are some overlapping check points on the 5x5 search window in the second step of I3SS, the total number of checking points will vary from a minimum of $(9+8)=17$, when step 2 can be skipped, to $(9+5+8)=22$ in the worst case. Two examples of I3SS search paths are shown in Figure 3.6.

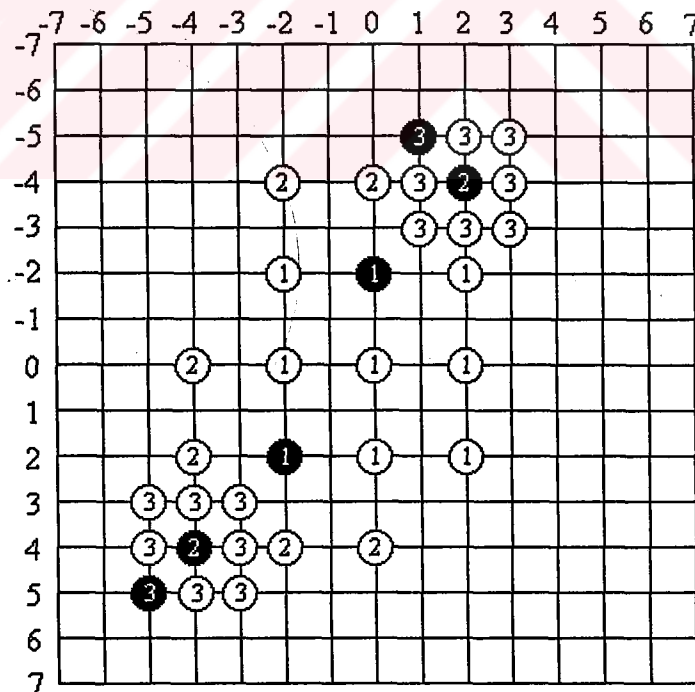


Figure 3.6 Two examples of I3SS search paths.

3.7 A New Three Step Search (N3SS)

This algorithm was designed in 1994 (Li et al., 1994). N3SS differs from 3SS by (1) assuming a center-biased checking point pattern in its first step and (2) incorporating a halfway-stop technique for stationary or quasi-stationary blocks. The steps of N3SS are given below;

Step 1: In the first step, in addition to the original checking points used in 3SS, eight extra points are added, which are the eight neighbors of the search window center, as shown in Figure 3.8. (As such, the checking point pattern is highly center-biased)

Step 2: A halfway-stop technique is used for stationary and quasi-stationary block in order to fast identify and then estimate the motions for these blocks:

- a) If the minimum MSE in the first step occurs at the search window center, stop the search. (This is called the first-step-stop)
- b) If the minimum MSE point in the first step is one of the eight neighbors of the window center, the search in the second step will be performed only for eight neighboring points of the minimum and then stop the search. (This is called the second-step-stop)

Although N3SS uses more checking points in its first step as compared to 3SS, the first-step-stop and second-step-stop can reduce computation significantly. For example, eight block matches will be saved once a first-step-stop occurs. Depending on the position of the minimum MSE point (in the first step) on the 8 neighbors of the window center, five or three block matches will be saved once a second-step-stop occurs: (1) if the minimum is one of the four neighboring positions along the horizontal or vertical directions, five block matches will be saved; (2) if the minimum is one of the four neighbors along the two diagonal directions, three block

matches will be saved. The block diagram and three examples of N3SS search paths are shown in Figure 3.7 and Figure 3.8.

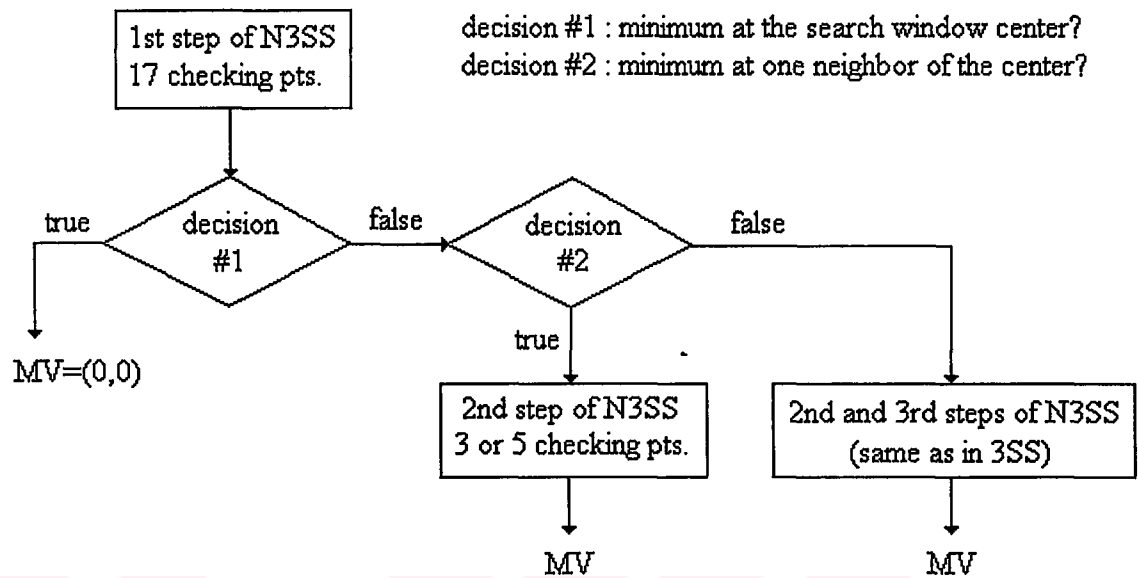


Figure 3.7 The block diagram of the N3SS algorithm

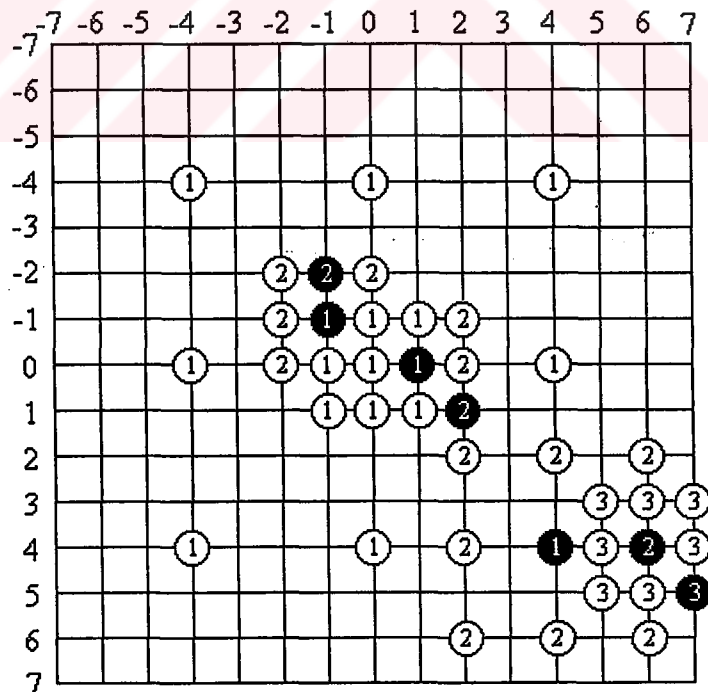


Figure 3.8 Three examples of N3SS search paths

3.8 A Novel Four-Step Search (N4SS)

This algorithm was designed in 1996 (Po & Ma, 1996). For the maximum motion displacements of ± 7 , the N4SS algorithm utilizes a center-biased search pattern with nine checking points on a 5×5 window in the first step instead of a 9×9 window in the 3SS. The center of the search window is then shifted to the point with minimum MSE. The search window size of the next two steps depended on the location of the minimum MSE points. If the minimum MSE point is found at the center of the search window, the search will go to final step with 3×3 search window. Otherwise, the search window size is maintained in 5×5 for step 2 or step 3. In the final step, the search window is reduced to 3×3 and the search stops at this small window. The N4SS algorithm is summarized as follows:

Step 1: A minimum MSE point is found from a nine checking points pattern on a 5×5 window located at the center of the 15×15 searching area as shown in Figure 3.9(a). If the minimum MSE point is found at the center of the search window, go to step 4; otherwise go to step 2.

Step 2: The search window size is maintained in 5×5 . However, the search pattern will depend on the position of the previous minimum MSE point.

a) If the previous minimum MSE point is located at the corner of the previous search window, five additional checking points as shown in Figure 3.9(b) are used.

b) If the previous minimum MSE point is located at the middle of horizontal or vertical axis of the previous search window, three additional checking points as shown in Figure 3.9(c) are used. If the minimum MSE point is found at the center of the search window, go to step 4; otherwise go to step 3.

Step 3: The searching pattern strategy is the same as step 2, but finally it will go to step 4.

Step 4: The search window is reduced to 3x3 as shown in Figure 3.9(d) and the direction of the overall motion vector is considered as the minimum MSE point among these nine searching points.

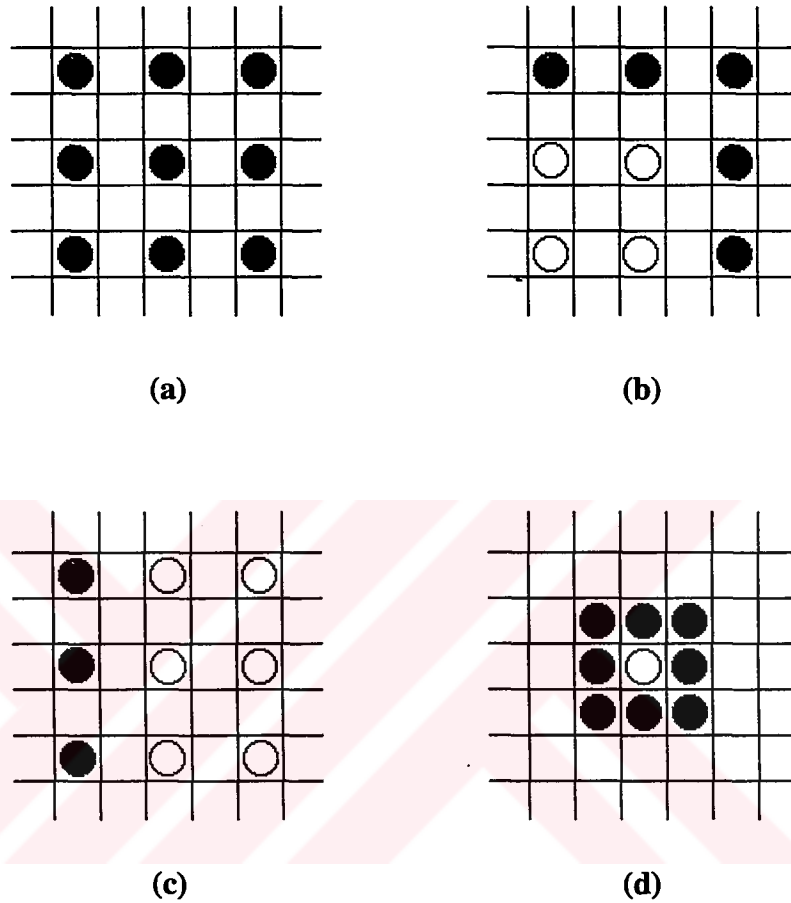


Figure 3.9 Search patterns of the N4SS. a) First step b) second/third step c) second/third step d) fourth step

Since there are overlapped checking points on the 5x5 search windows in the step 2 and step 3, then the total number of checking points is varied from $(9+8)=17$ to $(9+5+5+8)=27$. The worst case computational requirement of the N4SS is 27 block matches which happens for the estimation of large movement. Two examples of N4SS are shown in Figure 3.10 with different search paths. For the upper search path, a total of 25 checking points are used with the estimated motion vector equal to $(3, -7)$. For the worst case example of the lower search path, the number of checking points required is 27 and the estimated motion vector is equal to $(-7, 7)$. We can find

that the computational complexity of 4SS is only two block matches more than the 3SS while six block matches less than the N3SS in the worst case.

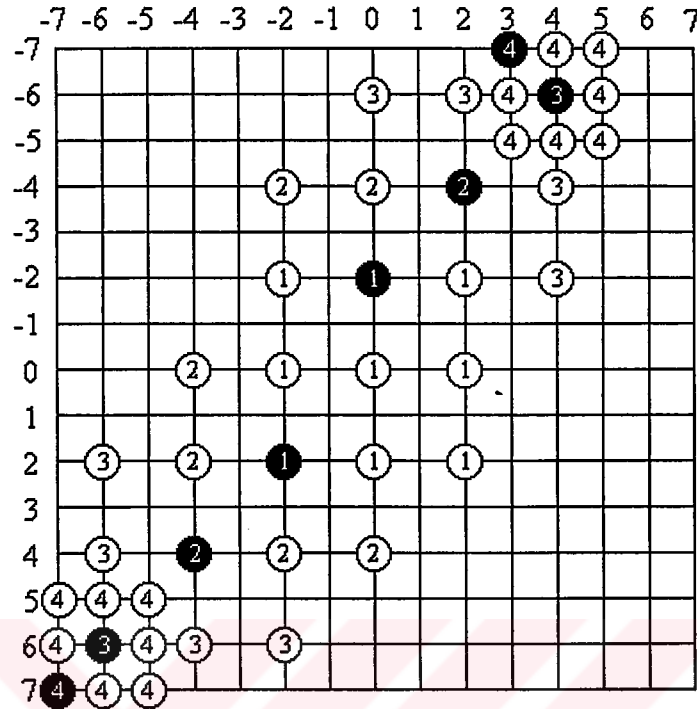


Figure 3.10 Two different search paths of N4SS

3.9 Simple and Efficient Search (SES)

This algorithm was designed in 1997 (Lu & Liou, 1997). The SES algorithm reduces the search point number of 3SS algorithm. All steps of SES begin with selecting a search quadrant in the search area. Then finds the minimum error location in the selected quadrant. The rule for determining a search quadrant on X-Y axes can be described as follows;

- If $MSE(A) \geq MSE(B)$ and $MSE(A) \geq MSE(C)$, I is selected;
- If $MSE(A) \geq MSE(B)$ and $MSE(A) < MSE(C)$, II is selected;
- If $MSE(A) < MSE(B)$ and $MSE(A) < MSE(C)$, III is selected;
- If $MSE(A) < MSE(B)$ and $MSE(A) \geq MSE(C)$, IV is selected.

The search points and selected quadrants are shown in Figure 3.11. The search patterns corresponding to each selected quadrant are shown in Figure 3.12.

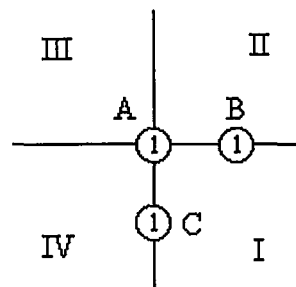


Figure 3.11 Search pattern in the first step of SES

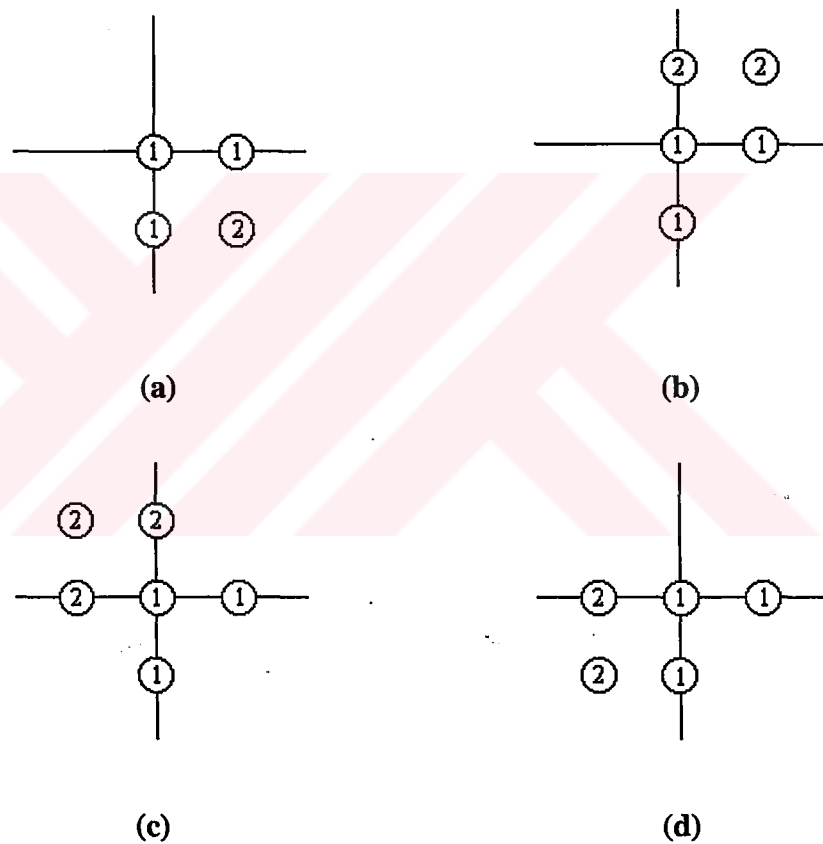


Figure 3.12 Search pattern corresponding to each selected quadrant a) quadrant I, b) quadrant II, c) quadrant III, d) quadrant IV (numbers in the circles show the first and the second phases)

The second and third steps follow the same procedure as the first step described above. An example of SES algorithm is shown in Figure 3.13.

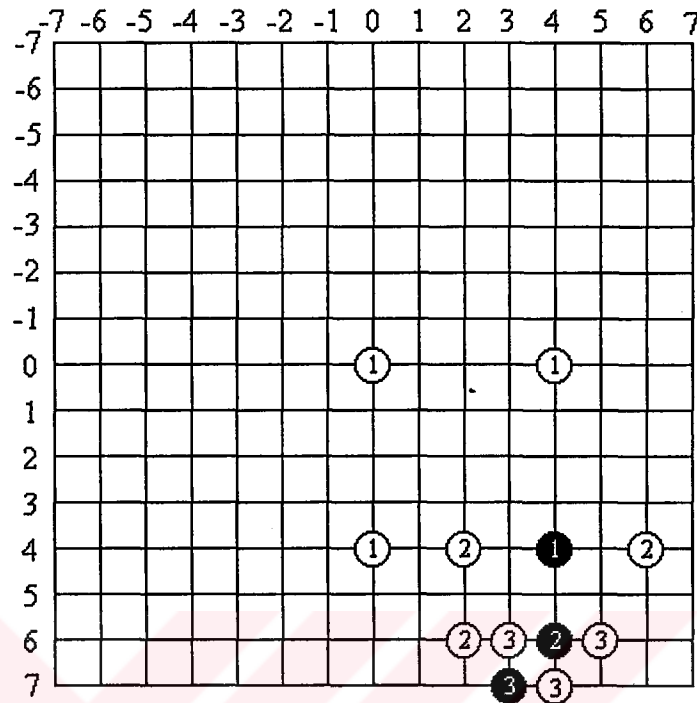


Figure 3.13 An example for the SES algorithm

It is observed that the number of checking points for the SES is five on average in the first step and four on average in the subsequent steps (excluding the location which is checked in the previous step). Therefore, the total number of checking points for each block-matching is further reduced compared with the 3SS.

3.10 New Diamond Search (NDS)

This algorithm was designed in February 2000 (Zhu & Ma, 2000). The NDS algorithm has two search patterns as illustrated in Figure 3.14. The first pattern, called large diamond search pattern (LDSP), consists of nine checking points from which eight points surround the center one to compose a diamond shape. The second pattern consists of five checking points that form a smaller diamond shape, called small diamond search pattern (SDSP).

In the searching procedure of the NDS algorithm, LDSP is repeatedly used until the step in which the minimum block distortion occurs at the center point. The search pattern is then switched from LDSP to SDSP as reaching to the final search stage. Among the five checking points in SDSP, the position yielding the minimum MSE provides the motion vector of the best matching block.

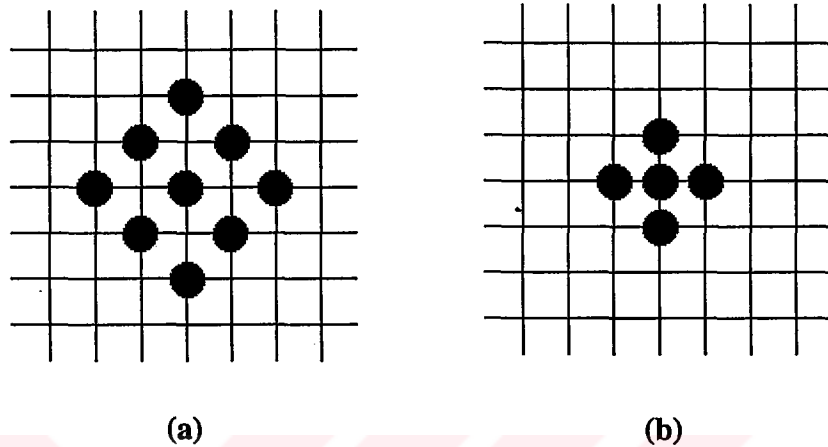


Figure 3.14 Two search patterns of the NDS a) LDSP b) SDSP

The NDS algorithm is summarized as follows.

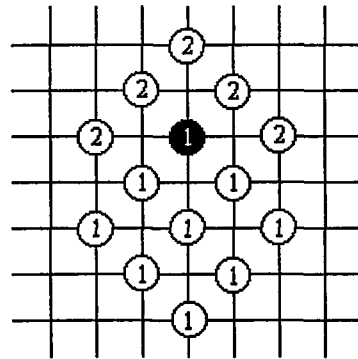
Step 1: The initial LDSP is centered at the origin of the search window, and the 9 checking points of LDSP are tested. If the MSE point calculated is located at the center position, go to step 3; otherwise, go to step 2.

Step 2: The MSE point found in the previous search step is re-positioned as the center point to form a new LDSP. If the new MSE point obtained is located at the center position, go to step 3; otherwise, repeat this step.

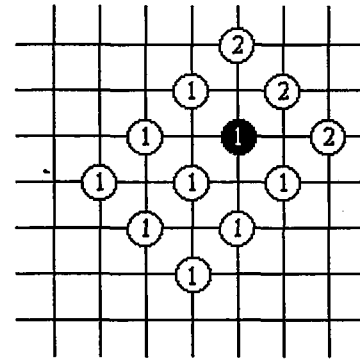
Step 3: Switch the search pattern from LDSP to SDSP. The MSE point found in this step is the final solution of the motion vector which points to the best matching block.

For illustration, three cases of check-point overlapping are presented in Figure 3.15. When the previous MSE point is located at one of the corners or edge points of LDSP, only five or three new check points are required to be tested as shown in

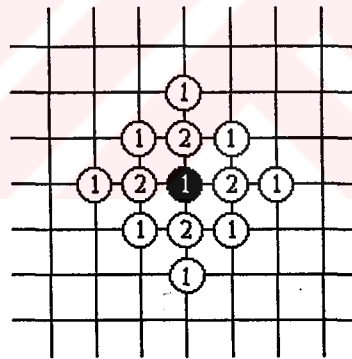
Figure 3.15(a) and Figure 3.15(b), respectively. If the center point of LDSP produces the MSE, the search pattern is changed from LDSP to SDSP in the final search. In this case, only four new points are required to be tested, as shown in Figure 3.15(c). An example of NDS is shown in Figure 3.16.



(a) Case 1: the corner point)
(LDSP → LDSP)



(b) Case 2: the edge point
(LDSP → LDSP)



(c) Case 3: the center point (LDSP → SDSP)

Figure 3.15 Three cases of checking-point overlapping in LDSP when the MSE point found in the previous search step (number 1) is located at a) one of the corner points, b) one of the edge points, and c) the center point. The points of number 2 are the the new checking points.

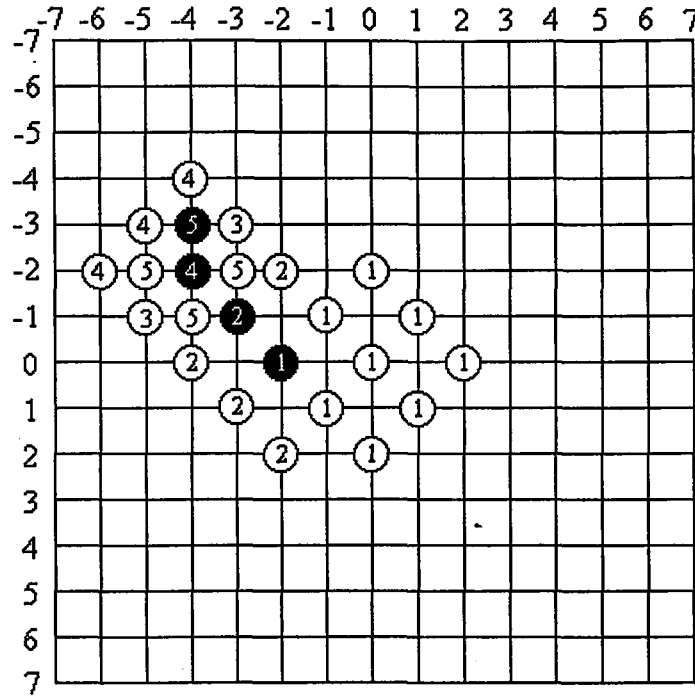


Figure 3.16 A search path example of NDS

3.11 Computer Simulation of Block Motion Estimation Algorithms and the Performance Comparison

The performance of the popular block motion estimation algorithms called FS, TDL, OTS, 3SS, I3SS, N3SS, N4SS, NDS and SES are compared by using standard 100 SIF frames of “Football” and 100 CIF frames of “Miss America” video sequences. The Mean Square Error (MSE) and the Search Point Number (SPN) criterion are used for the performance comparison. All block motion estimation algorithms are simulated using different size of macroblocks which are 4x2, 4x4, 8x2, 4x8, 8x4, 16x2, 8x8, 4x16, 16x8, 8x16, 16x16, 32x16. The MSE value between the current block(m,n) and the previous block($m+x,n+y$) is defined in Equation 3.1.

$$MSE_{(m,n)(x,y)} = \frac{1}{(a \times b)} \sum_{i=1}^a \sum_{j=1}^b (f_k(m+i, n+j) - f_{k-1}(m+x+i, n+y+j))^2 \quad (3.1)$$

Here, the $f_k(i, j)$ and $f_{k-1}(i, j)$ are the pixel values of k^{th} and $(k-1)^{\text{th}}$ frames at (i, j) respectively. The values a and b are the dimensions of a macroblock.

In Table 3.1 and Table 3.2, the average MSE and SPN values are shown for the Football sequence respectively. In Table 3.3 and Table 3.4, the average MSE and SPN values are shown for the Miss America sequence respectively.

It is seen from Table 3.2 and Table 3.4 that changing the macroblock size does not affect so much the SPN order from best to worst. The SPN order from best to worst is SES, TDL, NDS, I3SS, N4SS, N3SS, 3SS, OTS and FS for the Football sequence. On the other hand, for the Miss America sequence the SPN order from best to worst is SES, TDL, I3SS, NDS, N4SS, 3SS, N3SS, OTS and FS. It is clearly seen that SES, TDL and NDS algorithms are faster than the rest of the block motion estimation algorithms.

Table 3.1, 3.3 and Figure 3.17, 3.18 show that for the small macroblock sizes the OTS and N3SS better than the other algorithms according to the average MSE value. On the other hand, for the bigger macroblock sizes such as 16×16 and 32×16 the NDS block motion estimation is the best one.

Table 3.1 Average MSE values for 100 Football frames

	4x2	4x4	8x2	4x8	8x4	16x2	8x8	4x16	16x8	8x16	16x16	32x16
FS	56	78	82	102	102	117	126	137	161	161	198	255
3SS	97	124	128	151	151	163	173	182	200	201	230	274
OTS	88	113	118	141	142	157	166	175	198	198	228	275
TDL	107	126	129	147	145	160	164	175	189	191	218	262
N3SS	82	104	108	128	127	142	150	161	181	181	213	264
I3SS	126	144	150	168	169	188	191	202	224	222	256	304
N4SS	126	144	150	169	169	188	191	202	224	222	256	304
NDS	108	126	125	147	140	154	159	170	183	185	212	258
SES	142	169	174	196	197	212	222	228	250	249	277	320

Table 3.2 Average SPN values for 100 Football frames

	4x2	4x4	8x2	4x8	8x4	16x2	8x8	4x16	16x8	8x16	16x16	32x16
FS	218	217	216	215	215	212	213	208	209	207	202	193
3SS	25	24	24	24	24	24	24	24	24	24	23	22
OTS	43	43	43	43	43	43	43	42	42	42	42	41
TDL	15	15	15	15	15	15	15	15	15	15	15	14
N3SS	23	22	22	22	22	21	21	21	21	21	21	20
I3SS	18	18	18	18	18	18	18	18	17	17	17	16
N4SS	19	19	19	19	19	18	18	18	18	18	18	17
NDS	17	17	17	17	17	17	17	17	17	17	17	16
SES	15	14	14	14	14	14	14	14	14	14	14	14

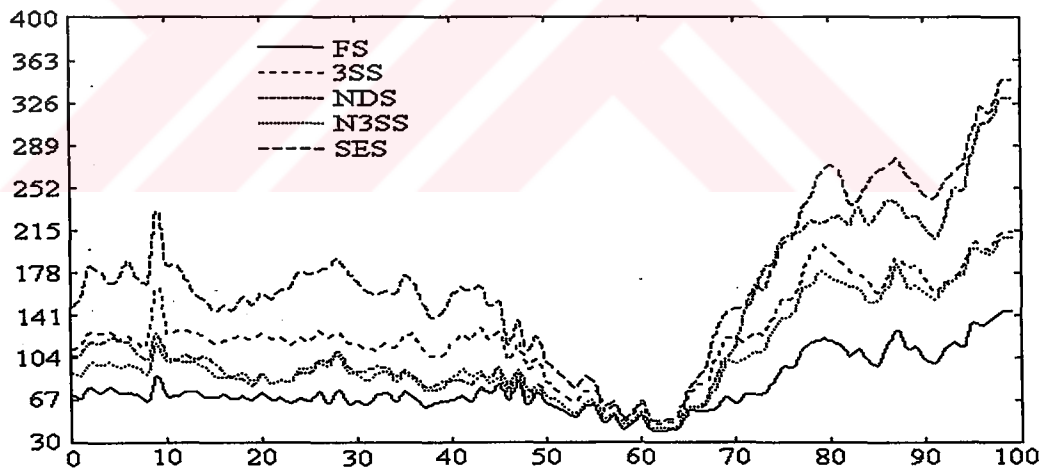
Table 3.3 Average MSE values for 100 Miss America frames

	4x2	4x4	8x2	4x8	8x4	16x2	8x8	4x16	16x8	8x16	16x16	32x16
FS	4	6	6	8	8	8	9	9	10	10	11	12
3SS	7	9	9	10	10	10	11	11	12	12	12	13
OTS	6	8	8	9	9	9	10	10	11	11	12	12
TDL	7	8	8	10	10	10	11	11	11	11	12	13
N3SS	6	7	7	9	9	9	10	10	11	11	11	12
I3SS	8	10	10	11	11	11	12	12	12	12	13	14
N4SS	8	10	10	11	11	11	12	12	12	12	13	14
NDS	7	10	8	11	10	10	11	11	12	11	12	13
SES	8	10	10	11	11	12	12	12	13	13	14	14

Table 3.4 Average SPN values for 100 Miss America frames

	4x2	4x4	8x2	4x8	8x4	16x2	8x8	4x16	16x8	8x16	16x16	32x16
FS	218	218	217	216	216	212	215	210	210	209	204	195
3SS	25	25	25	24	24	24	24	24	24	24	24	23
OTS	44	43	43	43	43	43	43	43	43	43	42	41
TDL	18	18	17	18	18	17	18	17	17	17	16	15
N3SS	27	26	26	26	26	25	25	25	24	24	23	22
I3SS	20	20	19	19	19	19	19	19	18	18	18	17
N4SS	21	21	21	20	20	20	20	20	19	19	19	18
NDS	21	21	20	22	21	20	21	20	20	20	19	18
SES	14	14	14	14	14	14	14	14	14	14	14	13

The MSE results for 4x4 macroblock size is also shown in Figure 3.17 and Figure 3.18.



(a)

Figure 3.17 The Football sequence MSE values at 4x4 macroblock size for (a) FS, 3SS, NDS, N3SS, SES (b) FS, TDL, OTS, N4SS

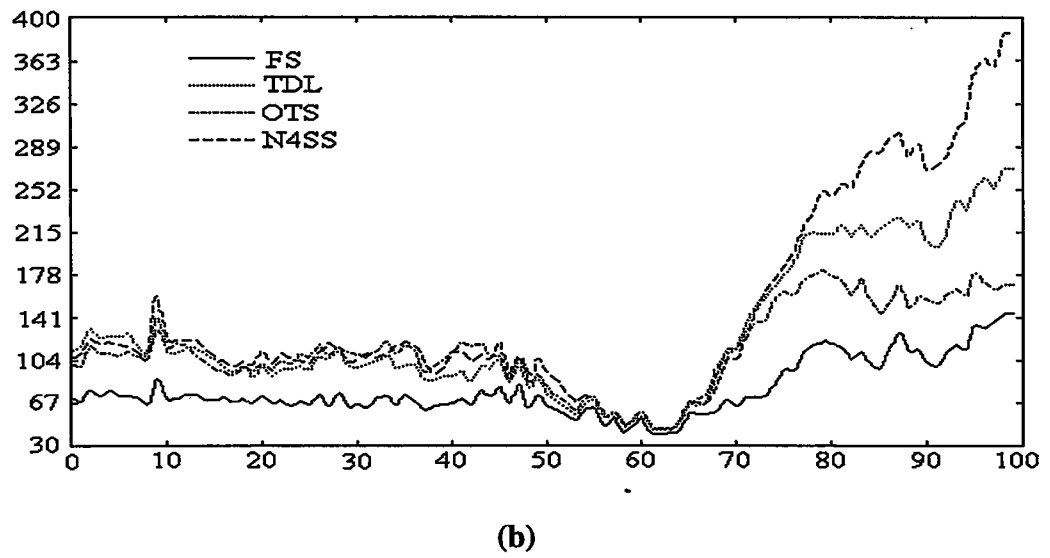


Figure 3.17 The Football sequence MSE values at 4x4 macroblock size for (a) FS, 3SS, NDS, N3SS, SES (b) FS, TDL, OTS, N4SS

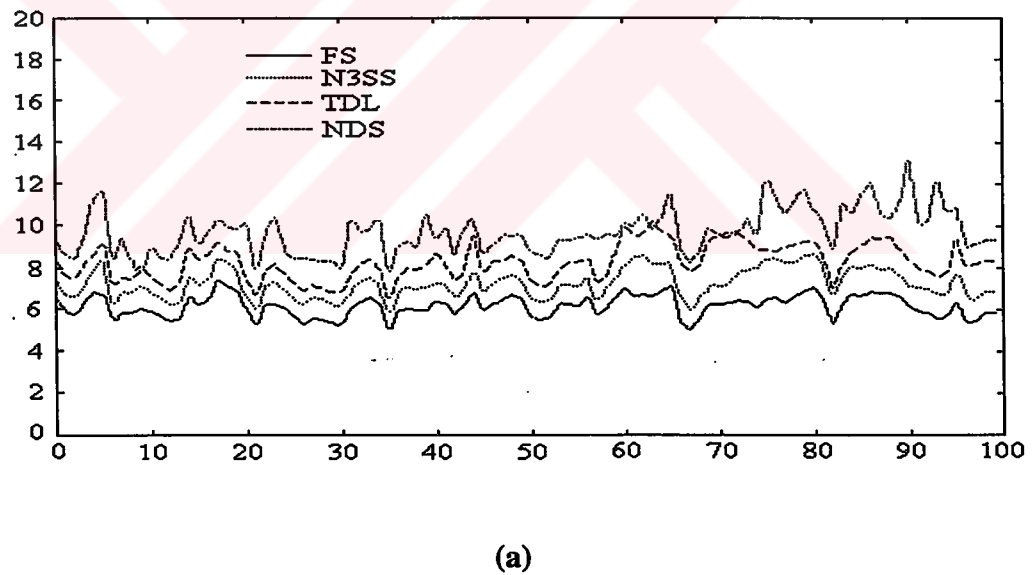


Figure 3.18 The Miss America MSE values at 4x4 macroblock size for (a) FS, N3SS, TDL, NDS (b) FS, OTS, 3SS, SES, N4SS

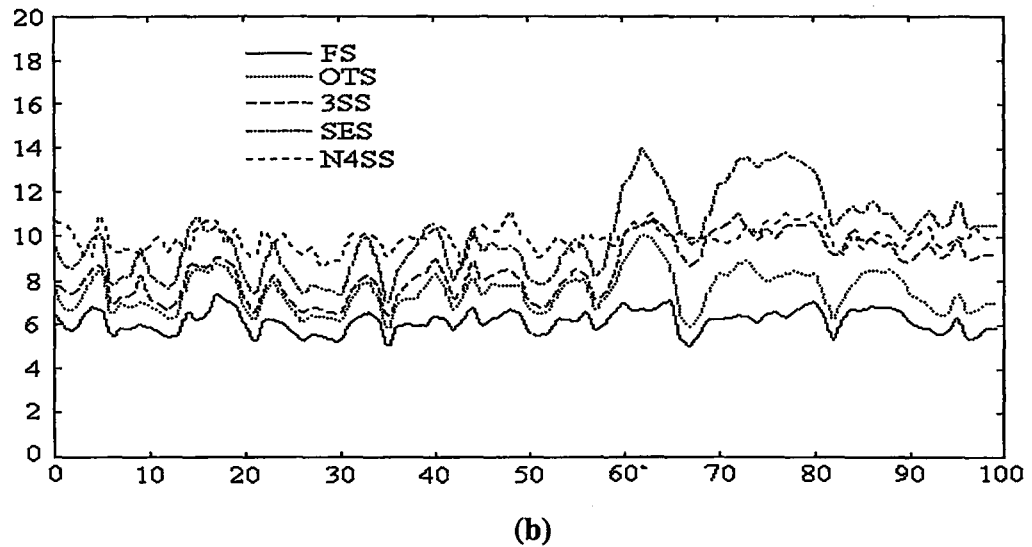


Figure 3.18 The Miss America MSE values at 4x4 macroblock size for (a) FS, N3SS, TDL, NDS (b) FS, OTS, 3SS, SES, N4SS

The average SPN results of different macroblock sizes are shown in Figure 3.19 and 3.20 for Football and Miss America sequence respectively. It is seen that the SES algorithm is faster than the other algorithms both in Football and Miss America sequence.

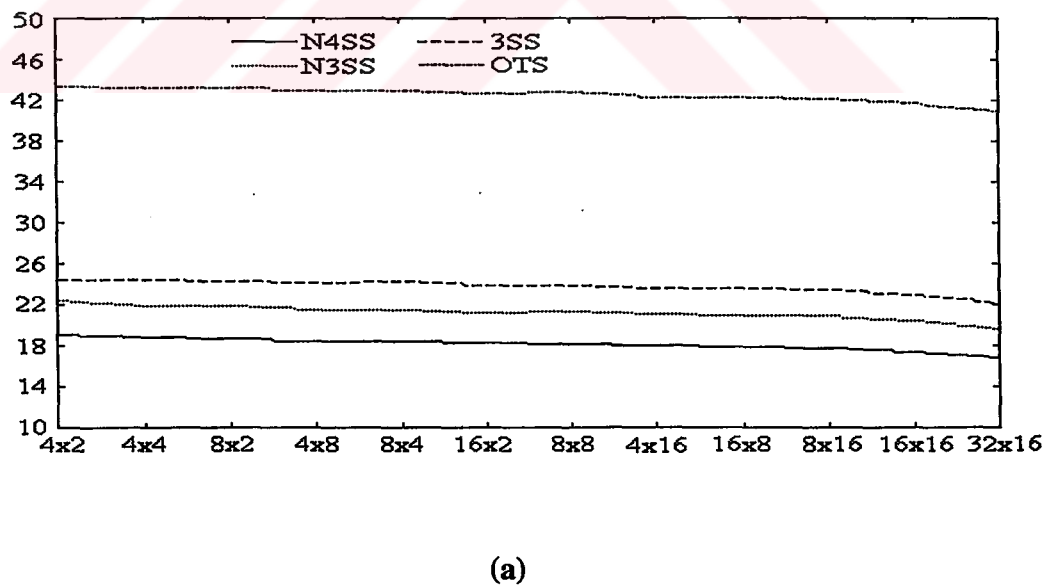


Figure 3.19 The Football sequence average SPN values at different macroblock sizes for (a) N4SS, N3SS, 3SS, OTS (b) SES, TDL, NDS, I3SS

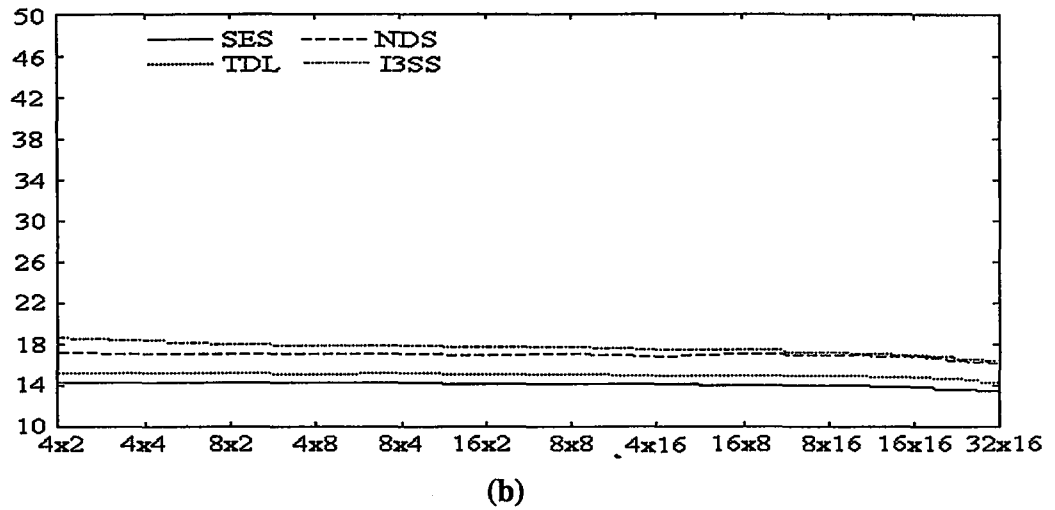
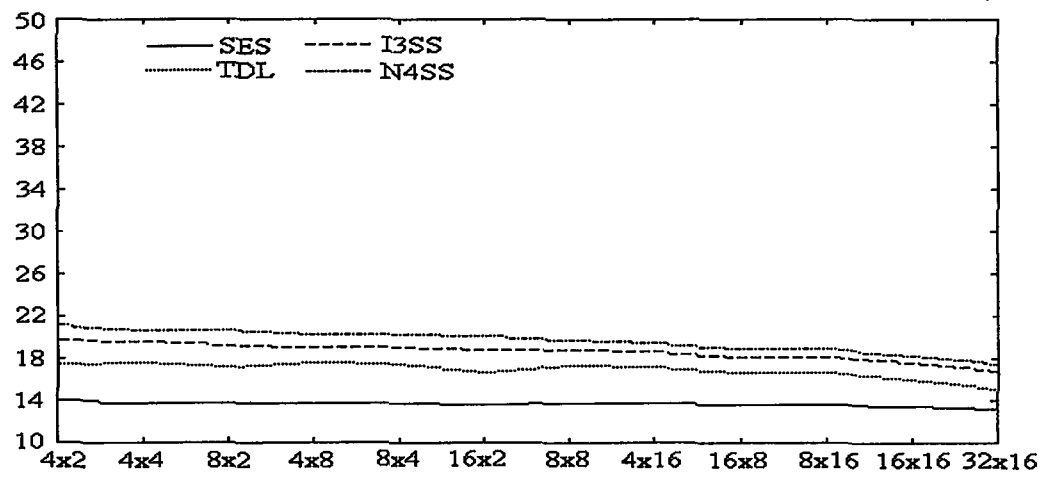


Figure 3.19 The Football sequence average SPN values at different macroblock sizes for (a) N4SS, N3SS, 3SS, OTS (b) SES, TDL, NDS, I3SS

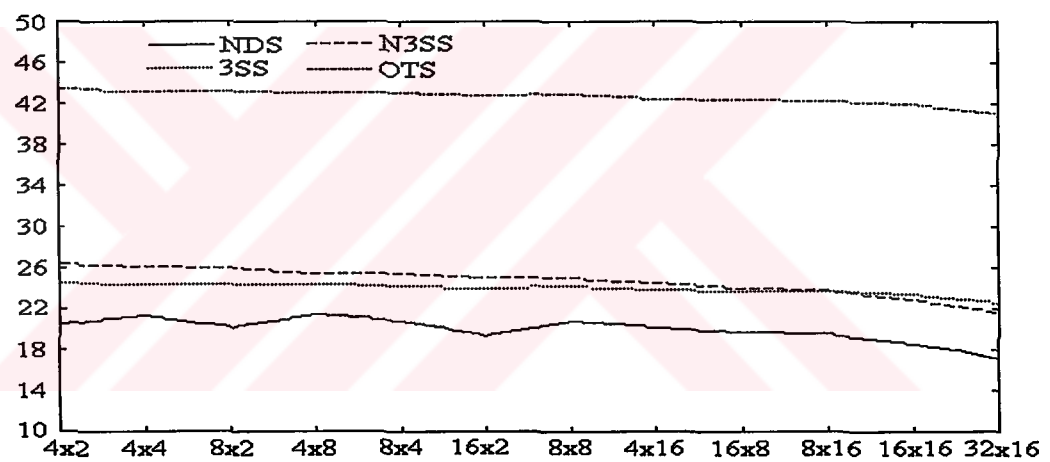
For the Football sequence and 16x16 macroblock size the SES algorithm is 14.5 times faster than the FS, 3 times faster than OTS and 1.7 times faster than 3SS. For the other algorithms, SES algorithm is faster than TDL 6.4 %, NDS 19.9 %, I3SS 21.7 %, N4SS 25 %, N3SS 46.5 %.

For the Miss America sequence and 16x16 macroblock size the SES algorithm is 15 times faster than the FS, 3.1 times faster than OTS and 1.7 times faster than 3SS. For the other algorithms, SES algorithm is faster than TDL 18.4 %, I3SS 29.6 %, N4SS 35.0 %, NDS 37.4 %, N3SS 68.7 %.

The average MSE results of different macroblock sizes are shown in Figure 3.21 and 3.22 for Football and Miss America sequence respectively.

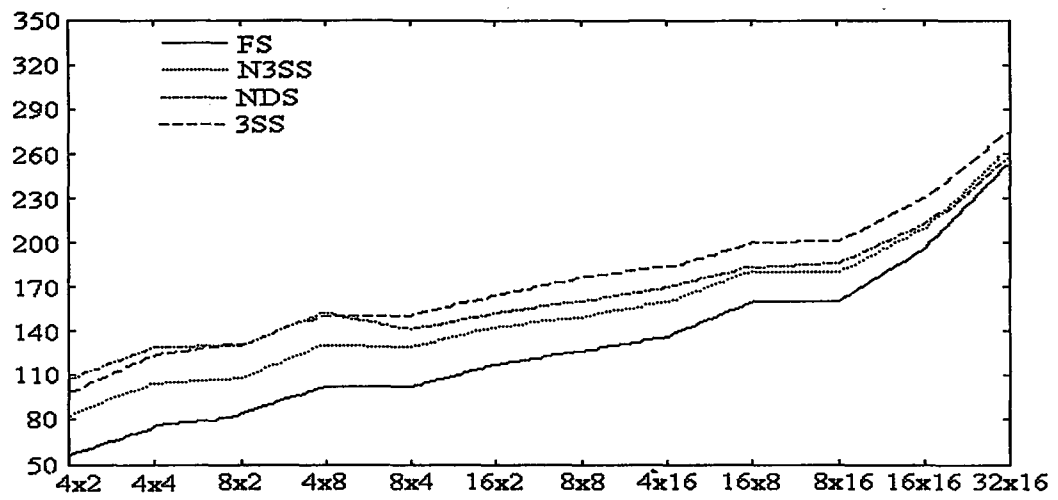


(a)

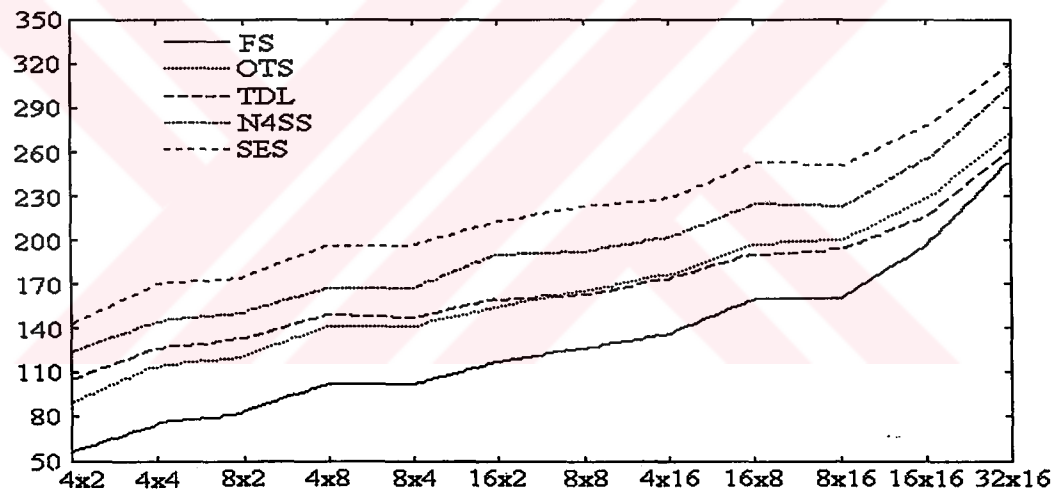


(b)

Figure 3.20 The Miss America sequence average SPN values at different macroblock sizes for (a) SES, TDL, N4SS, I3SS (b) NDS, 3SS, N3SS, OTS

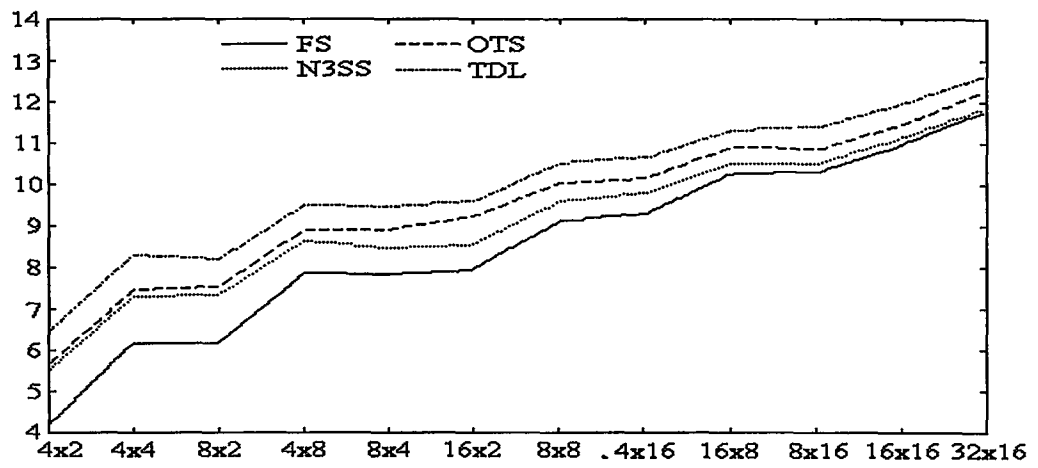


(a)

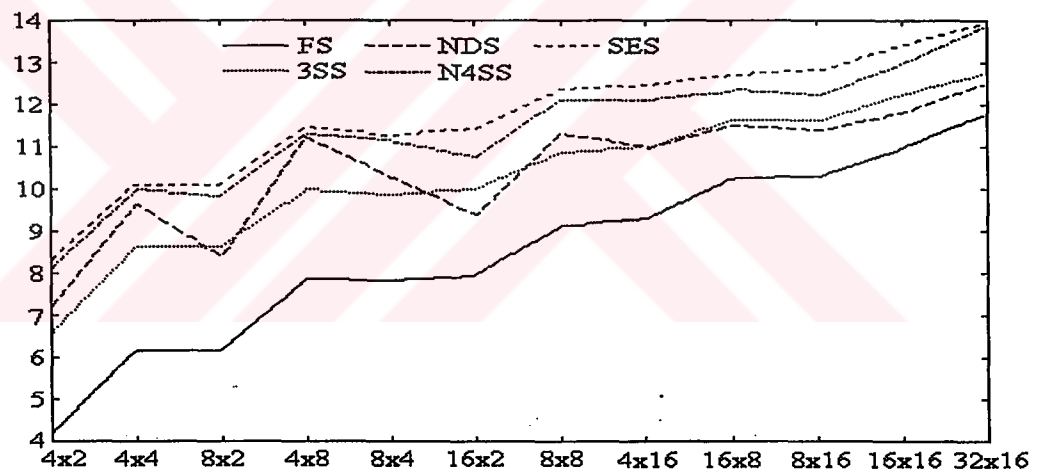


(b)

Figure 3.21 The Football sequence average MSE values at different macroblock sizes for (a) FS, N3SS, NDS, 3SS (b) FS, OTS, TDL, N4SS, SES



(a)



(b)

Figure 3.22 The Miss America sequence average MSE values at different macroblock sizes for (a) FS, N3SS, OTS, TDL (b) FS, 3SS, NDS, N4SS, SES

3.12 The Performance Results of Block Motion Estimation Algorithms with Overlapping Block Motion Compensation

Overlapping block motion compensation (OBMC) is an advanced scheme which is used by H.263 for block motion compensation. The OBMC algorithm overlaps, windows, and sums prediction blocks in order to reduce the error and effect of block boundary discontinuities (Meier et al., 1999). In this method, each 16x16 current macroblock (A_N) is divided into four 8x8 subblocks; upper-left (A_N^{UL}), upper-right (A_N^{UR}), down-left (A_N^{DL}) and down-right (A_N^{DR}). The current macroblock A_N and its subblocks A_N^{UL} , A_N^{UR} , A_N^{DL} , A_N^{DR} are shown in Figure 3.23. In the OBMC, each 8x8 subblock is motion compensated using three motion vectors; motion vector of current macroblock (MV1) and motion vectors of neighbour macroblocks (MV2 and MV3). In Figure 3.23 the motion compensation procedure for the upper-left 8x8 subblock is shown. C_N and B_N are the macroblock neighbours of A_N^{UL} . The motion vectors MV2 and MV3 are belong to the C_N and B_N respectively. C_{N-1}^{UL} is the upper-left 8x8 subblock of the macroblock C_{N-1} pointed by MV2 in the previous frame. Let RC_{N-1} be the macroblock to the right of the C_{N-1} in the previous frame. RC_{N-1}^{UL} is the upper-left subblock of RC_{N-1} . Since the A_N is on the right hand side of the C_N macroblock then the RC_{N-1}^{UL} subblock is used in the compensation algorithm. B_{N-1}^{UL} is the upper-left 8x8 subblock of the macroblock B_{N-1} pointed by MV3 in the previous frame. Let DB_{N-1} be the down-side macroblock of the B_{N-1} in the previous frame. DB_{N-1}^{UL} is the upper-left subblock of DB_{N-1} . Since the A_N is on the down-side of the B_N macroblock then the DB_{N-1}^{UL} subblock is used in the compensation algorithm. Let UD_{N-1} be the upper-side macroblock of the D_{N-1} in the previous frame. UD_{N-1}^{DL} is the down-left subblock of UD_{N-1} . Let LE_{N-1} be the left hand side macroblock of the E_{N-1} in the previous frame. LE_{N-1}^{DR} is the down-right subblock of LE_{N-1} . Let A_{N-1}^{UL} is the upper-left 8x8 subblock of the macroblock A_{N-1} pointed by MV1 in the previous frame. A_{N-1}^{UL} is also used in the compensation algorithm.

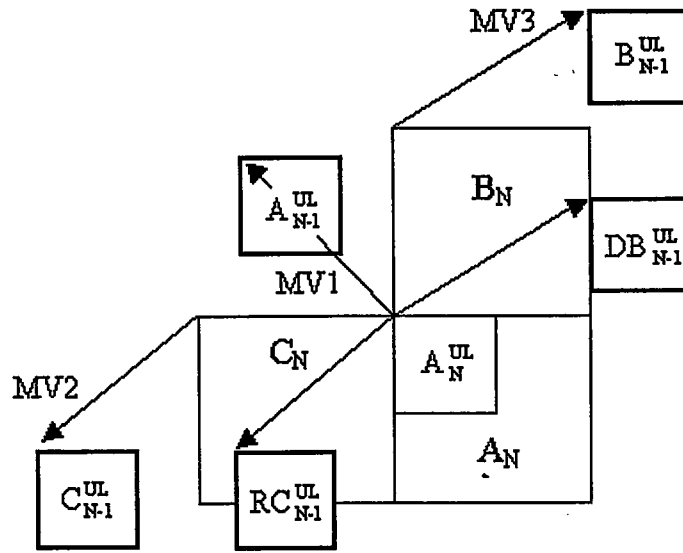


Figure 3.23 OBMC for upper left half of the Macroblock

In the compensation procedure the matrix M_C shown in Figure 3.24(a) is used for the subblock A_{N-1}^{UL} , the matrix M_{UD} shown in Figure 3.24(b) is used for upper or down subblocks (DB_{N-1}^{UL}), the matrix M_{RL} shown in Figure 3.24(c) is used for right or left subblocks (RC_{N-1}^{UL}). Assume that D_N and E_N are the down and right hand side macroblock neighbours of A_N . Therefore the subblocks of the the motion compensated macroblock \hat{A}_N are computed by

$$\begin{aligned}
 \hat{A}_N^{UL} &= M_C \cdot A_{N-1}^{UL} + M_{UD} \cdot DB_{N-1}^{UL} + M_{RL} \cdot RC_{N-1}^{UL} \\
 \hat{A}_N^{UR} &= M_C \cdot A_{N-1}^{UR} + M_{UD} \cdot DB_{N-1}^{UR} + M_{RL} \cdot LE_{N-1}^{UR} \\
 \hat{A}_N^{DL} &= M_C \cdot A_{N-1}^{DL} + M_{UD} \cdot UD_{N-1}^{DL} + M_{RL} \cdot RC_{N-1}^{DL} \\
 \hat{A}_N^{DR} &= M_C \cdot A_{N-1}^{DR} + M_{UD} \cdot UD_{N-1}^{DR} + M_{RL} \cdot LE_{N-1}^{DR}
 \end{aligned} \tag{3.2}$$

The OBMC Weights in H.263 are shown in Figure 3.24.

4	5	5	5	5	5	5	4
5	5	5	5	5	5	5	5
5	5	6	6	6	6	5	5
5	5	6	6	6	6	5	5
5	5	6	6	6	6	5	5
5	5	6	6	6	6	5	5
5	5	5	5	5	5	5	5
4	5	5	5	5	5	5	4

(a)

2	2	2	2	2	2	2	2
1	1	2	2	2	2	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	1	1
1	1	2	2	2	2	1	1
2	2	2	2	2	2	2	2

(b)

2	1	1	1	1	1	1	2
2	2	1	1	1	1	2	2
2	2	1	1	1	1	2	2
2	2	1	1	1	1	2	2
2	2	1	1	1	1	2	2
2	2	1	1	1	1	2	2
2	2	1	1	1	1	2	2
2	1	1	1	1	1	1	2

(c)

Figure 3.24 OBMC matrixes for a 16x16 macroblock. (a) M_C matrix for current subblock (b) M_{UD} matrix for top/bottom subblock.(c) M_{RL} matrix for left/right subblock.

In H.263 Standard the macroblock is 16x16 pixel size and therefore the size of motion compensated small luminance blocks are 8x8. Since the block motion estimation algorithms described above were tested with different size of macroblocks, additional new OBMC matrixes were also designed for 8x8, 16x8, 8x16, 4x4 macroblock sizes. Those OBMC matrixes and their coefficients are shown in Figure 3.25, 3.26, 3.27 and 3.28.

4	5	5	5	5	5	5	4
5	5	6	6	6	6	5	5
5	5	6	6	6	6	5	5
4	5	5	5	5	5	5	4

(a)

2	2	2	2	2	2	2	2
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2

(b)

2	1	1	1	1	1	1	2
2	2	1	1	1	1	2	2
2	2	1	1	1	1	2	2
2	1	1	1	1	1	1	2

(c)

Figure 3.25 OBMC matrixes for a 16x8 macroblock. (a) M_C matrix for current subblock (b) M_{UD} matrix for top/bottom subblock.(c) M_{RL} matrix for left/right subblock.

4	5	5	4
5	5	5	5
5	6	6	5
5	6	6	5
5	6	6	5
5	6	6	5
5	5	5	5
4	5	5	4

(a)

2	2	2	2
1	2	2	1
1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1
1	2	2	1
2	2	2	2

(b)

2	1	1	2
2	1	1	2
2	1	1	2
2	1	1	2
2	1	1	2
2	1	1	2
2	1	1	2
2	1	1	2

(c)

Figure 3.26 OBMC matrixes for a 8x16 macroblock. (a) M_C matrix for current subblock (b) M_{UD} matrix for top/bottom subblock.(c) M_{RL} matrix for left/right subblock.

4	5	5	4
5	6	6	5
5	6	6	5
4	5	5	4

(a)

2	2	2	2
1	1	1	1
1	1	1	1
2	2	2	2

(b)

2	1	1	2
2	1	1	2
2	1	1	2
2	1	1	2

(c)

Figure 3.27 OBMC matrixes for a 8x8 macroblock. (a) M_C matrix for current subblock (b) M_{UD} matrix for top/bottom subblock.(c) M_{RL} matrix for left/right subblock.

6	6
6	6

(a)

1	1
1	1

(b)

1	1
1	1

(c)

Figure 3.28 OBMC matrixes for a 4x4 macroblock. Each small luminance block size is 2x2.(a) matrix for motion vector of current luminance block. (b)matrix for motion vector of top/bottom luminance block.(c)matrix for motion vector of left / right luminance block

The OBMC algorithm results are taken by using 75th and 76th frames of Football, 27th and 28th frames of Miss America sequence in CIF format. The FS, 3SS, OTS, TDL, N3SS, I3SS, NDS, SES block motion estimation algorithms are used for the OBMC algorithm. The macroblock sizes are defined as 16x16, 16x8, 8x16, 8x8, 4x4. For each macroblock type the corresponding OBMC matrix is used.

In Table 3.5 to 3.10 the MSE results of the block motion estimation algorithms without OBMC, with OBMC and the decrease in MSE are shown for the Football and Miss America sequences.

It is seen from tables that the OBMC algorithm decreases the MSE value. It is also seen that when the MSE values are big the OBMC algorithm decreases MSE value sharply, but otherwise the decrease in MSE value is not significant.

Either much or less, decrease in MSE value by OBMC algorithm has some advantages in coding. Since the MSE value is smaller than the original value after OBMC, the number of significant wavelet coefficients after Discrete Wavelet Transform (DWT) decreases. This means, DWT of error frame contain more zero wavelet coefficients. Thus, the compression ratio increases.

THE YOKS...

Table 3.5 MSE results for 75th and 76th frames of Football sequence without OBMC

	FS	3SS	OTS	TDL	N3SS	I3SS	NDS	SES
16x16	287.4	309.7	380.2	330.8	303.4	365.7	317.8	376.8
16x8	226.8	269.4	320.8	275.7	258.9	313.1	281.0	328.9
8x16	227.7	272.6	311.3	274.8	268.4	308.2	269.6	328.2
8x8	173.5	232.4	265.7	237.8	218.8	264.7	242.7	294.3
4x4	95.5	154.8	162.9	177.7	137.0	186.2	206.5	203.8

Table 3.6 MSE results for 75th and 76th frames of Football sequence with OBMC

	FS	3SS	OTS	TDL	N3SS	I3SS	NDS	SES
16x16	237.8	252.8	314.8	269.6	247.8	308.6	260.4	314.2
16x8	190.8	223.3	262.3	228.0	215.1	263.4	235.2	269.9
8x16	192.7	226.7	262.9	227.4	222.3	264.9	226.3	271.3
8x8	148.2	198.4	227.0	200.8	183.9	226.3	202.7	249.2
4x4	85.7	136.9	143.8	156.5	119.7	165.6	174.6	178.8

Table 3.7 Decrease in MSE for 75th and 76th frames of Football sequence by OBMC

	FS	3SS	OTS	TDL	N3SS	I3SS	NDS	SES
16x16	%17.3	%18.4	%17.2	%18.5	%18.3	%15.6	%18.1	%16.6
16x8	%15.9	%17.1	%18.2	%17.3	%16.9	%15.9	%16.3	%17.9
8x16	%15.4	%16.8	%15.5	%17.2	%17.2	%14.1	%16.1	%17.3
8x8	%14.6	%14.6	%14.6	%15.6	%15.9	%14.5	%16.5	%15.3
4x4	%10.3	%11.6	%11.7	%11.9	%12.6	%11.1	%15.4	%12.3

Table 3.8 MSE results for 27th and 28th frames of Miss America without OBMC

	FS	3SS	OTS	TDL	N3SS	I3SS	NDS	SES
16x16	9.5	9.9	9.7	9.9	9.6	10.1	9.9	10.1
16x8	9.0	9.4	9.3	9.5	9.2	9.8	9.6	9.8
8x16	9.2	9.4	9.2	9.5	9.3	9.8	9.7	9.9
8x8	8.1	8.7	8.5	8.8	8.5	9.2	9.3	9.3
4x4	5.6	6.7	6.4	6.9	6.5	7.0	7.0	7.5

Table 3.9 MSE results for 27th and 28th frames of Miss America with OBMC

	FS	3SS	OTS	TDL	N3SS	I3SS	NDS	SES
16x16	8.6	8.9	8.8	8.9	8.7	9.1	9.0	9.2
16x8	7.8	8.3	8.1	8.4	7.8	8.5	8.4	8.5
8x16	8.5	8.7	8.6	8.6	8.4	8.7	8.6	9.6
8x8	7.8	7.9	7.6	7.7	7.4	8.3	8.1	8.3
4x4	5.4	6.0	5.9	6.8	5.8	6.1	6.2	6.7

Table 3.10 Decrease in MSE for 27th and 28th frames of Miss America by OBMC

	FS	3SS	OTS	TDL	N3SS	I3SS	NDS	SES
16x16	%9.7	%10.1	%9.1	%9.6	%10.0	%9.5	%9.7	%9.0
16x8	%13.4	%12.0	%12.5	%11.6	%15.0	%13.2	%13.0	%13.1
8x16	%7.9	%8.2	%7.2	%9.3	%9.7	%11.7	%10.6	%2.6
8x8	%3.8	%9.4	%10.9	%12.1	%12.6	%10.3	%12.9	%10.4
4x4	%3.2	%11.0	%8.0	%1.7	%11.1	%13.1	%10.6	%11.7

A visual result of the OBMC algorithm is shown in Figure 3.29. In Figure 3.29(a) the reconstructed 76th frame of Football sequence is shown when the block motion estimation algorithm is NDS and macroblock size is 8x8. In Figure 3.29(b) the OBMC result is shown. It is clearly seen from Figure 3.29(b) that the OBMC algorithm decreases both MSE value and block effects.

**(a)****(b)**

Figure 3.29 a) NDS result at 8x8 macroblock size for 76th frame of Football b) The result of the OBMC algorithm.

3.13 Conclusion

In this chapter, the popular block motion estimation algorithms are computer simulated using standard 100 frames of CIF Football and Miss America video sequences. These block motion estimation algorithms are Full Search (FS), Two Dimensional Logarithmic Search (TDL), One at a Time Search (OTS), Three Step Search (3SS), An Improved Three Step Search (I3SS), A New Three Step Search (N3SS), A Novel Four Step Search (N4SS), Simple and Efficient Search (SES) and New Diamond Search (NDS). The simulation results show that the SPN order from best to worst is SES, TDL, NDS, I3SS, N4SS, N3SS, 3SS, OTS and FS for the Football sequence and SES, TDL, I3SS, NDS, N4SS, 3SS, N3SS, OTS and FS for the Miss America sequence. It is clearly seen that SES, TDL and NDS algorithms are faster than the rest of the block motion estimation algorithms.

The block motion estimation algorithm is performed by using large macro block size (16x16). That's why the average MSE value for large block sizes is considered. The results show that the NDS has the best performance. Unfortunately the SES, TDL and I3SS, which have very good performances as far as SPN is concerned, have very poor MSE values. When both average SPN and MSE are concerned, it can be seen easily that NDS has the best performance. Therefore as a block motion estimation algorithm the NDS is selected to track the local motion.

The overlapping block motion compensation technique which is an advance scheme used in the H.263 standard is described, computer simulated and its performance results are also presented in this chapter. The OBMC technique is used after the New Diamond Search block motion algorithm to reduce the block artifacts.

CHAPTER FOUR

DISCRETE WAVELET TRANSFORM

4.1 Introduction

The early work about wavelet was in 1980's by Morlet, Grossman, Meyer, Mallat, and others, but it was the paper by Ingrid Daubechies (Daubechies I., 1988) that caught the attention of the larger applied mathematics communities in signal processing, statistics, numerical analysis and image processing.

The purpose of the most modern wavelet research is to create a set of basis functions and transforms that will give an informative, efficient, and useful description of a function or signal. If the signal is represented as a function of time, wavelets provide efficient localization in both time and frequency. Another feature of wavelets is the multiresolution where the decomposition of a signal is in terms of the resolution of detail.

The multiresolution decomposition seems to separate components of a signal in a way that is superior to most other methods for analysis, processing, or compression. Because of this powerful and flexible decomposition, linear and nonlinear processing of signals in the wavelet transform domain offers new methods for signal detection, filtering, and compression.

4.2 The Wavelet Characteristics

The wavelet expansion set is not unique. There are many different wavelet systems that can be used effectively, but all seem to have the following three general characteristics;

1) A wavelet system is a set of building blocks to construct or represent a signal or function. It is a two-dimensional expansion set (usually a basis) for some class of one (or higher) dimensional signals. In other words, if the wavelet set is given by $\Psi_{j,k}(t)$ for indices of $j, k = 1, 2, \dots$, a linear expansion would be $f(t) = \sum_k \sum_j a_{j,k} \Psi_{j,k}(t)$ for some set of coefficients $a_{j,k}$.

2) The wavelet expansion gives a time-frequency localization of the signal. This means most of the energy of the signal is well represented by a few expansion coefficients, $a_{j,k}$.

3) The calculation of the coefficients from the signal can be done efficiently. It turns out that many wavelet transforms (the set of expansion coefficients) can be calculated with $O(N)$ operations. This means the number of floating-point multiplications and additions increase linearly with the length of the signal. On the other hand the fast Fourier transform (FFT) requires $O(N \log(N))$ operations.

4) All first generation wavelet systems are generated from a single scaling function or wavelet by simple scaling and translation. The two dimensional parameterization is achieved from the function (sometimes called the generating wavelet or mother wavelet) $\Psi(t)$ by

$$\Psi_{j,k}(t) = 2^{j/2} \Psi(2^j t - k) \quad j, k \in \mathbb{Z} \quad (4.1)$$

where \mathbb{Z} is the set of all integers and the factor $2^{j/2}$ maintains a constant norm independent of scale j . This parameterization of the time or space location by k and the frequency or scale (actually the logarithm of scale) by j turns out to be effective.

5) Almost all useful wavelet systems also satisfy the multiresolution conditions. This means that if a set of signals can be represented by a weighted sum of scaling function $\Phi(t - k)$, then a larger set (including the original) can be represented by a weighted sum of $\Phi(2t - k)$. In other words, if the basic expansion signals are made

half as wide and translated in steps half as wide, they will represent a larger class of signals exactly or give better approximation of any signal.

6) The lower resolution coefficients can be calculated from the higher resolution coefficients by a tree-structured algorithm called a filter bank. This allows a very efficient calculation of the expansion coefficients (also known as the discrete wavelet transform) and relates wavelet transform to an older area in digital signal processing.

4.3 The Wavelet Expansion

A signal or function $f(t)$ can often be better analyzed, described, or processed if expressed as a linear decomposition by

$$f(t) = \sum_l a_l \Psi_l(t) \quad (4.2)$$

Where l is an integer index for the finite or infinite sum, a_l are the real-valued expansion coefficients, and $\Psi_l(t)$ are a set of real-valued functions of t called the expansion set. If the expansion (4.2) is unique, the set is called a basis for the class of functions that can be so expressed. If the basis is orthogonal, meaning

$$\langle \Psi_k(t), \Psi_l(t) \rangle = \int \Psi_k(t) \Psi_l(t) dt = 0 \quad (4.3)$$

then the coefficients can be calculated by the inner product

$$a_k = \langle f(t), \Psi_k(t) \rangle = \int f(t) \Psi_k(t) dt \quad (4.4)$$

One can see that substituting equation (4.2) into equation (4.4) and using equation (4.3) gives the single a_k coefficient. For the wavelet expansion, a two-parameter system is constructed such that expansion (4.2) becomes

$$f(t) = \sum_k \sum_j a_{j,k} \Psi_{j,k}(t) \quad (4.5)$$

here both j and k are integer indices and the $\Psi_{j,k}(t)$ are the wavelet expansion functions that usually form an orthogonal basis. The set of expansion coefficients

$a_{j,k}$ are called the discrete wavelet transform (DWT) of $f(t)$ and definition (4.5) is the inverse transform.

The goal here is to generate a set of expansion functions such that any signal in $L^2(\mathbb{R})$ (the space of square integrable functions) can be represented by the series

$$f(t) = \sum_{j,k} a_{j,k} 2^{j/2} \Psi(2^j t - k) \quad (4.6)$$

The equation (4.6) can be obtained by substituting equation (4.1) into equation (4.5). A more specific form indicating how the $a_{j,k}$'s are calculated can be written using inner products as

$$f(t) = \sum_{j,k} \langle \Psi_{j,k}(t), f(t) \rangle \Psi_{j,k}(t) \quad (4.7)$$

4.4 The Scaling Function

In order to understand the idea of multiresolution, first of all the scaling function definition and then wavelet function definition have to be understood. A set of scaling functions can be defined in terms of integer translates of the basic scaling function by

$$\Phi_k(t) = \Phi(t - k) \quad k \in \mathbb{Z}, \quad \Phi \in L^2 \quad (4.8)$$

The subspace of $L^2(\mathbb{R})$ spanned by these functions is defined as

$$V_0 = \overline{\text{Span}\{\Phi_k(t)\}} \quad (4.9)$$

for all integers k from minus infinity to infinity. The over-bar denotes closure. This means that

$$f(t) = \sum_k a_k \Phi_k(t) \quad \text{for any } f(t) \in V_0 \quad (4.10)$$

One can generally increase the size of the subspace spanned by changing the time scale of the scaling functions. A two-dimensional family of functions is generated from the basic scaling function by scaling and translation by

$$\Phi_{j,k}(t) = 2^{j/2} \Phi(2^j t - k) \quad (4.11)$$

whose span over k is

$$V_j = \overline{\text{Span}\{\Phi_k(2^j t)\}} = \overline{\text{Span}\{\Phi_{j,k}(t)\}} \quad (4.12)$$

for all integers $k \in \mathbb{Z}$. This means that if $f(t) \in V_j$, then it can be expressed as

$$f(t) = \sum_k a_k \Phi(2^j t + k) \quad (4.13)$$

4.5 Multiresolution Analysis

In order to explain scale or resolution, the basic requirement of multiresolution analysis is formulated by a nesting of the spanned spaces as

$$\dots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \dots \subset L^2 \quad (4.14)$$

or

$$V_j \subset V_{j+1} \quad \text{for all } j \in \mathbb{Z} \quad (4.15)$$

with

$$V_{-\infty} = \{0\}, \quad V_{\infty} = L^2 \quad (4.16)$$

The space that contains high resolution signals will contain those of lower resolution also. Because of the definition of V_j , the spaces have to satisfy a natural scaling condition

$$f(t) \in V_j \Leftrightarrow f(2t) \in V_{j+1} \quad (4.17)$$

which obtains elements in a space are simply scaled version of the elements in the next space. The relationship of the spanned spaces is illustrated in Figure 4.1.

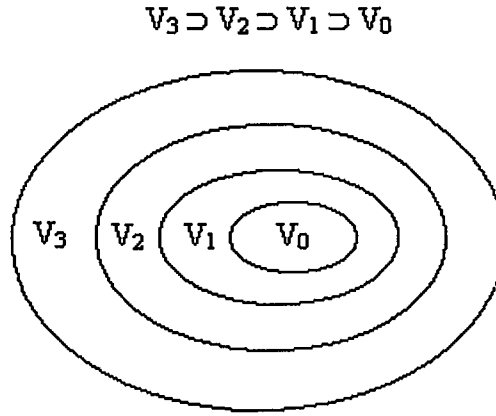


Figure 4.1 Nested vector spaces spanned by the scaling functions

The nesting of the spans of $\Phi(2^j t - k)$, denoted by V_j and shown in expression (4.14) and (4.17) and graphically illustrated in Figure 4.1, is achieved by requiring that $\Phi(t) \in V_1$, which means that if $\Phi(t)$ is in V_0 , it is also in V_1 , the space spanned by $\Phi(2t)$. Therefore $\Phi(t)$ can be expressed in terms of a weighted sum of shifted $\Phi(2t)$ as

$$\Phi(t) = \sum_n h_s(n) \sqrt{2} \Phi(2t - n), \quad n \in \mathbb{Z} \quad (4.18)$$

where the coefficients $h_s(n)$ are a sequence of real or sometimes complex numbers called the ‘scaling function coefficients’ and the $\sqrt{2}$ maintains the norm of the scaling function with the scale of two.

4.6 The Wavelet Functions Used by the Introduced Encoder

The important features of a signal can be better described or parameterized, not by using only scaling functions $\Phi_{j,k}(t)$, but also by defining a slightly different set of wavelet functions $\Psi_{j,k}(t)$ that span the differences between the spaces spanned by the various scales of the scaling function (Rao & Bopardikar, 1998). These functions are called wavelet functions.

There are several advantages when the scaling functions and wavelets be orthogonal. Orthogonal basis functions allow simple calculation of expansion coefficients and have a Parseval's theorem that allows a partitioning of the signal energy in the wavelet transform domain. The orthogonal complement of V_j in V_{j+1} is defined as W_j . Therefore all the members of V_j are orthogonal to all members of W_j . The orthogonality feature is shown in equation (4.19).

$$\langle \Phi_{j,k}(t), \Psi_{j,l}(t) \rangle = \int \Phi_{j,k}(t) \Psi_{j,l}(t) dt = 0, \quad j, k, l \in Z \quad (4.19)$$

The relationship of the various subspaces can be seen from the following expressions. From equation (4.14) it is seen that we may start at any V_j , say at $j=0$, and write

$$V_0 \subset V_1 \subset V_2 \subset \dots \subset L^2 \quad (4.20)$$

The wavelet subspace W_0 is defined such that

$$V_1 = V_0 \oplus W_0 \quad (4.21)$$

which extends to

$$V_2 = V_0 \oplus W_0 \oplus W_1 \quad (4.22)$$

In general this gives

$$L^2 = V_0 \oplus W_0 \oplus W_1 \oplus \dots \quad (4.23)$$

when V_0 is the initial space spanned by the scaling function $\Phi(t-k)$. Figure 4.2 shows the nesting of the scaling function spaces V_j for different scales j and the relationship between wavelet and scaling functions.

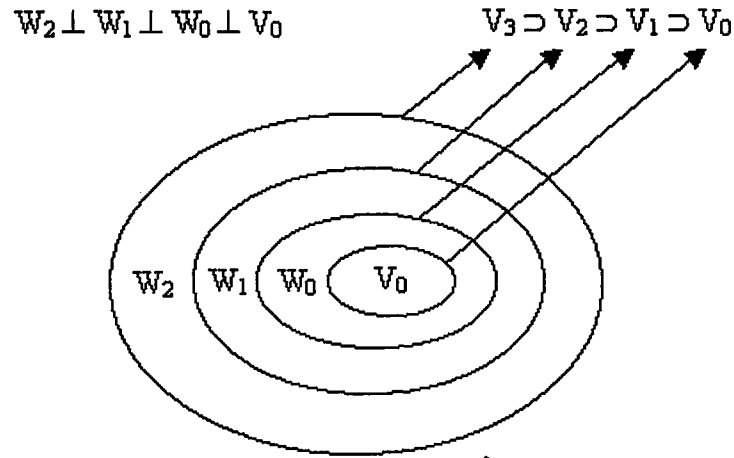


Figure 4.2 Scaling function and wavelet vector spaces

The scale of initial space is arbitrary and could be chosen at a higher resolution. For high resolution $j = 10$ to give

$$L^2 = V_{10} \oplus W_{10} \oplus W_{11} \oplus \dots \quad (4.24)$$

or at a lower resolution such as $j = -5$ to give

$$L^2 = V_{-5} \oplus W_{-5} \oplus W_{-4} \oplus \dots \quad (4.25)$$

or at even $j = -\infty$ where the equation (4.23) becomes

$$L^2 = \dots \oplus W_{-2} \oplus W_{-1} \oplus W_0 \oplus W_1 \oplus W_2 \oplus \dots \quad (4.26)$$

Since the wavelet functions form in the space spanned by the next narrower scaling function $W_0 \subset V_1$, they can be represented by a weighted sum of shifted scaling function $\Phi(2t)$ defined in equation (4.18) by

$$\Psi(t) = \sum_n h_w(n) \sqrt{2} \Phi(2t - n), \quad n \in \mathbb{Z} \quad (4.27)$$

for some set of coefficients $h_w(n)$. It is seen that the wavelet functions spans the 'difference' or orthogonal complement spaces. The wavelet coefficients are required by orthogonality to related to the scaling function coefficients by

$$h_w(n) = (-1)^n h_s(1-n) \quad (4.28)$$

The function generated by equation (4.27) gives the mother wavelet $\Psi(t)$, and expansion of the mother wavelet function is given by equation (4.1).

It is seen that a set of functions $\Phi_k(t)$ and $\Psi_{j,k}(t)$ can span all of $L^2(\mathbb{R})$. According to definition (4.23) any function $f(t) \in L^2(\mathbb{R})$ can be written

$$f(t) = \sum_{k=-\infty}^{\infty} c(k) \Phi_k(t) + \sum_{j=0}^{\infty} \sum_{k=-\infty}^{\infty} d_j(k) \Psi_{j,k}(t) \quad (4.29)$$

as a series expansion in terms of the scaling and wavelet functions. In the definition (4.29), the first summation gives a function that is a low resolution or coarse approximation of $f(t)$. For each increasing index j in the second summation, a finer resolution function is added, which adds increasing detail.

The coefficients in the wavelet expansion (4.29) are called 'discrete wavelet transform' of the signal $f(t)$. If the wavelet system is orthogonal, the wavelet and the scaling function coefficients can be calculated by inner products that is given in the equation (4.30) and (4.31).

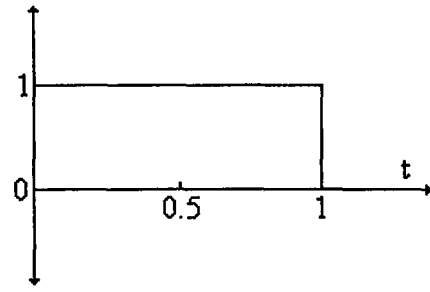
$$c_j(k) = \langle f(t), \Phi_{j,k}(t) \rangle = \int f(t) \Phi_{j,k}(t) dt \quad (4.30)$$

$$d_j(k) = \langle f(t), \Psi_{j,k}(t) \rangle = \int f(t) \Psi_{j,k}(t) dt \quad (4.31)$$

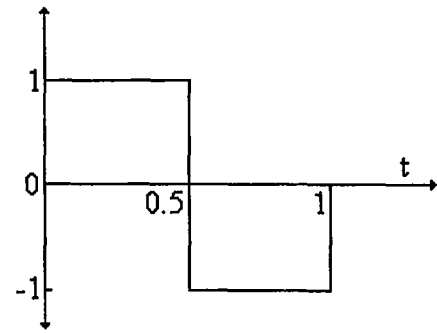
Here the $c_j(k)$ and $d_j(k)$ represent the scaling function coefficients and the wavelet function coefficients respectively. Some of the wavelet and scaling function coefficients are given in Table 4.1 (Kim & Li, 1998). The Haar, Daubechies(4) and Biorthogonal(9,7) wavelet and scaling functions are also shown in Figure 4.3.

Table 4.1 Haar, Daubechies and Biorthogonal wavelet and scaling function coefficients

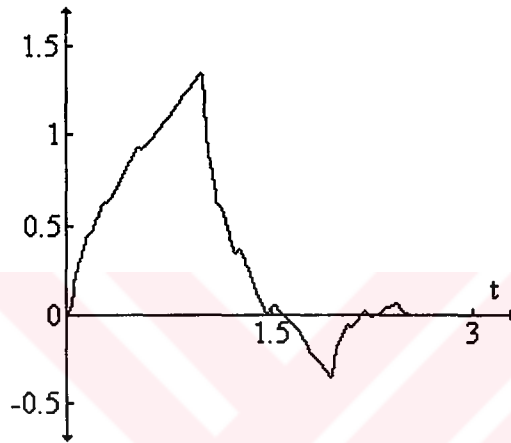
	Scaling Function Coefficients	Wavelet Function Coefficient
Haar	$h_s(1)=h_s(2)=0.707106781$	$h_w(1)=-h_w(2)=0.707106781$
Daubechies (4)	$h_s(1)=0.482962913$ $h_s(2)=0.836516303$ $h_s(3)=0.224143868$ $h_s(4)=-0.12940952$	$h_w(1)=h_s(4)$ $h_w(2)=-h_s(3)$ $h_w(3)=h_s(2)$ $h_w(4)=-h_s(1)$
Daubechies (6)	$h_s(1)=0.33267055$ $h_s(2)=0.806891508$ $h_s(3)=0.459877504$ $h_s(4)=-0.13501101$ $h_s(5)=-0.08544127$ $h_s(6)=0.035226283$	$h_w(1)=0.035226283$ $h_w(2)=0.08544127$ $h_w(3)=-0.13501101$ $h_w(4)=-0.459877504$ $h_w(5)=0.806891508$ $h_w(6)=-0.33267055$
Biorthogonal(9.7)	$h_s(1)=h_s(9)=0.603162084$ $h_s(2)=h_s(8)=0.266862099$ $h_s(3)=h_s(7)=-0.07822015$ $h_s(4)=h_s(6)=-0.01686378$ $h_s(5)=0.026749849$	$h_w(1)=h_w(7)=0.557553697$ $h_w(2)=h_w(6)=-0.29564134$ $h_w(3)=h_w(5)=-0.02877217$ $h_w(4)=0.045636671$
Biorthogonal(9.3)	$h_s(1)=h_s(9)=0.03314563$ $h_s(2)=h_s(8)=-0.06629126$ $h_s(3)=h_s(7)=-0.17677669$ $h_s(4)=h_s(6)=0.419844651$ $h_s(5)=0.99436891$	$h_w(1)=h_w(3)=-0.35355339$ $h_w(2)=0.707106781$



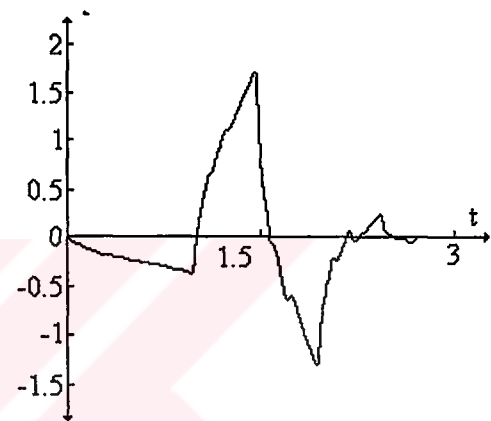
(a)



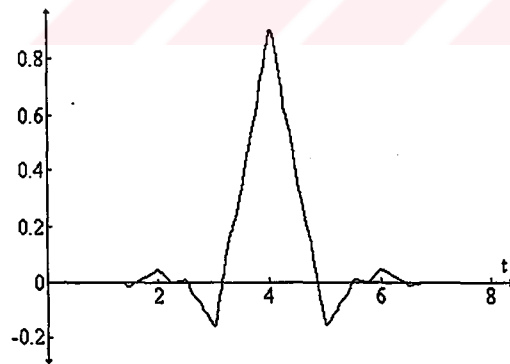
(b)



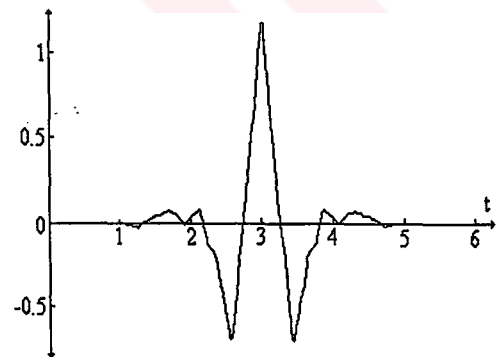
(c)



(d)



(e)



(f)

Figure 4.3 Functions of a) Haar scaling b) Haar wavelet c) Daubechies(4) scaling d) Daubechies (4) wavelet e) Biorthogonal (9.7) scaling f) Biorthogonal (9.7) wavelet

In the proposed video compression system a short and a long wavelet filter banks are used together. Since the long filters provide good frequency localization the Biorthogonal (9.3) is used at the first level of DWT. Biorthogonal (9.3) is also very good at image compression (Xiong et al., 1999), (Zhou & Ma, 2000). Since the short filters provide less ringing artifacts the Haar is used at the second and third levels of DWT (Chang & Lu, 1999), (Shen & Delp, 1999), (Sodagar et al., 1999), (Yeung, 1997), (Zheng & Quan, 1996).

In many applications only the coefficients of $h_s(n)$ and $h_w(n)$ are used instead of scaling and wavelet functions. In the video processing, the $h_s(n)$ and the $h_w(n)$ coefficients are used to analysis or synthesis a video frame. The two-band analysis tree is shown in Figure 4.4.

In the first stage of the two-band analysis tree given in Figure 4.4, the spectrum of $c_{j+1}(k)$ is divided into a lowpass and highpass band. The lowpass coefficients of $c_j(k)$ are obtained by using scaling function coefficients. On the other hand the highpass coefficients of $d_j(k)$ are obtained by using wavelet function coefficients. The first stage of DWT divides the spectrum into two equal parts. The second stage divides the lower half into quarters and so on.

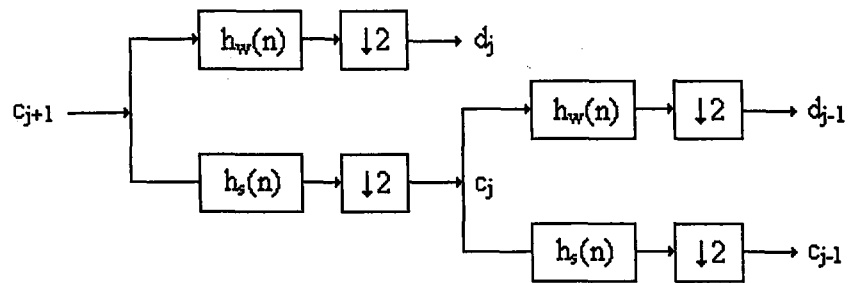


Figure 4.4 Two-stage two-band analysis tree

For synthesis in the filter bank we have a sequence of first up-sampling or stretching, then filtering that is shown in Figure 4.5. In the synthesis process, firstly the input signal is stretched to twice its original length. Then using the same scaling

and wavelet filter coefficients as in the analysis the original signal can be obtained without any loss.

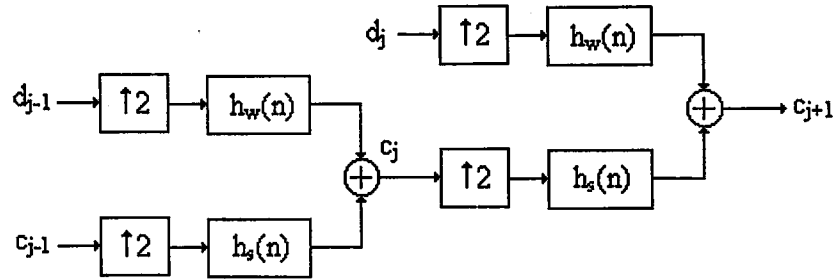


Figure 4.5 Two-stage two-band synthesis tree

The refinement matrix M is important to determining the properties of wavelet systems [30]. The matrix M is obtained from the basic recursion equation (4.18) by evaluating $\Phi(t)$ at integers. This looks like a convolution matrix with the even (or odd) rows removed. This matrix is illustrated for $N=6$ by

$$\sqrt{2} \begin{bmatrix} h_0 & 0 & 0 & 0 & 0 & 0 \\ h_2 & h_1 & h_0 & 0 & 0 & 0 \\ h_4 & h_3 & h_2 & h_1 & h_0 & 0 \\ 0 & h_5 & h_4 & h_3 & h_2 & h_1 \\ 0 & 0 & 0 & h_5 & h_4 & h_3 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Phi_0 \\ \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \end{bmatrix} = \begin{bmatrix} \Phi_0 \\ \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \end{bmatrix} \quad (4.32)$$

which we write in matrix form as $M\Phi = \Phi$. Here M is the 6×6 matrix of the $h(n)$ and Φ is the 6×1 vectors of integer samples of $\Phi(t)$. This matrix representation can also be written for the $\Psi(t)$ by using wavelet coefficients instead of scaling coefficients.

4.7 Conclusion

The Chapter 4 describes some useful features of the discrete wavelet transform. These are the definition of discrete wavelet transform, wavelet and scaling function expansion, multiresolution analysis and matrix representations.

In order to remove the spatial correlation a three level of DWT is used instead of DCT. DWT seems to have better performance than DCT for image and video coding (Xiong, Z. et al., 1999). DWT also provides us the scalability functionalities of image and video. The encoder uses different filter bank at each level of the decomposition. Since the longer filters provide good frequency localization and shorter filters cause less ringing artifacts, the Biorthogonal 9.3 filter bank is used at the first level of the DWT decomposition followed by the Haar filter for the remaining two levels.

The scaling and the wavelet function coefficients of the Daubechies 4, Daubechies 6, Biorthogonal 9.7, Biorthogonal 9.3 and the Haar wavelet (Rao, R.M. & Bopardikar, A.S., 1998) are presented. The graphical representation of the Biorthogonal 9.3 and the Haar scaling and wavelet functions, which are used by the proposed encoder are also given in this chapter.

CHAPTER FIVE

QUANTIZATION

Digital signal processing involves sampling of an analog signal and quantization of the sampled data into a finite number of levels. These levels are called quantization levels that are based on some criteria such as minimization of the quantizer distortion.

Quantizers can be classified as uniform or nonuniform. A uniform quantizer is completely defined by the number of levels it has, its step size, and whether it is a midriser or a midtreader. A nonuniform quantizer implies that the step sizes are not constant. Therefore, the nonuniform quantizers has to be specified by the input and output levels.

Quantization is many-to-one mapping which is irreversible. It can be performed by scalar and vector quantizers. Scalar quantization refers to element-by-element quantization of data, whereas quantization of a block of data is known as vector quantization.

5.1 Scalar Quantization

In scalar quantization each source output is quantized into a number of levels and these levels then are encoded into a K-ary sequence where K is the number of levels. The goal of the scalar quantizer is to map the set of real numbers into a finite set and at the same time minimize the distortion introduced. In scalar quantization the set of real numbers R is partitioned into K disjoint subsets or intervals denoted $I_k = (k = 1, 2, \dots, K)$. Corresponding to each interval I_k , a reproduction point \hat{x}_k that

usually belongs to I_k is chosen. If the source output at a time i , $x(i)$, belongs to I_k , then it is represented by \hat{x}_k , which is the quantized version of $x(i)$. The reproduction points \hat{x}_k can be represented by a binary codeword of fixed length. Since there are K possibilities for the quantized levels, $\log_2 K$ bits are enough to encode these levels into binary sequences (K is generally chosen to be a power of 2). The price to pay for a decrease in rate from infinity to $\log_2 K$, of course, the introduction of distortion.

The quantization function $Q(x)$ is defined by

$$Q(x) = \hat{x}_i \text{ for all } x \in I_k \quad (5.1)$$

This function is shown in Figure 5.1 for an 8 level quantizer. The reproduction letters $\{\hat{x}_1, \hat{x}_2, \dots, \hat{x}_k\}$ are also called quantizer levels and the interval endpoints $\{x_0, x_1, \dots, x_k\}$ are called decision thresholds. $Q(x)$ is a non invertible function and some information is lost during the process of quantization.

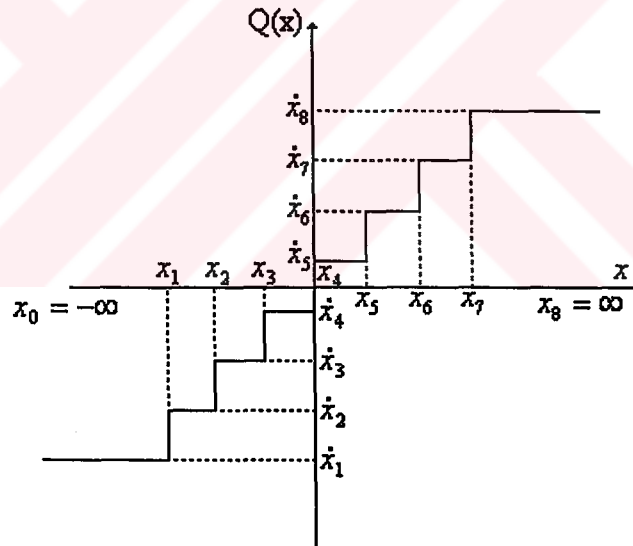


Figure 5.1 Transfer function of a uniform 8-level midriser quantizer

The distortion measure $d(x, \hat{x}_k)$ is defined between the source output realization and its corresponding reproduction value. The distortion is calculated by the distance

between x_k and \dot{x}_k . The squared error is the most common distortion measure and its defined as

$$d(x, \dot{x}_k) = (x - x_k)^2 \quad (5.2)$$

The quantization error q is calculated by taking the average of $d(x, Q(x))$ as

$$q = E[d(x, Q(x))] = \int_{-\infty}^{\infty} d(x, Q(x)) P_x(x) dx \quad (5.3)$$

Where the $P_x(x)$ is the probability distribution of the source. Using the definition (5.1), the expression (5.3) can be written as

$$q = \sum_{i=1}^K \int_{x_{k-1}}^{x_k} d(x, \dot{x}_k) P_x(x) dx \quad (5.4)$$

The quantizer parameters(quantizer levels and decision thresholds) are chosen to minimize the quantization error q given a number of quantizer levels and source distribution. Because $d(x, \dot{x}_k)$ is a monotone, non-decreasing function of distance between x and \dot{x}_k the decision thresholds are defined through $d(x, \dot{x}_k)$ once the quantizer levels $\{\dot{x}_1, \dot{x}_2, \dots, \dot{x}_k\}$ are specified. Therefore the problem of quantizer design reduces to find the set of levels $\{\dot{x}_1, \dot{x}_2, \dots, \dot{x}_k\}$ such that q is minimized. The necessary conditions for the minimization of q are obtained by differentiating equation (5.4) with respect to x_k and \dot{x}_k and equating the results to zero. This gives

$$x_k = \frac{\dot{x}_k + \dot{x}_{k-1}}{2} \quad (5.5)$$

$$\dot{x}_k = \frac{\int_{x_k}^{x_{k+1}} x P_x(x) dx}{\int_{x_k}^{x_{k+1}} P_x(x) dx} \quad (5.6)$$

The equation (5.5) and (5.6) are known as the nearest neighbor condition and the centroid condition respectively. A quantizer satisfying both equations is called

Lloyd-Max quantizer. An iterative algorithm to simultaneously solve both equations is given in (Lloyd, 1982).

5.2 Lloyd-Max Quantization

Lloyd-Max is a non-uniform quantizer based on Mean Square Error (MSE) criterion (Yin, 2000). Let X be a random variable with a probability density function $f(x)$. The decision levels x_i and reconstruction levels r_i are defined by minimizing the average distortion D in equation (5.7).

$$D = E[X - \hat{X}]^2 = \int_{-\infty}^{\infty} f_x(x)(x - \hat{x})^2 dx \quad (5.7)$$

Using equation (5.7) the L-step optimal scalar quantizer can be written as

$$D = \sum_{i=1}^L \int_{x_{i-1}}^{x_i} f_x(x)(x - r_i)^2 dx \quad (5.8)$$

The minimization of D is

$$\begin{cases} \frac{\partial D}{\partial r_i} = 0, & 1 \leq i \leq L \\ \frac{\partial D}{\partial x_i} = 0, & 1 \leq i \leq L \\ x_0 = -\infty \\ x_L = \infty \end{cases} \quad (5.9)$$

When the equations (5.8) and (5.9) are solved, the decision levels x_i and reconstruction levels r_i are obtained as

$$\begin{cases} r_i = \frac{\int_{x_{i-1}}^{x_i} x f_x(x) dx}{\int_{x_{i-1}}^{x_i} f_x(x) dx}, & 1 \leq i \leq L \\ x_i = \frac{r_i + r_{i+1}}{2}, & 1 \leq i \leq L-1 \\ x_0 = -\infty \\ x_L = \infty \end{cases} \quad (5.10)$$

Since the Laplacian distribution $f_x(x) = \frac{1}{\sqrt{2}\sigma} e^{-\frac{\sqrt{2}}{\sigma}|x|}$ is symmetric, then it is enough to design the Lloyd-Max quantizer for positive values only and use the corresponding negatives for negative X . Therefore, the goal is to implement the Lloyd-Max quantizer which satisfies

$$\left\{ \begin{array}{l} r_i = \frac{\sigma \left(x + \frac{1}{\sqrt{2}} \right) e^{-\sqrt{2}x} \frac{x_i}{\sigma} - \frac{x_{i-1}}{\sigma}}{e^{-\sqrt{2}x} \frac{x_i}{\sigma} - \frac{x_{i-1}}{\sigma}}, \quad 1 \leq i \leq L \\ x_i = \frac{r_i + r_{i+1}}{2}, \quad 1 \leq i \leq L-1 \\ x_0 = 0 \\ x_L = \infty \end{array} \right. \quad (5.11)$$

to all subbands of DWT except the lowest subband, where σ is the standard deviation corresponding to that subband. In practice, equation (5.11) can be solved by an iterative scheme such as the Newton method.

In order to design a $2L$ step Lloyd-Max quantizer for a Laplacian distribution, we need to have S refining intervals. The relationship between L and S is

$$\frac{(S+1)(S+2)}{2} = L \quad (5.12)$$

Figure 5.2 shows an example of 10-step Lloyd-Max quantizer on the positive side of the coefficients. Since $L=10$ then $S=3$ refining intervals are required.

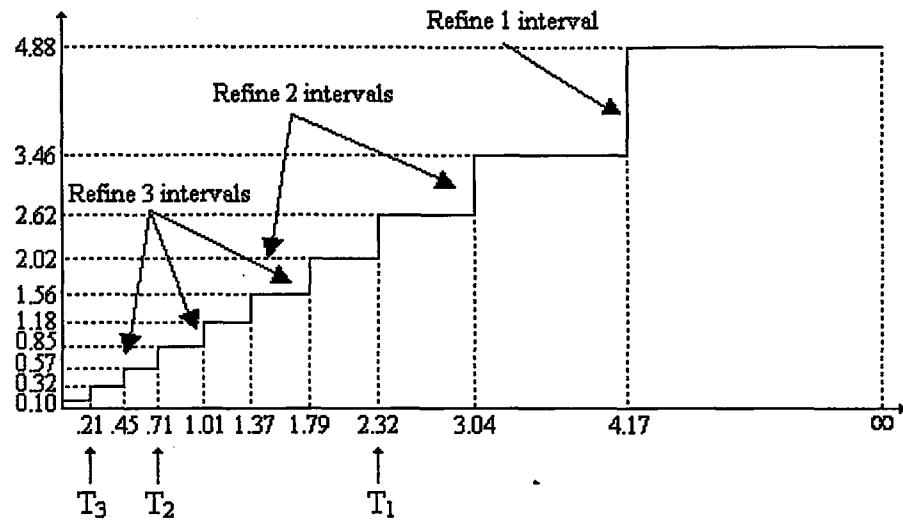
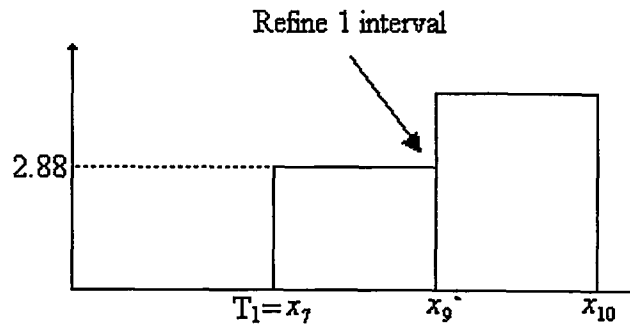


Figure 5.2 20-step Lloyd-Max quantizer. 10 steps on the positive side are shown. Negative side is symmetric.

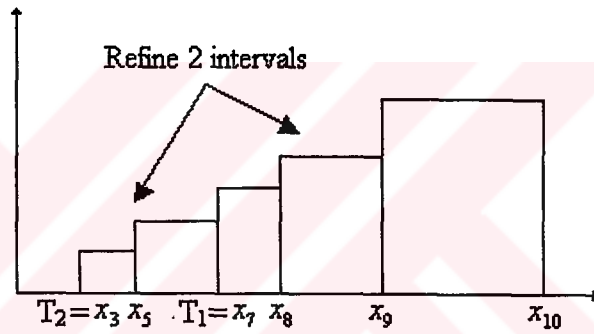
Figure 5.2 also shows the important checking thresholds and the refining thresholds. Figure 5.3 explains the Lloyd-Max algorithm in detail and shows the refining thresholds. Since the $S=3$ in this figure, the third stage is the last stage of the Lloyd-Max quantizer shown in Figure 5.3(c) and shows the final optimum quantizer step sizes. In the first stage shown in Figure 5.3(a), $x_7=2.32$ is defined as the initial threshold and $x_9=4.17$ as the refining threshold to do the important checking and the refinement on the interval $[2.32, \infty)$, respectively. In order to remind the decoder, one bit is sent to indicate if the important wavelet coefficient is in the upper level or the lower level of the interval. Therefore the interval $[2.32, \infty)$ is refined to two intervals, $[2.32, 4.17)$ and $[4.17, \infty)$. The reconstructed value for those important wavelet coefficients in the interval $[4.17, \infty)$ is $r_{10}=4.88$. For the important wavelet coefficients in the interval $[2.32, 4.17)$ the equation

$$\frac{\int_{2.32}^{4.17} x f_x(x) dx}{\int_{2.32}^{4.17} f_x(x) dx} = \frac{\left(x + \frac{1}{\sqrt{2}} \right) e^{-\sqrt{2}x} \Big|_{2.32}^{4.17}}{e^{-\sqrt{2}x} \Big|_{2.32}^{4.17}} = 2.88$$

is used to determine the reconstructed value. Similarly the same operation is applied to second and third stages. Figure 5.3(b) and figure 5.3(c) show the details of the operations for the stage 2 and stage 3 respectively.



(a)



(b)

Figure 5.3 The refining algorithm for 3 stages of Lloyd-Max quantizer (a) Refine 1 interval $[x_7, x_{10})$ and take x_9 as the refining threshold in the first stage. (b) Take x_5 and x_8 as the refining thresholds and refine the intervals $[x_3, x_7)$ and $[x_7, x_9)$ in the second stage. (c) Take x_2 , x_4 and x_6 as the refining thresholds and refine the intervals $[x_1, x_3)$, $[x_3, x_5)$ and $[x_5, x_7)$ in the third stage.

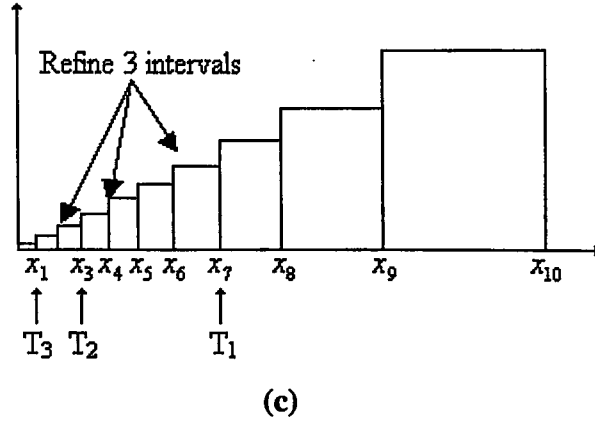


Figure 5.3 The refining algorithm for 3 stages of Lloyd-Max quantizer (a) Refine 1 interval $[x_7, x_{10})$ and take x_9 as the refining threshold in the first stage. (b) Take x_5 and x_8 as the refining thresholds and refine the intervals $[x_3, x_7)$ and $[x_7, x_9)$ in the second stage. (c) Take x_2 , x_4 and x_6 as the refining thresholds and refine the intervals $[x_1, x_3)$, $[x_3, x_5)$ and $[x_5, x_7)$ in the third stage.

In order to design the Lloyd-Max quantizer for each wavelet layer, the standard deviations for all wavelet layers have to be known except the lowest frequency subband. Therefore the encoder has to calculate the standard deviation for those wavelet layers and obtains the Lloyd-Max quantizer for each layer based on the required number of stages. Thus, the number of stages and the standard deviations belong to each wavelet layer have to be sent to the decoder as side information. In fact, we do not need to design the Lloyd-Max quantizer for each wavelet layer with different standard deviations. Because the Lloyd-Max quantizer for the Laplacian distribution with standard deviation σ is $\sigma \times$ (Lloyd-Max quantizer for the Laplacian distribution with unit standard deviation).

It is required to modify the significance map coding to adapt to those wavelet layers with different standard deviations. The significance checking is done by comparing the magnitude with the thresholds

$$T = \sigma_m T_s, m = 2, \dots, M, s = 1, \dots, S \quad (5.13)$$

for the coefficients from the wavelet layer m at stage s .

5.3 Vector Quantization

Image data compression using vector quantization (VQ) has received a lot of attention in the last twenty years because of its simplicity and adaptability (Gray, 1984) (Idris & Panchanathan, 1997). Vector quantization requires the input image to be processed as vectors or blocks of image pixels. The encoder takes a vector and finds the best or closest match, based on some distortion criterion, from its stored codebook. The address of the best match is then transmitted to the decoder. The decoder takes this address number and finds the corresponding vector from its codebook. The reconstructed image is obtained. Data compression is achieved in this process because the transmission of the address requires fewer bits than transmitting the vector itself.

The performance of encoding and decoding by vector quantization is dependent on the available codebook and the distribution of the image data relative to it. Hence, the design of an efficient and robust codebook is very important in vector quantization. Linde, Buzo and Gray first suggested a practical suboptimal clustering analysis algorithm, known as the LBG algorithm (Lin & Tai, 1998) to generate a codebook based on some training set. It is said that the algorithm only guarantees a locally optimum codebook relative to the source data used (the training set). The simulated annealing (SA) method of generating a codebook tries to obtain a global optimum by a stochastic relaxation technique. In vector quantization N -dimensional data space is divided into L regions called Voronoi regions. Figure 5.4 shows an example of such regions for the dimension of $N=2$.

The size of codebook and the vector dimension also play a major role in determining the overall performance. From Shannon's rate distortion theory we know that the larger the vector dimension, the better the potential performance. However, with increased vector dimension the required codebook size also increases, and the result is an exponential increase in encoding complexity. So for practical limitations one is forced to work with low-dimensionality vector quantization with less quality, despite the fact that better vector quantization performance is

theoretically possible. The increase in codebook size also introduces the problem of empty cells in the generated codebook. Empty cells makes the codeword zero.

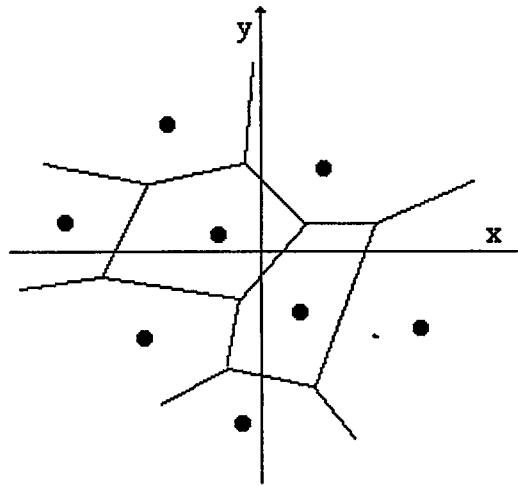


Figure 5.4 Voronoi regions

5.3.1 Codebook design by LBG algorithm

A vector quantizer can be defined as a mapping Q of K -dimensional Euclidean space R^k into a finite subset Y of R^k . Thus,

$$Q: R^k \rightarrow Y \quad (5.14)$$

where $Y = (\hat{x}_i; i = 1, 2, 3, \dots, N)$ is the set of reproduction vectors and N the number of vectors in Y . It can also be seen as a combination of two functions : an encoder, which views the input vector x and produces the address of the reproduction vector specified by $Q(x)$, and a decoder, which uses this address to produce the reproduction vector \hat{x} . If a distortion measure $d(x, \hat{x})$ which represents the penalty or cost associated with reproducing vectors x by \hat{x} is defined, then the best mapping Q is the one which minimizes the distortion. The LBG algorithm and other variations of this algorithm are based upon this minimization, using a training data set as the signal.

One simple distortion measure for waveform coding is the square error distortion given by

$$d(x, \hat{x}) = \|x - \hat{x}\|^2 = \sum_{j=0}^{K-1} (x_j - \hat{x}_j)^2 \quad (5.15)$$

K is the number of x vectors.

The goal in designing a vector quantizer is to obtain a quantizer consisting of N reproduction vectors, such that it minimizes the expected distortion. If there is no other quantizer that can achieve the minimum expected distortion this condition can be provided. Lloyd proposed an iterative nonvariational method known as the design of scalar quantizers. For some years later, Linde, Buzo and Gray extended Lloyds' basic approach to the general case of vector quantizers and the training algorithm was called LBG algorithm.

Let the expected distortion be approximated by the expression

$$D(x, q(x)) = \frac{1}{N} \sum_{i=0}^{N-1} d(x_i, \hat{x}_i) \quad (5.16)$$

The LBG algorithm for a known distribution training sequence follows these rules :

1) Let N=number of levels ; distortion threshold $\varepsilon \geq 0$. Assume an initial N level reproduction alphabet \hat{A}_0 , and a training sequence $(x_j ; j = 0, 1, 2, \dots, n-1)$, and m= number of iterations, set to zero.

2) Given $\hat{A}_m = (y_i ; i = 1, 2, \dots, N)$, find the minimum distortion partition $P(\hat{A}_m) = (S_i ; i = 1, \dots, N)$ of the training sequence : $x_j \in S_i$ if $d(x_j, y_i) \leq d(x_j, y_l)$ for all l. Compute the average distortion.

$$D_m = D[\hat{A}_m, P(\hat{A}_m)] = \frac{1}{n} \sum_{j=0}^{n-1} \min_{y \in \hat{A}_m} d(x_j, y) \quad (5.17)$$

3) If $(D_{m-1} - D_m) / D_m \leq \varepsilon$, stop the iteration with \hat{A}_m as the final reproduction alphabet; otherwise continue.

4) Find the optimal reproduction alphabet $\hat{x}(P(\hat{A}_m)) = (\hat{x}(S_i); i = 1, 2, \dots, N)$ for $P(\hat{A}_m)$ where

$$\hat{x}(S_i) = \frac{1}{\|S_i\|} \sum_{j: x_j \in S_i}^m x_j \quad (5.18)$$

5) Set $\hat{A}_{m+1} = \hat{x}(S_i)$, increment m to $m+1$, and go to (2).

In the above iterative algorithm an initial reproduction alphabet \hat{A}_0 was assumed in order to begin the algorithm. There are several techniques to obtain the initial codebook. The simplest technique is to use the first widely spaced words from the training data. Linde, Buzo and Gray used a splitting technique where the centroid for the training sequence was calculated and split into two close vectors. The centroids or the reproduction vectors for the two partitions were then found. Each resulting vector was then split into two vectors and the above procedure was repeated until an N level initial reproduction vector was obtained. Splitting was performed by adding a fixed perturbation vector ε to each vector y_i generating two vectors $y_i + \varepsilon, y_i - \varepsilon$. Figure 5.5 shows the block diagram of simple vector quantization.

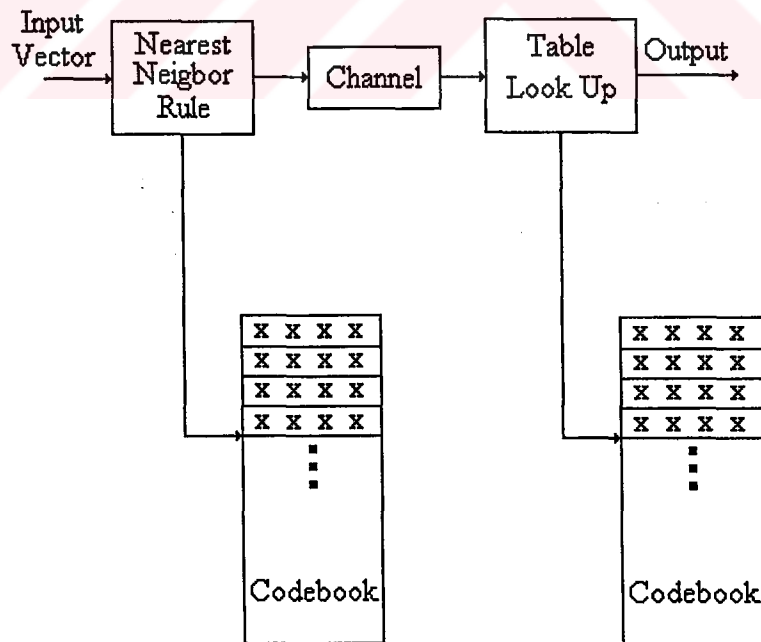


Figure 5.5 Block diagram of simple vector quantizer

5.4 Hierarchical Finite State Vector Quantization

The hierarchical finite state vector quantization (HFSVQ) (Yu & Venetsanopoulos, 1994) is a new technique which is developed on the basis of different VQ techniques. In the HFSVQ, an original image is divided to blocks of different sizes, which are then assigned into different layers, according to their variance. The blocks of small variance which are located in a smooth region of the image will have large sizes and be assigned into higher layers. Fewer codewords can be used in the higher layers since there are strong correlations between the pixels in the smooth regions. The blocks located in edge regions where the variance vary dramatically from pixel to pixel need large number of codewords to represent all of the different block types.

There are two parts in HFSVQ. One part consist of the structure codes which provide the data of layer assignment of the image blocks. The other part consists of the local address codes of the codewords in the codebook. It is proved that this coding scheme is more efficient than VQ and by using HFSVQ we achieve a further bit rate reduction.

5.4.1 Structure map construction

Any gray level image can be divided into several regions according to its variance. The regions with small variance correspond to the smooth area of the image and the regions which have edges with different sharpness demonstrate relatively grater variances. In this thesis, three types of layers are used. The higher layers have the small sizes of blocks that are obtained from the second or third wavelet transform decomposition and the lower layers have the bigger size of blocks that are obtained from the first wavelet transform decomposition.

First, the whole image is decomposed into a group of blocks of size 16×16 . If the variance in a block is lower than a given threshold, the block is assigned into layer 1 (L_1), which collects the blocks located in the smoothest regions of the image. Then the rest of the blocks are decomposed into subblocks of size 8×8 . The same threshold

is used and blocks located in a fairly smooth regions are assigned into layer 2 (L_2). Finally, we further decompose the remainder into blocks of size 4x4 and assign them to the edge layer, layer 3 (L_3). In order to obtain more perceptual result we need to provide a more accurate reconstruction. Therefore a special layer for edges, layer 4 (L_4) is constructed from layer 3. Layer 4 collects the blocks with large variance by using a threshold with higher value. Here, the variance is calculated by using whole pixels in a block

If the image or frame size is 512x512, then the procedure above is used. Otherwise, for example, if the image or frame size is 256x256 then the corresponding block sizes must be 8x8 for (L_1), 4x4 for (L_2) and 2x2 for (L_3) and (L_4).

After each layer has its own blocks, then the LBG rule is applied to the layers to find the codebook for each layer. There are some advantages for this subcodebook generation strategy :

1) Only layer members are involved in training for generation of the codebook. Both the iteration times and the number of comparisons in each iteration are considerably reduced. As a result, the total training time can be shortened.

2) By applying layer assignment, we can adjust the size of each codebook according to the accuracy requirement for reconstruction of the different regions of an image by either choosing the threshold values or changing the sizes of the codebook for specific layers.

In the application, small size of codeword list for layer1 and layer 2, medium size of codeword list for layer 3 and big size of codeword list for layer 4 are selected by the LBG algorithm. This means, we need less bits to represent the layer 1 and layer 2 that have big size of blocks and more bits are required for the the layer 3 and layer 4. Therefore a compression is achieved. Compression result are stored into the computer memory by the help of a structure tree shown in Figure 5.6. The

assignment result can be demonstrated as a structure map shown in Figure 5.7. In Figure 5.7 there are zeros and ones which are called structure codes. Structure codes are needed to record the structure map which is based on the branch distribution of the structure tree. The symbol “*” in Figure 5.7 represents the codeword address. This is inserted between the structure codes. By using the reconstruction algorithm the structure codes are decoded again by the help of the structure tree.

In this thesis the hierarchical finite state vector quantization algorithm is applied to discrete wavelet transformed and zero tree structured intra video frame or a inter error frame data. The zero tree structure will be described in the following chapter.

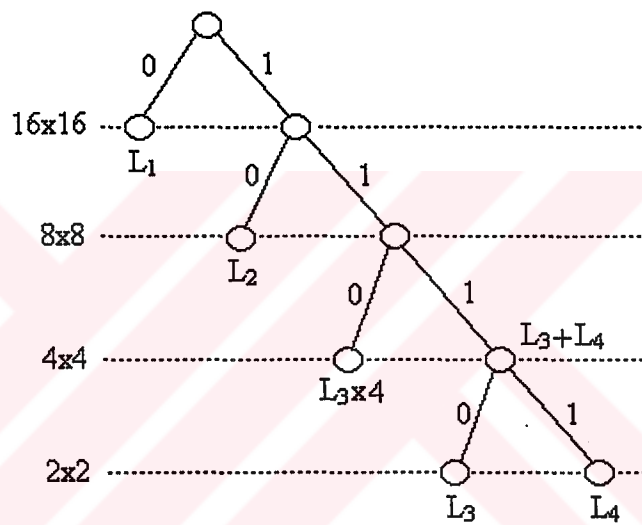


Figure 5.6 Structure Tree

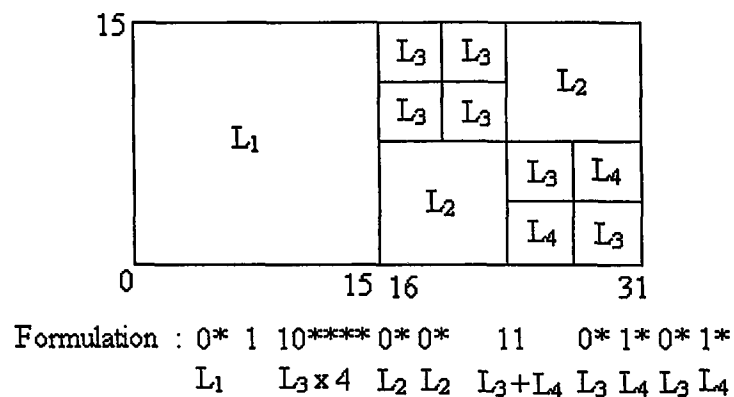


Figure 5.7 Structure Map

CHAPTER 6

A VIDEO COMPRESSION TECHNIQUE USING ZEROTREE WAVELET AND HFSVQ

6.1 Introduction

In this chapter a video compression technique with comparable low bit rates is presented. The proposed encoder is similar to other motion compensated, block based DCT video coding standards such as MPEG-1/2, H.261/3 and MPEG-4/VM (Rao & Hwang, 1996)(Sikora, 1997).

In this introduced encoder, the DWT is used instead of DCT, the OBMC (Auyeung et al., 1992) is added to the block motion estimation algorithm to match better DWT and for coding the wavelet coefficients, the zerotree wavelet structure is combined with the HFSVQ. DWT seems to have better performance than DCT for image and video coding (Yeung, 1997). DWT also provides us the scalability functionalities of image and video. The specific components of the coding algorithm as shown in Figure 6.1 are the block motion estimation algorithm to track the local motion, OBMC to remove the temporal redundancy, DWT to remove the spatial correlation and the Lloyd Max quantization (Lloyd, 1982) to quantize the wavelet coefficients.

The proposed algorithm is similar to the scheme introduced in (Martucci et al., 1997). The main difference is combining the zerotree wavelet structure with HFSVQ instead of arithmetic coding. It is important to note that, the zerotree structure consists of different size of wavelet blocks that contain similar informations. Because

of that it is very suitable for these blocks to be coded by HFSVQ algorithm. ZTW structure itself has the ability to control the bit rate. When the HFSVQ is applied to any ZTW structure, then each type of wavelet block will be represented by different size of codeword. That will give us the second opportunity of controlling the bit rate. In our proposed encoder, as a quantizer the Lloyd-Max and as a block motion estimation algorithm the New Diamond Search are used.

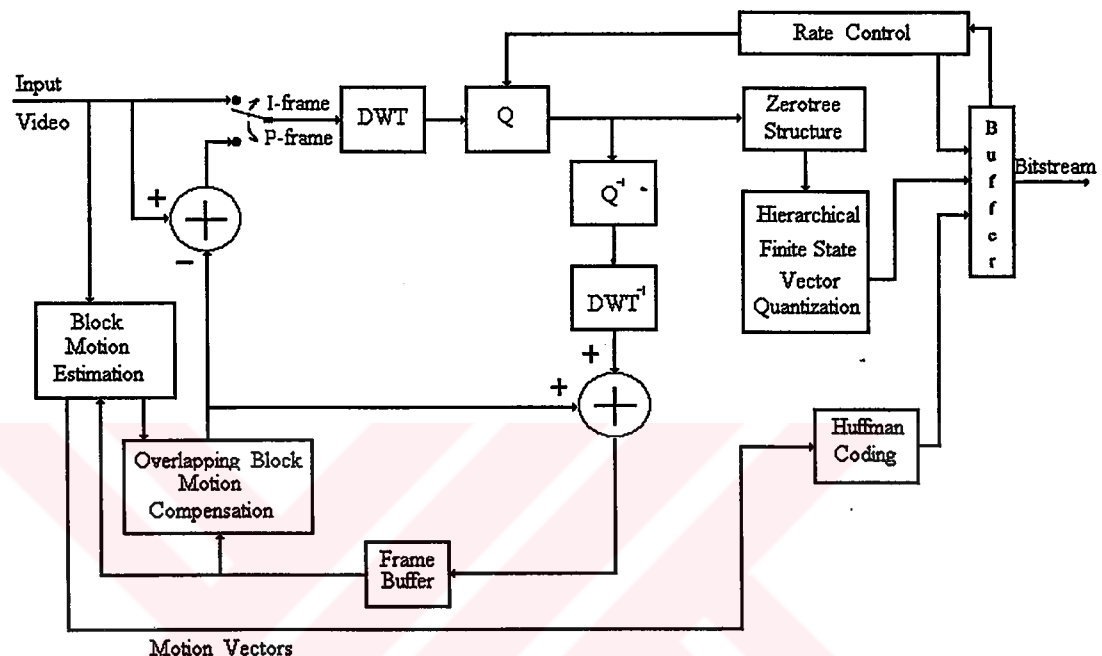


Figure 6.1 Block diagram of the proposed video encoder

The computer simulation of whole system is done by using the Object Oriented Borland C++ Builder software. The simulation results of the proposed encoder was taken using three types of standard video sequences; QCIF Akiyo which contains a few movements at the rates of 10 kb/s and 5 fr/s, QCIF Carphone which contains relatively more movements than Akiyo at the rates of 30 kb/s and 10 fr/s and QCIF Coastguard which contains more movements than the others at the rates of 48 kb/s and 7.5 fr/s. Average PSNR over the entire coded sequences show that the proposed ZTW-HFSVQ video compression technique achieves comparable performance over MPEG-4/VM, H.263, ZTE and EZW for both I and P frames.

The proposed encoder consists of NDS, Huffman coding, OBMC, multiresolution of DWT, ZTW structure, Lloyd-Max quantizer, HFSVQ and rate control structure. In this chapter, the combination of ZTW structure with the HFSVQ, which is the novel feature of this new introduced encoder is described in details.

6.2 Motion Estimation and Compensation

The proposed video coding algorithm uses the block motion estimation technique called New Diamond Search (NDS) (Zhu & Ma, 2000). As explained in chapter 3, the NDS performs better than the other well known block based search algorithms such as tree step search, the logarithmic search, the novel four step search, the simple and efficient search and the one at a time search both in reaching the lowest mean square error and achieving a quite small average search point numbers. This work which is the part of this thesis is published in (Kılıç & Yılmaz, 2001). In block motion estimation algorithm as a performance criteria, the mean square error is used. Motion estimation is performed on the luminance 16x16 blocks. The search area is constrained to 7 pixels in all four directions from the center of the macroblock. The motion vectors of the macroblocks are Huffman encoded.

Overlapping block motion compensation (OBMC) used in our proposed encoder is an advanced scheme which is used by H.263 (Auyeung et al., 1992). The OBMC algorithm overlaps, windows, and sums prediction blocks in order to reduce the effect of block boundary discontinuities. In this method, each 8x8 block is overlapped with two major neighboring blocks : 1) the upper left 8x8 block of each 16x16 macroblock is overlapped with the adjacent blocks located at the above and left sides; 2) the upper right 8x8 block of each 16x16 macroblock is overlapped with the adjacent blocks located at the above and right sides; 3) the lower left 8x8 block of each 16x16 macroblock is overlapped with the adjacent blocks located at the below and left sides; 4) the lower right 8x8 block of each 16x16 macroblock is overlapped with the adjacent blocks located at the below and right sides. Therefore, each pixel of motion compensated frame is a weighted sum of three prediction values from the previous reconstructed frame: one value predicted using the current block motion

vector and two other values predicted using the neighboring motion vectors. This overlapping algorithm provides a coherent motion compensated frame and therefore, a coherent motion residual frame which is free of artificial block discontinuities, otherwise a nonoverlapping motion compensation scheme can naturally produce some block discontinuities. Note that intra blocks are not overlapped with their neighboring blocks.

The visual examples of block motion estimation by NDS algorithm and the OBMC result are given in Figure 6.2 for the 48th frame of Akiyo. In this figure the encoder uses the standard QCIF Akiyo at the rates of 5 f/s and 10 kb/s. The reconstructed 42nd frame is used as an original past frame by the encoder in order to obtain the 48th current frame of Akiyo. The MSE of the 42nd original past frame is 16.1. After the NDS algorithm is applied, the reconstructed 48th frame has a MSE value of 30.22. In order to decrease block effects and MSE value, the OBMC algorithm is applied to the reconstructed 48th frame of Akiyo. Then the error decreases to MSE=24.43 which is suitable value for the error frame coding. Here, 68 % of the error macroblocks are the zerotrees, therefore the compression ratio is high and 0.048 bpp.

The probability distributions of motion vector differences, which are obtained by using motion vectors of reconstructed 48th frame are given in Table 6.1. The motion vector differences are Huffman encoded. The symbol codes which represents specific the motion vector differences are given in Table 6.2 (Rao & Hwang, 1996). The total symbol code of the motion vector differences is 32 bits. That can be seen from Table 6.1 and Table 6.2. The total Huffman code belongs to the motion vectors is 117 bits. The bit requirement for encoding the 48th error frame is 1360 bit. Therefore in order to encode the 48th interframe of Akiyo, the encoder requires totally 1501 bits.

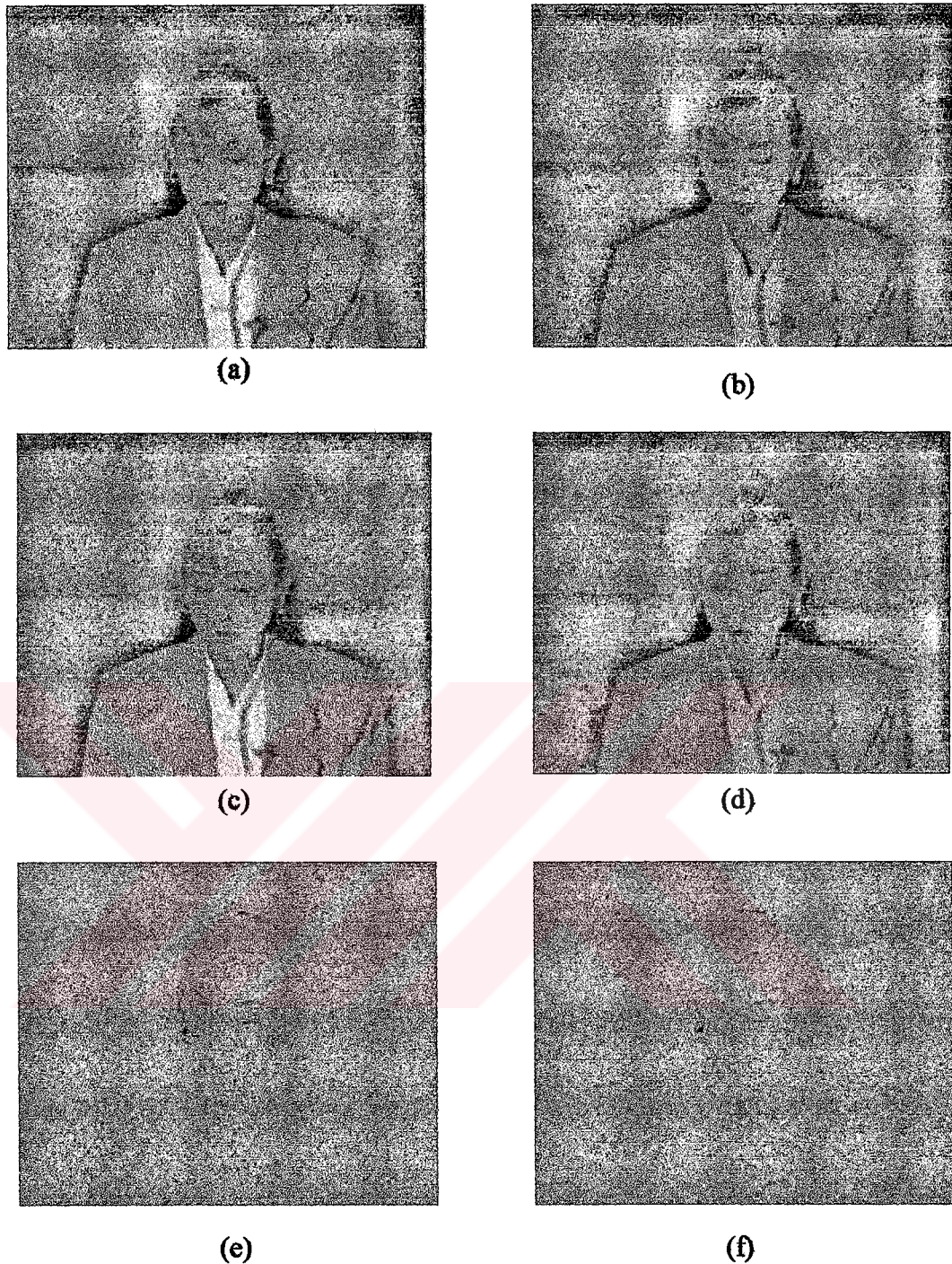


Figure 6.2 The encoder results by using OCIF Akiyo sequence at the rates of 5 f/s and 10 kb/s (a) Reconstructed 42nd frame with MSE=16.1 (b) Original 48th frame (c) Reconstructed 48th frame by NDS with MSE=30.22 (d) Reconstructed 48th frame by OBMC after the NDS with MSE=24.43 (e) Reconstructed 48th error frame after OBMC (f) Reconstructed 48th error frame after error coding with MSE=14.6.

Table 6.1 Probabilities of motion vector differences and corresponding Huffman codes for reconstructed 48th frame of Akiyo

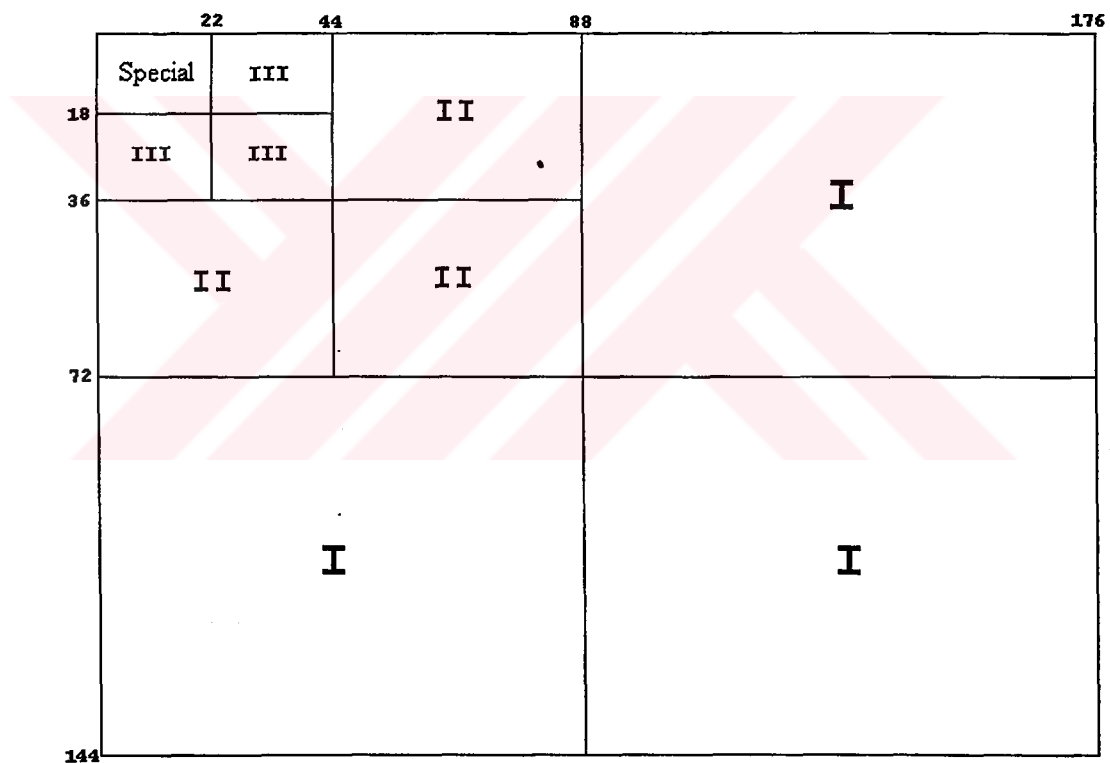
Numbers of Symbols	Difference on the X axis	Difference on the Y axis	Probabilities	Corresponding Huffman Codes
1	0	0	0.82828	0
2	0	-1	0.10101	1
3	1	-1	0.0303	110
4	-2	2	0.0101	1010
5	-1	0	0.0101	1011
6	0	1	0.0101	1000
7	1	0	0.0101	1001

Table 6.2 H.261 VLC Table of motion vector differences

Motion Vector Difference	VLC Code	Motion Vector Difference	VLC Code
-16, 16	0000 0011 001	0	1
-15, 17	0000 0011 011	1	010
-14, 18	0000 0011 101	2, -30	0010
-13, 19	0000 0011 111	3, -29	0001 0
-12, 20	0000 0100 001	4, -28	0000 110
-11, 21	0000 0100 011	5, -27	0000 1010
-10, 22	0000 0100 11	6, -26	0000 1000
-9, 23	0000 0100 01	7, -25	0000 0110
-8, 24	0000 0101 11	8, -24	0000 0101 10
-7, 25	0000 0111	9, -23	0000 0101 00
-6, 26	0000 1001	10, -22	0000 0100 10
-5, 27	0000 1011	11, -21	0000 0100 010
-4, 28	0000 111	12, -20	0000 0100 000
-3, 29	0001 1	13, -19	0000 0011 110
-2, 30	0011	14, -18	0000 0011 100
-1	011	15, -17	0000 0011 010

6.3 Discrete Wavelet Transform of QCIF Video Frames

The wavelet transform performs decomposition of video frames or motion compensated residuals into a multiresolution subband representation. An important feature of our implementation is the ability to use different filter banks at each level of the decomposition. This is important because longer filters provide good frequency localization but can cause ringing artifacts along the edges of objects while the use of shorter filter banks cause less ringing artifacts but more blockiness in the reconstructed frame. Therefore the Biorthogonal 9.3 filter bank (Stang & Nguyen, 1996) is used at the first level of the DWT decomposition followed by the Haar filter for the remaining two levels.



In order to explain how the DWT is applied to QCIF video frame denoted by matrix A , let's assume that the scaling function and the wavelet function filters are represented by H and G respectively. The DWT coefficients can be calculated by inner products that are given in the equations (4.30) and (4.31) by using wavelet and the scaling function coefficients. The process of the DWT decomposition can be explained as follows. On the rows of the matrix A the filters H and G are applied. The two resulting matrices are obtained; $A_h = H_r A$ and $A_g = G_r A$, both of dimensions 88×144 (Symbols r and c denote that the filters are applied row-wise and column-wise respectively to the A). Now on the columns of A_h and A_g , filters H and G are applied again and the four resulting matrices $A_{hh} = H_c H_r A$, $A_{hg} = G_c H_r A$, $A_{gh} = H_c G_r A$ and $A_{gg} = G_c G_r A$ of dimension 88×72 are obtained. The matrix A_{hh} is the low frequency subband, while the matrices A_{hg} , A_{gh} and A_{gg} are the high frequency subbands. This decomposition process is shown in Figure 6.4.

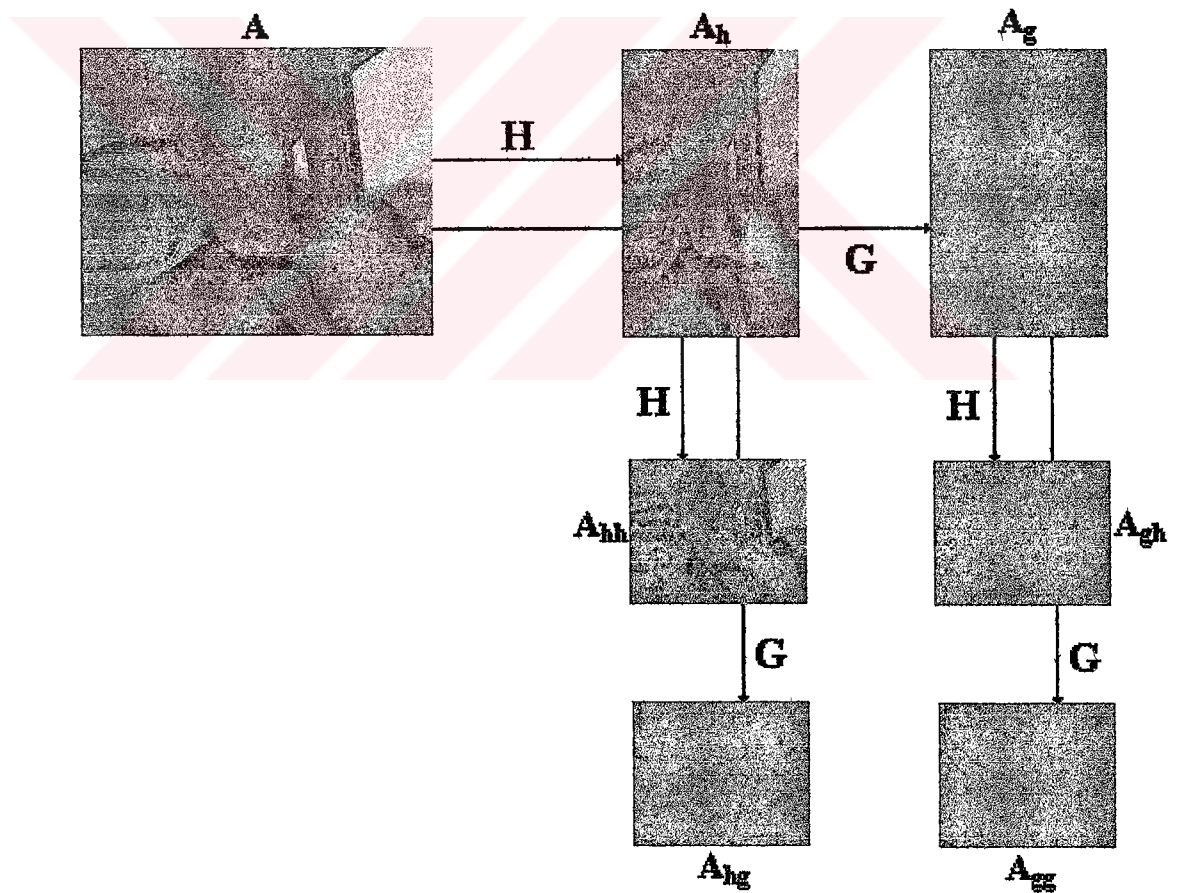


Figure 6.4 The first layer of the DWT decomposition of a Carphone video frame

The H and G operators are applied pixel couple by pixel couple on the rows and columns of A. Let's assume that a_n , $n=1, 2, \dots, 176$ is a row sequence of QCIF video frame, h_{Bi} , $i=1, 2, \dots, 9$ and g_{Bi} , $i=1, 2, 3$ are the Biorthogonal (9.3), h_{Hi} , $i=1, 2$ and g_{Hi} , $i=1, 2$ are the Haar scaling and wavelet function coefficients respectively, which are shown in Table 4.1. The $b_j=H(a_i)$ and $c_j=G(a_i)$ are the j^{th} result of the Biorthogonal (9.3), $d_j=H(a_i)$ and $e_j=G(a_i)$ are the j^{th} result of Haar scaling (H) and wavelet (G) operators respectively for any row sequence.

The H and G operators are applied every row and column sequence of a_n . The first level DWT application formulations using Biorthogonal (9.3) for a row sequence are shown in Equation (6.1). The same formulations are applied to the columns of A_h and A_g matrices in order to obtain A_{hh} , A_{gh} , A_{hg} and A_{gg} .

$$\begin{aligned} b_{j+1} &= a_m (h_{B1} + h_{B3} + h_{B5} + h_{B7} + h_{B9}) + a_{m+1} (h_{B2} + h_{B4} + h_{B6} + h_{B8}) \\ c_{j+1} &= a_m (g_{B1} + g_{B3}) + a_{m+1} g_{B2} \quad \dots (m = 2j + 1, j = 0, 1, 2, \dots, 87) \end{aligned} \quad (6.1)$$

The second and third level DWT decompositions are obtained by applying the same procedure on A_{hh} and A_{hhhh} matrices by using Haar function. The second and third level DWT application formulations for a row sequence are shown in Equation (6.2).

$$\begin{aligned} d_{j+1} &= a_m h_{H1} + a_{m+1} h_{H2} \\ e_{j+1} &= a_m g_{H1} + a_{m+1} g_{H2} \quad \dots (m = 2j + 1, j = 0, 1, 2, \dots, 87) \end{aligned} \quad (6.2)$$

Note that, the wavelet and scaling function coefficients are orthogonal as given by Equation (4.3). Therefore the same procedure described above can be used in the computations of inverse DWTs.

6.4 Lloyd-Max Quantization of Wavelet Coefficients

Lloyd-Max quantization algorithm (Yin, 2000) is a nonuniform quantization process. The distributions of the wavelet coefficients of any video frames are not uniform. Except for the lowest frequency coefficients, the histograms of all remaining wavelet transform layers have a nearly zero symmetric Laplacian distribution that is shown in Figure 6.5.

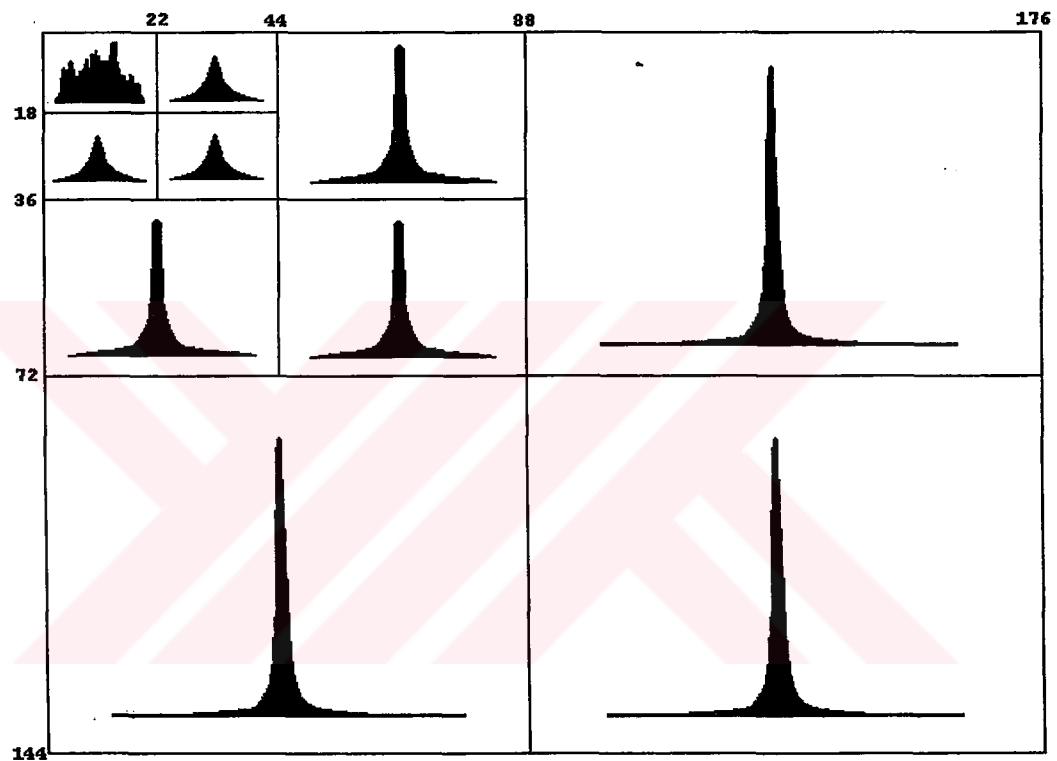


Figure 6.5 Histogram for three levels DWT of a QCIF video frame

Lloyd-Max quantization process follows the Gaussian curve and finds an optimum number of nonuniform quantization intervals depending on the standard deviation criteria. In our proposed encoder only the coarsest subband is uniformly quantized and the rest of the subbands are nonuniformly quantized by using Lloyd-Max algorithm. At the end of nonuniform quantization process each standard deviation and the number of quantization intervals for each subband are sent as side

information. Note that the Lloyd-Max algorithm is applied to entire wavelet coefficients in a subband except deadzone area in which all amplitudes are too small and set to zero by a predefined threshold value before the process. The application area of Lloyd-Max quantization algorithm is shown in Figure 6.6.

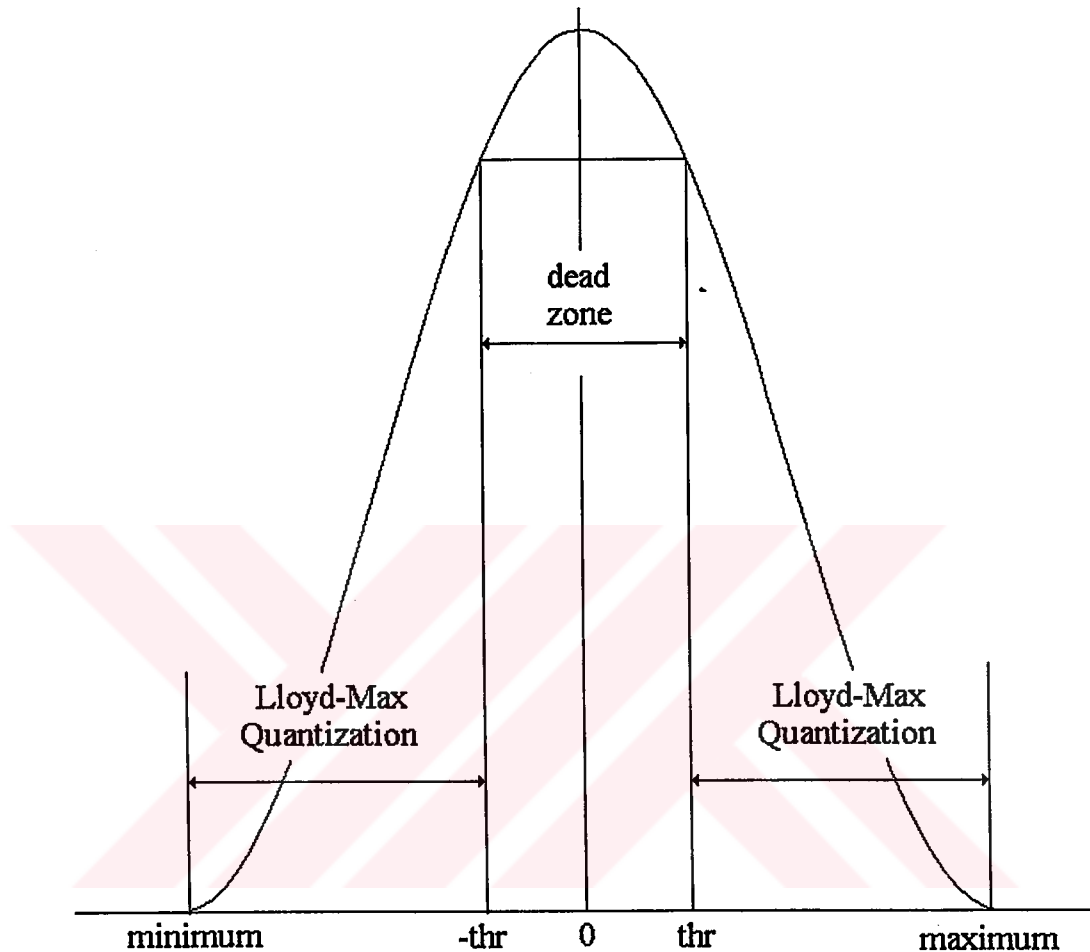


Figure 6.6 Deadzone and Lloyd-Max quantization locations in a wavelet transform layer. Horizontal scale is the wavelet coefficient amplitudes, vertical scale is the number of wavelet coefficients.

The wavelet coefficients belong to high frequency layers have both narrower bands and smaller amplitudes, but in low frequency layers the wavelet coefficients have both more wide bands and bigger amplitudes. As an example, the Biorthogonal

9.3 DWT is applied to the 256x256 Lena image and the maximum, minimum coefficients for different layers are shown in Table 6.3

Table 6.3 The maximum and minimum Biorthogonal 9.3 DWT coefficient values for layers of 256x256 LENA image

	Layer Special	Layer 6	Layer 5	Layer 4	Layer 3	Layer 2	Layer 1
Minimum	5550.2	-2111.9	-1517.9	-727.3	-419.6	-302	-168
Maximum	10266.6	1516.1	1433.9	779.9	426.1	222.7	137

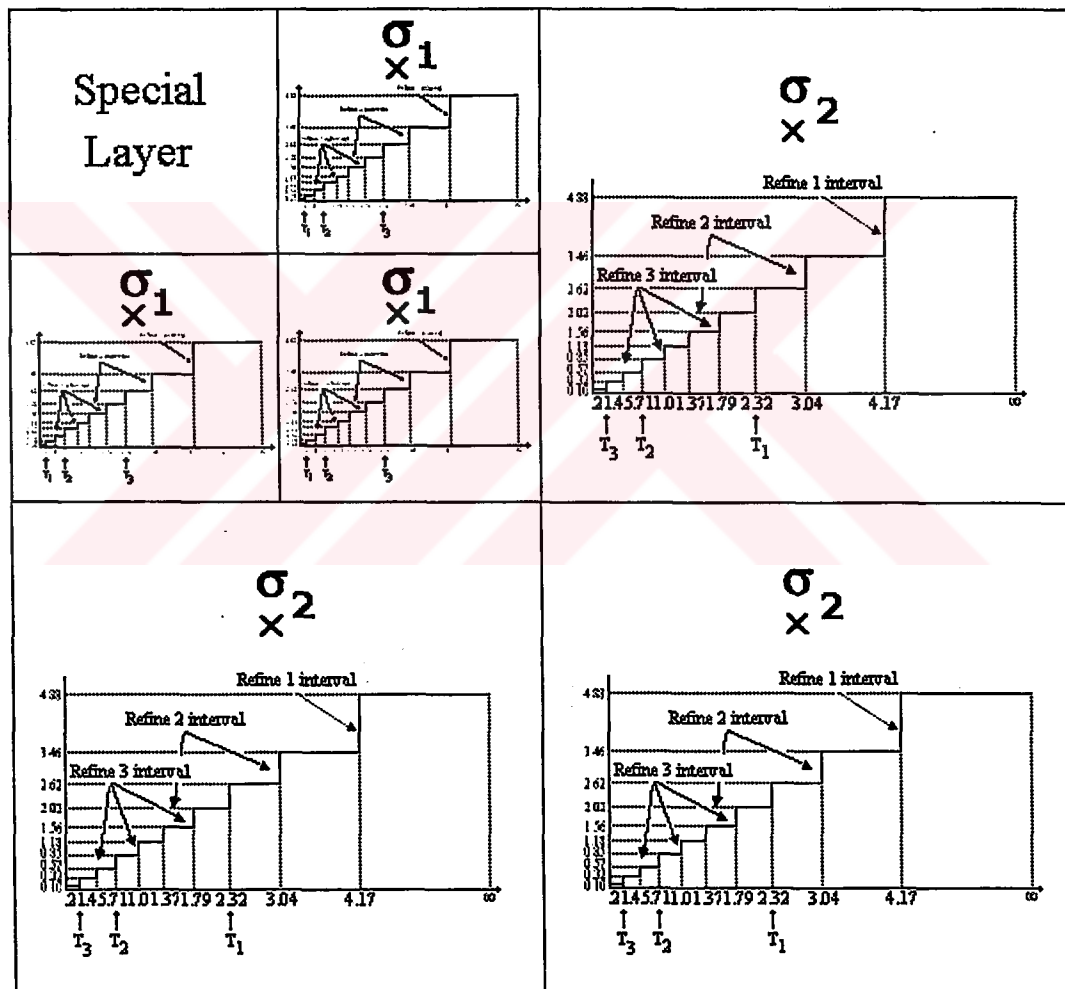


Figure 6.7 Lloyd-Max illustration of a video frame for two levels DWT.

In Figure 6.7, the Lloyd-Max quantization representation is shown for each layer of a video frame. In this figure, only the two levels of DWT structure is shown for Lloyd-Max quantization illustration. Note that each layer has its own standard deviation that is sent as a side information by the encoder.

6.5 Zerotree Structure

The DWT decomposes the input video frame into a set of subbands of varying resolutions (Rogers & Cosman, 1998). The coarsest subband is a lowpass approximation of the original frame, and the other subbands are finer-scale refinements. In the hierarchical subband system such as that of the wavelet transform, with the exception of the highest frequency subbands, every coefficient at a given scale can be related to a set of coefficients of similar orientation at the next finer scale (Averbuch et al., 1996). The coefficient at the coarse scale is called the parent, and all coefficients at the same spatial location and of similar orientation at the next finer scale are called children. As an example, Figure 6.8 shows a wavelet tree descending from a coefficient in the subband. For the lowest frequency subband, in the example, the parent-child relationship (Martucci et al., 1997) is defined such that each parent node has three children, one in each subband at the same scale and spatial location but different orientation.

Zerotree structure is built on the parent-child relationship (Lo et al., 2000), (Wang & Ghanbari, 1997). The zerotree structure takes advantage of the principle that if a wavelet coefficient at a coarse scale is insignificant (quantized to zero) with respect to a given threshold, then all wavelet coefficients of the same orientation at the same spatial location at finer wavelet scales are also likely to be insignificant with respect to that. The zerotree structure is similar to the zigzag scanning and end-of-block symbol commonly used in coding DCT coefficients.

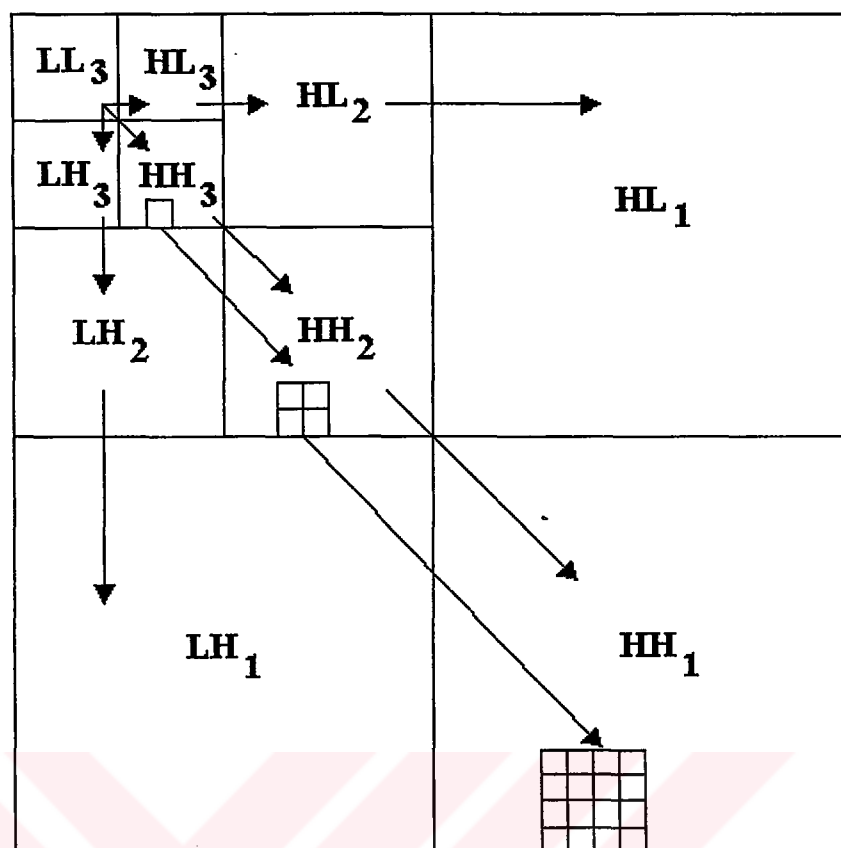


Figure 6.8 Parent-child relationship for wavelet coefficients.

In zerotree structure, the coefficients of each wavelet tree are reorganized to form a wavelet block as shown in Figure 6.9. Each wavelet block comprises those coefficients at all scales and orientations that correspond to the frame at the spatial location of that block. The concept of the wavelet block provides an association between wavelet coefficients and what they represent spatially in the frame. Quantization of the wavelet transform coefficients can be done prior to the construction of the wavelet tree, as a separate task, or quantization can be incorporated into the wavelet tree construction.

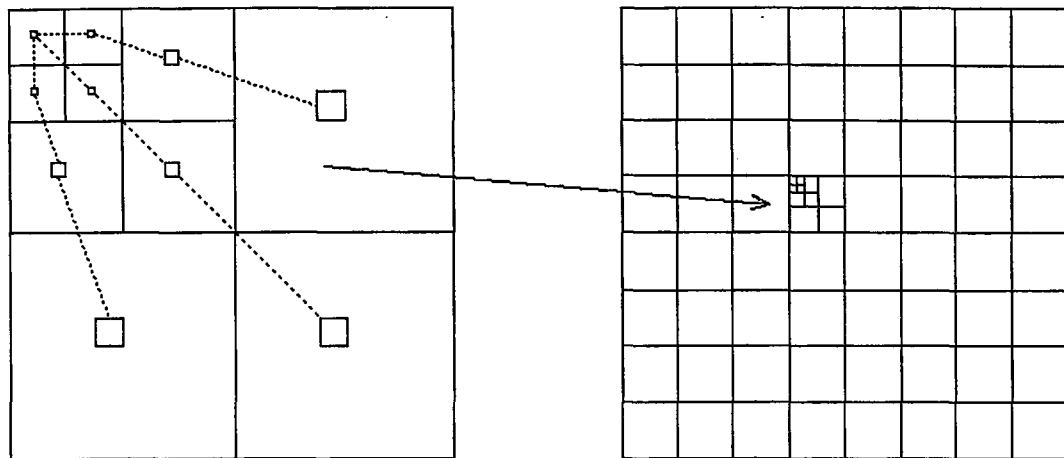


Figure 6.9 Reorganization of a wavelet tree into a ZeroTree wavelet block.

In the zerotree structure if all coefficients of any parent block are zero or nearly zero value then all other children blocks of that zerotree are unimportant. That means we can easily represent that zerotree with the value of zero.

Zerotree structure is a new efficient technique, for coding wavelet transform coefficients of motion-compensated video residuals or of video frames. This new zerotree structure provides us to reach the same wavelet information belongs to different frequencies through the DWT layers. Zerotree structure organizes wavelet coefficients into wavelet trees and then uses zerotrees to reduce the number of bits required to represent those trees. Zerotree structure has these advantages: 1) quantization can be performed distinct from the zerotree growing process or can be incorporated into the process thereby making it possible to adjust the quantization according to where the transform coefficient lies and what it represents in the frame; 2) coefficient scanning, tree growing, and coding can be done in one pass instead of bit-plane-by-bit-plane; 3) coefficient scanning is changed from subband-by-subband.

Since a zerotree collects different size of blocks then it is very suitable for coding those blocks by a variable block size coding algorithm. Therefore in my proposed coding algorithm hierarchical finite state vector quantization algorithm is used to encode the zerotrees.

6.6 HFSVQ and Rate Control

Hierarchical finite state vector quantization is an advanced algorithm of vector quantization that uses different size of blocks in the processing of an image. A zerotree consists of variable block sizes of the similar information that belongs to different layers. The proposed encoder produces the codebooks and codewords of each subband using LBG algorithm (Lin & Tai, 1998). The similar sizes of wavelet blocks are used to produce a codebook list that belong to that DWT layer. The amplitudes and the bandwidths of the wavelet coefficients decrease dramatically from the coarsest low frequency subband to the high frequency subbands. Therefore the size of the codebooks for the low frequency subbands are defined bigger than the codebooks for the higher frequency subbands. Hence, the big children blocks are represented by a small number of bits which increases the coding efficiency. In the proposed encoder, the standard QCIF video frames are used and three DWT is applied to each video or error frame. Therefore three types of wavelet blocks are used in a zerotrees structure, which are 2x2, 4x4 and 8x8 pixels sizes. The possible block sizes for each DWT layers and the number of zerotree blocks are shown in Table 6.4.

Table 6.4 For a 176x144 QCIF Frame, possible block sizes and number of zerotree blocks in each level of DWT decomposition

DWT Level	Possible Corresponding Block Sizes for Each Level					TOTAL Zerotree Block In The Frame
	V	IV	III	II	I	
I	-	-	-	-	1x1	6336
	-	-	-	-	2x2	1584
	-	-	-	-	4x4	396
	-	-	-	-	8x8	99
II	-	-	-	1x1	2x2	1584
	-	-	-	2x2	4x4	396
	-	-	-	4x4	8x8	99
III	-	-	1x1	2x2	4x4	396
	-	-	2x2	4x4	8x8	99

Since each QCIF error frame contains a few amount of wavelet coefficients small sizes of codebooks are defined for each DWT level. The number of codeword requirements for each DWT level of an error frame is, 4-16 for level 1, 8-32 for level 2 and 16-32 for level 3. The Intra-frame coding requires more codewords than the error frames do. The number of codeword requirements for each DWT level of an intra-frame is, 8-32 for level 1, 16-64 for level 2 and 32-128 for level 3. Here level 1 represents the wavelet coefficients of highest frequency and level 3 represents the wavelet coefficients of lowest frequency.

The HFSVQ coding tree and the block sizes are shown in Figure 6.10. All the codebooks that belong to DWT layers are Huffman encoded.

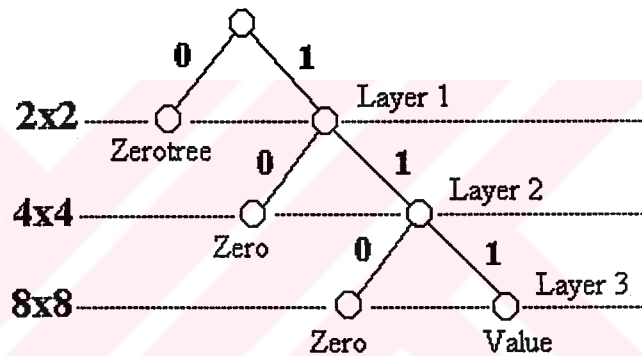


Figure 6.10 The HFSVQ block sizes and coding tree that are used by the encoder

In order to obtain the Zerotree blocks for the HFSVQ, minimum sizes of block (2x2) is defined in the highest DWT decomposition. The zerotree structure collects the double sizes of corresponding blocks in the lower DWT decompositions. For example, if 2x2 blocks are taken from the the highest DWT decomposition level of a QCIF frame, that is third level, then the corresponding block which contains the similar information in the second level must be in 4x4 pixel size.

The advantages of for this zerotree subcodebook generation strategy. 1) Only layer members are involved in training for generation of the codebook. Both the iteration times and the number of comparisons in each iteration are considerably

reduced. As a result, the total training time is shortened. 2) By applying layer assignment, the size of each codebook is adjusted according to the accuracy requirement for reconstruction of the different regions of a video frame.

In the encoding process if the coefficients of a parent block in the coarsest subband are zero then this is defined as a zerotree and encoder passes to scan an other zerotree. If the coefficients of any subband blocks are zero then quadtree scan gives zero to this layer and changes the layer. If the blocks have no zero information then corresponding adress from the codebook is used instead of the original block. The more zerotrees and zerolayers the less bits.encoder requires in the coding process.

The rate control is performed by the encoder in three possible mechanism: 1) increasing the number of the Lloyd-Max quantization intervals of any subband also increases the total bitrate and quality of the video frame. On the contrary decreasing it causes low quality and low bit rate (initial value is $S=3$), 2) increasing the bandwidth of the deadzone area of any subband causes a decrease both in the quality of video frame and total bit rate. On the other hand smaller deadzone areas of any subband increases the bitrate and the quality of the frame. Note that increasing or decreasing the parameters mentioned above are performed in all subbands, (initial area is 5% x maximum wavelet value), 3) increasing and decreasing the codebook sizes in each DWT level change the bit rate (initial codebook sizes for level 1=4, level 2=8, level 3=16)

6.7 Simulation Results

The proposed encoder is designed to encode I and P frames using diamond search block motion estimation, OBMC, Lloyd-Max quantization, DWT, Zerotree wavelet structure and the HFSVQ algorithm.

The experimental results of the proposed encoder was taken using the three components (Y, Cb, Cr) of the standart QCIF video sequences such as Akiyo,

Foreman, News, Carphone and Coastguard. We compared the proposed encoder with zerotree coding(ZTE), embedded zerotree wavelet coding(EZW) and with the block based MPEG-4 verification model (VM) (Sikora, 1997). The block based MPEG-4 is a slightly modified version of DCT-based H.263 coder.

In our first experiment, I frame coding results of ZTE, EZW, MPEG-4/VM and ZTW-HFSVQ were compared. The comparison was made at 14 kb, 27 kb and at 28 kb using the first frames of Akiyo, News, Foreman. The PSNR results for the luminance component (Y) and the average for the two chrominance components Cb and Cr labeled as C are shown in Table 6.5. The performance results of MPEG-4, EZW and ZTE shown in Table 6.5 were given in (Martucci et al., 1997). It is seen that ZTE gives better performance than MPEG-4/VM and EZW. On the other hand the proposed encoder seems to be better than the other three algorithms mentioned above both in luminance and chrominance for all test sequences.

Tablo 6.5 QCIF first I-Frame Coding PSNR Results in dB

QCIF First I Frames	Bits (kb)	Y C	MPEG-4 VM	EZW	ZTE	ZTW-HFSVQ
Akiyo	14	Y	33.06	33.77	34.62	34.62
		C	36.31	35.71	36.19	36.41
Akiyo	28	Y	38.42	40.18	40.18	40.82
		C	40.81	39.50	40.81	41.43
News	14	Y	28.60	29.38	29.38	29.42
		C	33.82	31.78	33.47	34.25
News	27	Y	33.38	34.68	34.49	35.13
		C	37.39	35.12	36.84	40.06
Foreman	14	Y	30.11	30.85	30.86	30.88
		C	38.27	37.57	38.69	39.42
Foreman	27	Y	35.05	35.49	35.27	36.16
		C	40.71	41.23	40.76	41.70

A comparison of H.263 Intra coding with ZTW-HFSVQ using the first frame of Akiyo QCIF is given in Figure 6.11. For high compression ratio the proposed algorithm is seems to have approximately 1 dB gain and for low compression ratio the gain is approximately 2 dB.

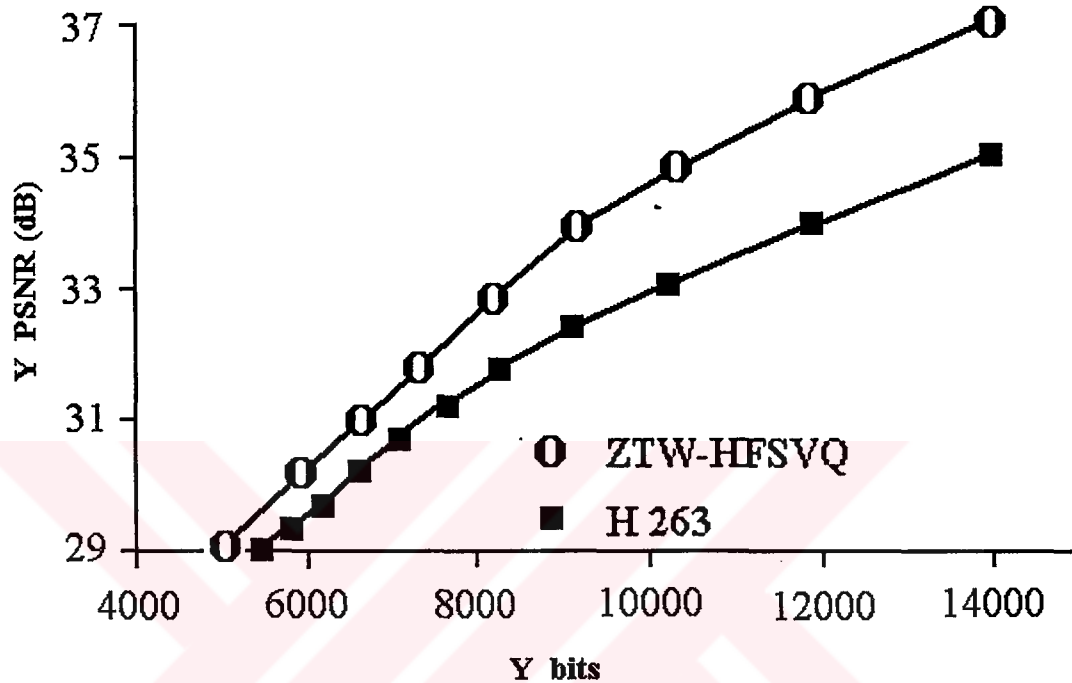


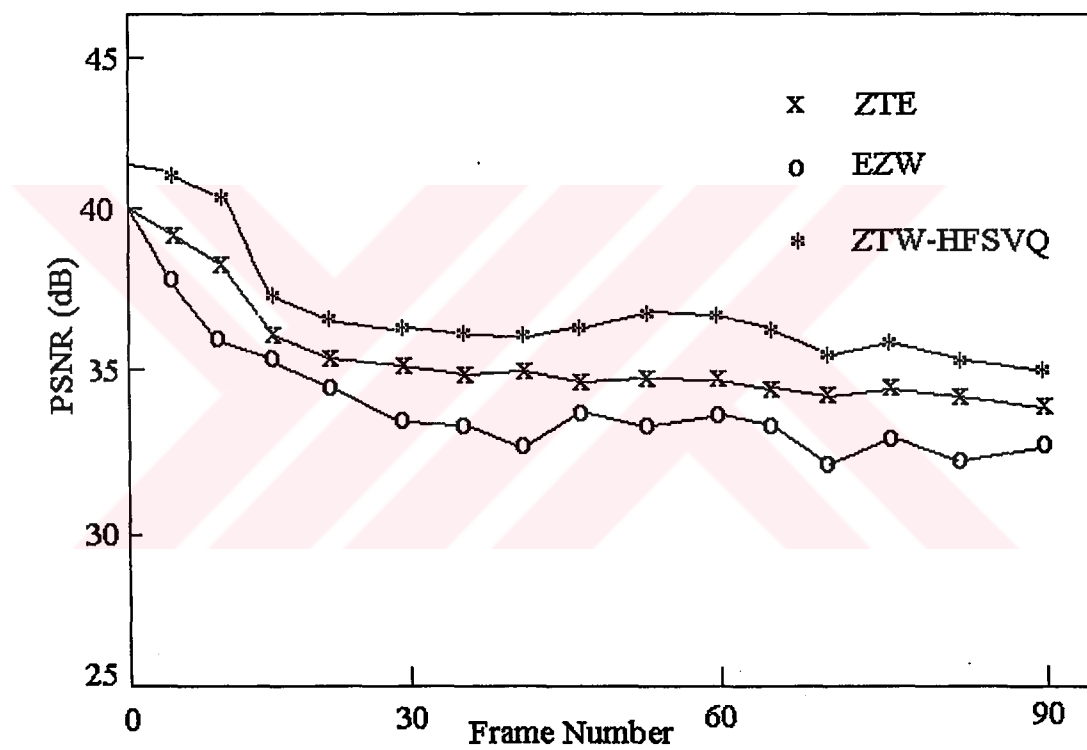
Figure 6.11 Intra coding comparison between H.263 and ZTW-HFSVQ

The second experimental result is a comparison of ZTE, EZW, MPEG-4/VM and ZTW-HFSVQ for P-frame coding.

The first comparison was made using Akiyo QCIF video sequence at the rates of 5 fr/s and 10 kb/s. This is shown in Figure 6.12(a), Figure 6.12(b) and Table 6.6 for luminance and chrominance frames (Martucci et al., 1997) (ITU, 1997). It is seen that, the ZTW-HFSVQ algorithm has 1.19 dB and 1.08 dB better performance than MPEG-4/VM and ZTE respectively for luminance frames and 2.34 dB and 1.96 dB for chrominance frames. It is also seen that ZTW-HFSVQ algorithm is 2.45 dB better than H.263 for luminance frames and 3.7 dB for chrominance frames.

Table 6.6 P Frame PSNR results for Akiyo at the rates of 5 fr/s and 10 kb/s

Sequence	Bit Rate	Y/C	MPEG4 VM	ZTE	H.263	ZTW-HFSVQ
Akiyo	10 kb/s	Y	35.64	35.75	34.38	36.83
		C	39.48	39.86	38.12	41.82

**Figure 6.12 a)** Luminance PSNR results for Akiyo at the rates of 5 fr/s and 10 kb/s

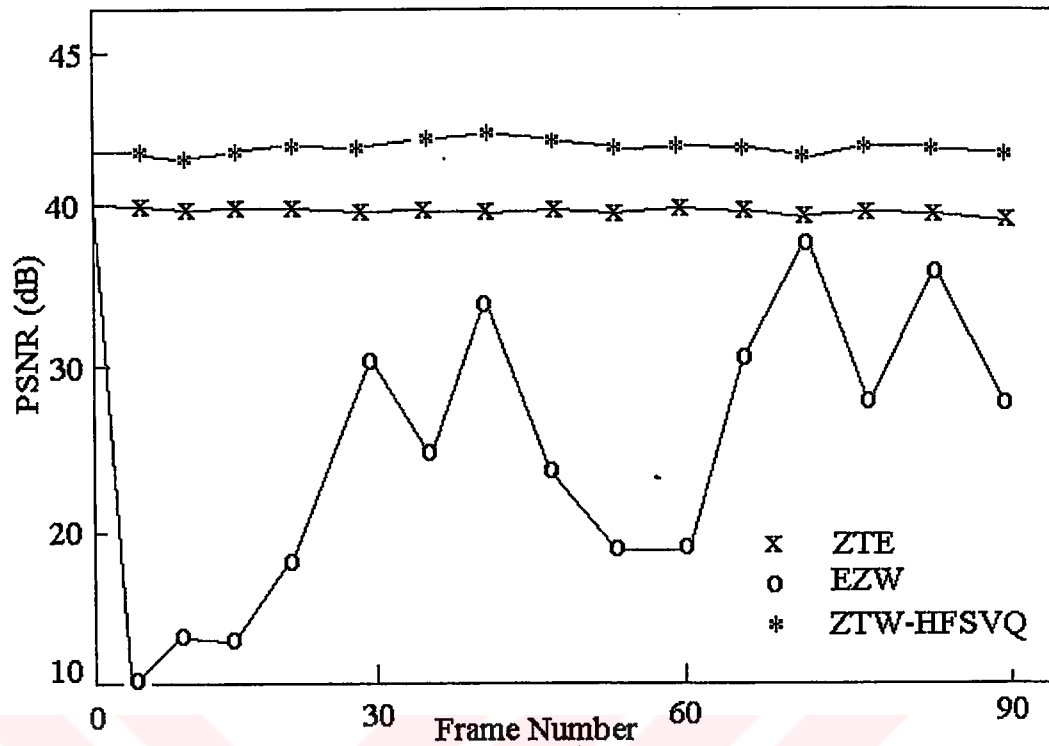


Figure 6.12 b) Chrominance PSNR results for Akiyo at the rates of 5 fr/s and 10 kb/s

The second comparison was made for Carphone QCIF video sequence at the rates of 10 f/s and 30 kb/s . The results are shown in Figure 6.13 and Table 6.7. The proposed coder achieves approximately 1.68 dB and 2.54 dB gain over H.263, 2.04 dB and 2.79 dB gain over MPEG-4 / VM for luminance and chrominance frames respectively.

Table 6.7 Carphone QCIF sequence at the rates of 10 fr/s and 30 kb/s

Algorithm	Frame Rate	Bit Rate	PSNR Y/C
H263	10 f/s	30 kb/s	31.42 dB / 37.98 dB
MPEG-4 / VM	10 f/s	30 kb/s	31.06 dB / 37.73 dB
ZTW-HFSVQ	10 f/s	30 kb/s	33.10 dB / 40.52 dB

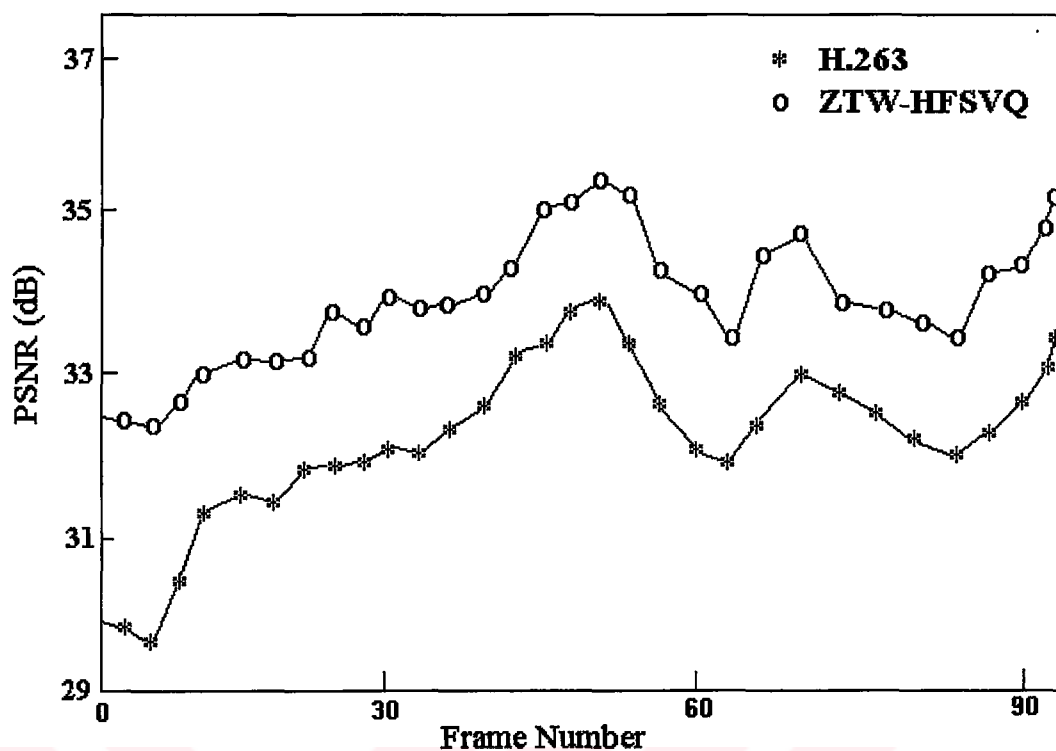


Figure 6.13 PSNR luminance results for Carphone at the rates of 10 f/s and 30 kb/s

The interframe visual comparison of 72nd Carphone luminance frame between ZTW-HFSVQ and H.263 is shown in Figure 6.14. When the PSNR values are compared, it is seen that the ZTW-HFSVQ algorithm has 1.2 dB improvement over H.263 algorithm.



(a)



(b)

Figure 6.14 Visual comparison of 72nd Carphone frame a) original frame b) H.263 result with PSNR=32.5 dB and c) ZTW-HFSVQ result with PSNR=33.7 dB.

2020-01-17 10:10:10



(c)

Figure 6.14 Visual comparison of 72nd Carphone frame a) original frame b) H.263 result with PSNR=32.5 dB and c) ZTW-HFSVQ result with PSNR=33.7 dB.

The third comparison was made for Coastguard QCIF video at the rates of 7.5 f/s and 48 kb/s . The results are shown in Figure 6.15 and Table 6.8 (Martucci et al., 1997) (Sun, 2002).

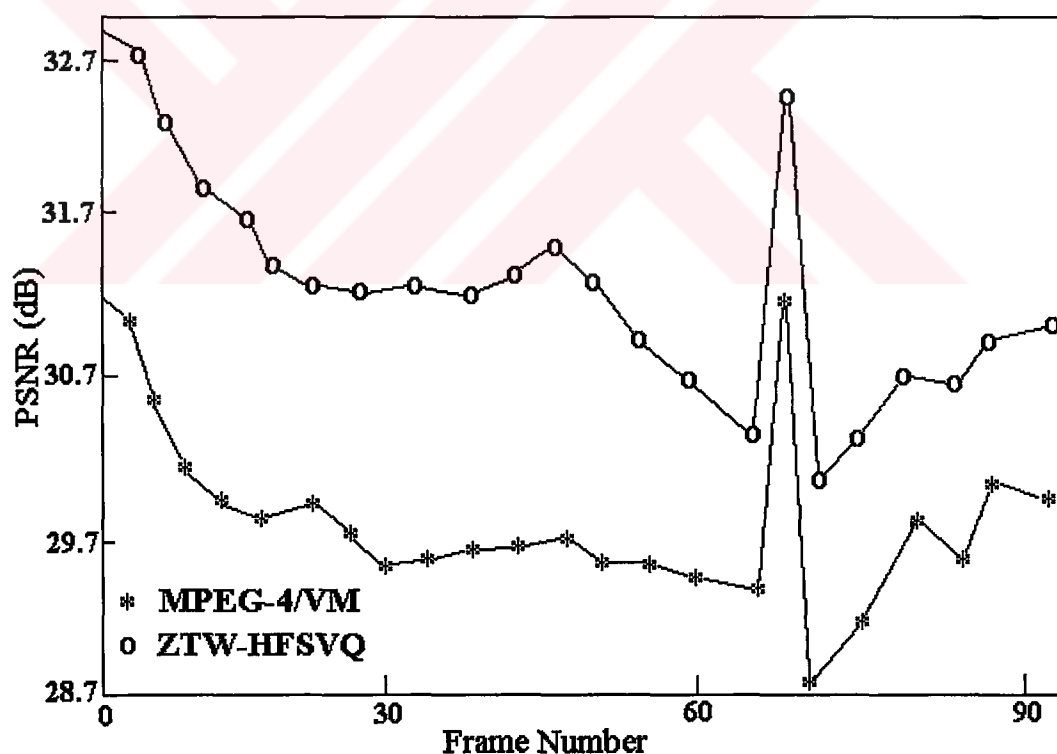


Figure 6.15 Luminance PSNR results for Coastguard at rates of 7.5 f/s and 48 kb/s

Table 6.8 Coastguard QCIF video sequence at the rates of 7.5 fr/s and 48 kb/s

Algorithm	Frame Rate	Bit Rate	PSNR Y / C
MPEG4-VM	7.5 f/s	48 kb/s	29.74 dB / 40.78 dB
H.263	7.5 f/s	48 kb/s	29.82 dB / 40.70 dB
ZTE	7.5 f/s	48 kb/s	29.20 dB / 40.88 dB
ZTW-HFSVQ	7.5 f/s	48 kb/s	30.92 dB / 42.82 dB

The average PSNR value of the ZTW-HFSVQ encoder over the entire coded sequence is 30.92 dB for luminance and 42.82 dB for chrominance frames. The average improvements in PSNR of the introduced encoder over MPEG-4/VM, ZTE and H.263 are 1.18 dB, 1.72 dB and 1.10 dB respectively for luminance frames and 2.04 dB, 1.94 dB and 2.12 dB for chrominance frames. The interframe visual comparison of 40th luminance Coastguard frame between ZTW-HFSVQ and MPEG-4/VM is shown in Figure 6.16. When the PSNR values are compared, it is seen that the ZTW-HFSVQ algorithm has 1.6 dB improvement over MPEG-4/VM algorithm.

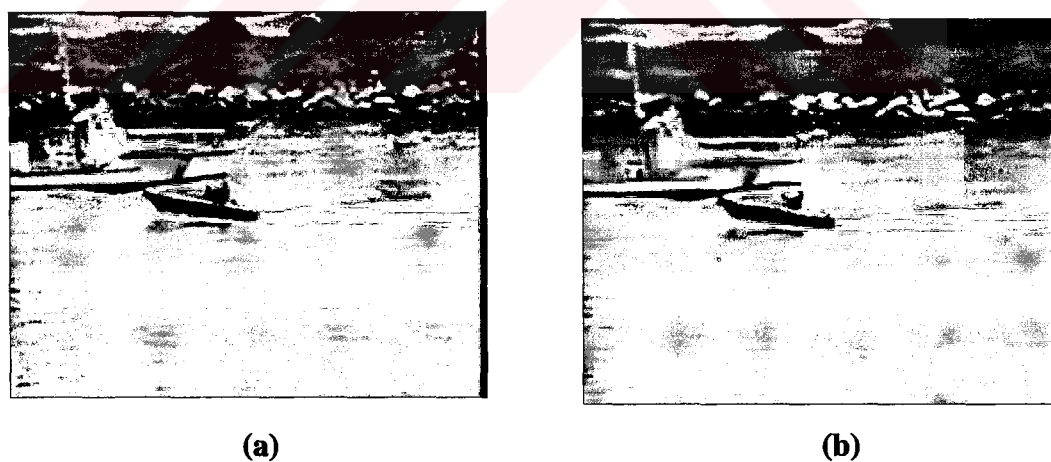


Figure 6.16 Visual comparison of 40th luminance Coastguard frame a) original frame b) Reconstructed by MPEG-4/VM, PSNR=29.7 dB and c) Reconstructed by ZTW-HFSVQ, PSNR=31.3 dB.



(c)

Figure 6.16 Visual comparison of 40th luminance Coastguard frame a) original frame b) Reconstructed by MPEG-4/VM, PSNR=29.7 dB and c) Reconstructed by ZTW-HFSVQ, PSNR=31.3 dB.

The performance results of H.263, MPEG-4/VM and ZTW-HFSVQ video compression techniques using Akiyo, Carphone and Coastguard QCIF video sequences are shown in Table 6.9 (Martucci et al., 1997) (ITU, 1997)(Sun, 2002). It is seen that for the QCIF videos that contains low movement such as Akiyo the performance of MPEG-4 / VM is better than H.263. For the QCIF videos that contains medium or much movement such as Carphone and Coastguard respectively, the performance of H.263 is better than MPEG-4 / VM. On the other hand for the three type of QCIF videos which contains low, medium and much movements such as Akiyo, Carphone and Coastguard respectively the performance of the proposed video coding technique is better than the H.263 and MPEG-4 / VM techniques.

Table 6.9 The average results of H.263, MPEG-4/VM and ZTW-HFSVQ in dB.

Sequence	Fr. R.	B.R	H 263		MPEG 4 / VM		ZTW HFSVQ	
	(fr/s)	(kb/s)	Y	C	Y	C	Y	C
Akiyo	5	10	34.4	38.1	35.6	39.5	36.8	41.8
Carphone	10	30	31.4	38.0	31.1	37.7	33.1	40.5
Coastguard	7.5	48	29.8	40.7	29.7	40.8	30.9	42.8

CHAPTER 7

CONCLUSION

In this thesis, the search for improving the video compression techniques is addressed. The main concern is to design a video compression technique with comparable low bit rates. The proposed encoder is similar to other motion compensated, block based discrete cosine transform (DCT) video coding standards such as MPEG-1/2, H.261/3 and MPEG-4/Verification Model (VM). In this introduced encoder, the discrete wavelet transform (DWT) is used instead of DCT, overlapping block motion compensation (OBMC) is added to the block motion estimation algorithm to match better DWT and for coding the wavelet coefficients, the zerotree wavelet structure is combined with the HFSVQ.

In this study, performances of the popular block motion estimation algorithms are compared using standard 100 frames of Football and Miss America video sequences. These block motion estimation algorithms are Two Dimensional Logarithmic Search (TDL), One at a Time Search (OTS), Three Step Search(3SS), New Three Step Search (N3SS), Improved Three Step Search (I3SS), Novel Four Step Search (N4SS), Simple and Efficient Search (SES), New Diamond Search (NDS). The Mean Square Error (MSE) and the Search Point Number (SPN) criterion are used for the performance comparison.

The block motion estimation algorithm is performed by using large macro block size (16x16). That's why the average MSE value for large block sizes is considered. The results show that the NDS has the best performance. Unfortunately the SES, TDL and I3SS, which have very good performances as far as SPN is concerned, have very poor MSE values. When both average SPN and MSE are concerned, it can be

seen easily that NDS has the best performance. Therefore as a block motion estimation algorithm the NDS is selected to track the local motion.

The overlapping block motion compensation (OBMC) (Auyeung et al., 1992) technique which is an advance scheme in the H.263 standard is described and its performance results are also presented. The standard OBMC filters are 8x8 pixel size for the 16x16 macroblocks. Since the performance comparison of the block motion estimation algorithms mentioned above are done using different size of macroblocks then additional new OBMC filters are designed for 8x8, 16x8, 8x16, 4x4 macroblock sizes. The OBMC technique is used after the block motion algorithm to reduce the block artifacts.

In order to remove the spatial correlation a three level of discrete wavelet transform (DWT) is used instead of DCT. DWT seems to have better performance than DCT for image and video coding (Xiong, Z. et al., 1999). DWT also provides us the scalability functionalities of image and video. The encoder uses different filter bank at each level of the decomposition. Since the longer filters provide good frequency localization and shorter filters cause less ringing artifacts, the Biorthogonal 9.3 filter bank is used at the first level of the DWT decomposition followed by the Haar filter for the remaining two levels.

It is seen that the DWT coefficients of a QCIF video frame are Gaussian shape distributed in each level except the lowest frequency subband. The Lloyd-Max is a non-uniform quantizer (Yin, C.Y., 2000) that follows the Gaussian curve and finds an optimum number of nonuniform quantization intervals depending on the standard deviation criteria. In the proposed encoder only the lowest frequency subband is uniformly quantized and the rest of the subbands are nonuniformly quantized by using the Lloyd-Max quantizer. At the end of nonuniform quantization process each standard deviation and the number of quantization intervals for each subband are sent as side information.

In a multiresolution DWT system, except the highest frequency subbands, every coefficient at a given scale can be related to a set of coefficient of similar orientation at the next finer scale. The coefficient at the coarse scale is called the parent, and all coefficients at the same spatial location and of similar orientation at the next finer scale are called children (Martucci, S.A., 1997). The zerotree wavelet structure(ZTW) collects all parent and children wavelet blocks together. The proposed algorithm is similar to the ZTE scheme introduced in (Martucci, S.A., 1997). The main difference is combining the zerotree wavelet structure with HFSVQ instead of arithmetic coding. It is important to note that, the zerotree structure consists of different size of wavelet blocks that contain similar informations. Because of that it is very suitable for these blocks to be coded by HFSVQ algorithm.

The rate control was performed by the encoder in three possible mechanism; 1) Increasing the number of the Lloyd-Max quantization intervals of any subband also increases the total bitrate and quality of the video frame. On the contrary decreasing it causes low quality and low bit rate, 2) Increasing the bandwidth of the deadzone area of any subband causes a decrease both in the quality of video frame and total bit rate. On the other hand smaller deadzone areas of any subband increases the bitrate and the quality of the frame.3) When the HFSVQ is applied to any ZTW structure, then each type of wavelet block will be represented by different size of codeword. That will give us the third opportunity of controlling the bit rate. Note that increasing or decreasing the parameters mentioned above are performed in all subbands.

The encoder is computer simulated by using the Object Oriented Borland C++ Builder software. The simulation results of the proposed encoder was taken using three types of standard video sequences; QCIF Akiyo which contains a few movements at the rates of 10 kb/s and 5 fr/s, QCIF Carphone which contains relatively more movements than Akiyo at the rates of 30 kb/s and 10 fr/s and QCIF Coastguard which contains more movements than the others at the rates of 48 kb/s and 7.5 fr/s.

In the first stage of simulation, the intra frame coding results of the ZTE, EZW, MPEG-4/VM and ZTW-HFSVQ techniques are compared at QCIF resolution. The comparison was made at 14 kb, 27 kb and at 28 kb using the first frames of Akiyo, News, Foreman. It is seen that ZTE gives better performance than MPEG-4/VM and EZW. On the other hand the proposed encoder seems to be better than the other three algorithms mentioned above both in luminance and chrominance for all intra test frames. In this stage of simulation, the Intra coding results of H.263 and ZTW-HFSVQ are also compared at different compression ratios using the first frame of QCIF Akiyo video. At the high compression ratios the proposed algorithm is seems to have approximately 1 dB gain and for low compression ratio the gain is approximately 2 dB over the H.263 standard.

In the second stage, MPEG-4/VM, EZW, ZTE and H263 are compared with the introduced ZTW-HFSVQ scheme for P-frames. The first comparison was made using Akiyo QCIF video sequence at the rate of 5 fr/s and 10 kb/s. It is seen that the ZTW-HFSVQ algorithm has 1.19 dB and 1.08 dB better performance than MPEG-4/VM and ZTE respectively for luminance frames and 2.34 dB and 1.96 dB for chrominance frames. It is also seen that ZTW-HFSVQ algorithm is 2.45 dB and 3.7 dB better than H.263 for luminance and chrominance frames respectively. The second comparison was made for Carphone QCIF video sequence at the rates of 10 f/s and 30 kb/s. The proposed coder achieves approximately 1.68 dB and 2.54 dB gain over H.263, 2.04 dB and 2.79 dB gain over MPEG-4 / VM for luminance and chrominance frames respectively. The third comparison was made for Coastguard QCIF video sequence at the rates of 7.5 f/s and 48 kb/s. The average improvements in PSNR of the introduced encoder over MPEG-4/VM, ZTE and H.263 are 1.18 dB, 1.72 dB and 1.10 dB respectively for luminance frames and 2.04 dB, 1.94 dB and 2.12 dB for chrominance frames.

Average PSNR over the entire coded sequences show that the proposed ZTW-HFSVQ video compression technique achieves comparable performance over MPEG-4/VM, H.263, ZTE and EZW for both I and P frames.

REFERENCES

- Al-Shaykh, O.K., Miloslavsky, E., Nomura, T., Neff, R. & Zakhor, A. (1999). Video Compression Using Matching Pursuits. IEEE Transactions on Circuits and Systems for Video Technology, 9, 123-143.
- Auyeung, C, Kosmach, J., Orchard, M. & Kalafatis, T. (1992). Overlapped Block Motion Compansation. SPIE'92, pp. 561-572.
- Averbuch, A., Lazar, D. & Israeli, M. (1996). Image Compression Using Wavelet Transform and Multiresolution Decomposition. IEEE Transactions on Image Processing, 5, 4-15.
- Burrus, C.S., Gopinath, R.A. & Guo, H. (1998). Introduction to Wavelets and Wavelet Transforms, New Jersey: Prentice Hall.
- Chang, P.C. & Lu, T.T. (1999). A Scalable Video Compression Technique Based on Wavelet Transform and MPEG Coding. IEEE Transactions on Consumer Electronics, 45, 788 – 793.
- Chiariglione, L. (1995). The development of an integrated audiovisual coding standard: MPEG, Proceeding IEEE, Vol. 83, pp. 151-157.
- Cho, S. & Pearlman, W.A. (2002). A Full-Featured, Error-Resilient, Scalable Wavelet Video Codec Based on the Set Partitioning in Hierarchical Trees (SPIHT) Algorithm. IEEE Transactions on Circuits and Systems for Video Technology, 12, 157 - 171.
- Computer Architecture Research Unit, School of Science and Technology University of Teesside. (1998). An Improved Three-Step Search Block-Matching Algorithm for Low Bit-Rate Video Coding Applications, United Kingdom: Xu, D., Bailey, C. & Sotudeh, R.
- Daubechies, I. (1988). Orthonormal bases of compactly supported wavelets. Communications on Pure and Applied Mathematics, 41, 909-996.

- Gall, D. J. (1992). The MPEG video compression algorithm, Signal Processing: Image Communication, Vol. 4, No. 4, pp. 129-140.
- Gray, R.M. (1984). Vector Quantization. IEEE ASSP Magazine, pp. 4-23.
- Idris, F.M. & Panchanathan, S. (1997). Spatio-Temporal Indexing of Vector Quantized Video Sequences. IEEE Transactions on Circuits and Systems for Video Technology, 7, 728 - 740.
- ITU – Telecommunications Standardization Sector. (1997). Using improved coding efficiency to increase frame rate . Sunriver, Oregon : Study Group 16.
- Khalil, H., Atiya, A.F. & Shaheen, S. (1999). Three Dimensional Video Compression Using Subband / Wavelet Transform with Lower Buffering Requirements. IEEE Transactions on Image Processing, 8, 762-773.
- Kılıç, İ. & Yılmaz, R. (2001). Güncel Hareket Kestirim Algoritmalarının Başarımlarının Karşılaştırılması. SIU'2001, pp.41-46.
- Kim, B.J., Xiong, Z. & Pearlman, W.A. (2000). Low Bit-Rate Scalable Video Coding with 3-D Set Partitioning in Hierarchical Trees (3-D SPIHT). IEEE Transactions on Circuits and Systems for Video Technology, 10, 1374-1387.
- Kim, H.J. & Li, C.C. (1998). Lossless and Lossy Image Compression Using Biorthogonal Wavelet Transforms with Multiplierless Operations. IEEE Transactions on Circuits and Systems for Video Technology, 45, 1113-1118.
- Lin, Y.C & Tai, S.C. (1998). A Fast Linde-Buzo-Gray Algorithm in Image Vector Quantization. IEEE Transactions on Circuits and Systems-II : Analog and Digital Signal Processing, 45, 432-435.
- Li, R., Zeng, B. & Liou, M.L. (1994). A New Three-Step Search Algorithm for Block Motion Estimation. IEEE Transactions on Circuits and Systems for Video Technology, 4, 438-442.
- Lloyd, S.P. (1982). Least Squares Quantization in PCM. IEEE Transactions on Information Theory, 28, 129-137.
- Lo, K.T., Zhang, X.D., Feng, J. & Wang, D.S. (2000). Universal Perceptual Weighted Zerotree Coding for Image and Video Compression. IEE Proceeding, Vision, Image, Signal Processing, 147, 261-265.

- Lu, J. & Liou, M.L. (1997) A Simple and Efficient Search Algorithm for Block-Matching Motion Estimation. IEEE Transactions on Circuits and Systems for Video Technology, 7, 429-433.
- Marpe, D. & Cycon, H.L. (1999). Very Low Bit-Rate Video Coding Using Wavelet-Based Techniques. IEEE Transactions on Circuits and Systems for Video Technology, 9, 85 - 94.
- Martucci, S.A., Sodagar, I., Chiang, T., & Zhang, Y.Q. (1997). A Zerotree Wavelet Video Coder. IEEE Transactions on Circuits and Systems for Video Technology, 7, 109-118.
- Meier, T., Ngan, K.N. & Crebbin, G. (1999). Reduction of Blocking Artifacts in Image and Video Coding. IEEE Transactions on Circuits and Systems for Video Technology, 9, 490 – 500.
- Munteanu, A., Cornelis, J., Auwera, G.V.D. & Cristea, P. (1999). Wavelet Image Compression – The Quadtree Coding Approach. IEEE Transactions on Information Technology in Biomedicine, 3, 176-185.
- Po, L.M. & Ma, W.C. (1996). A Novel Four-Step Search Algorithm for Fast Block Motion Estimation. IEEE Transactions on Circuits and Systems for Video Technology, 6, 313-317.
- Rao, K.R. & Hwang, J.J. (1996). Techniques and Standards for Image, Video, and Audio Coding, New Jersey: Prentice Hall.
- Rao, R.M. & Bopardikar, A.S. (1998). Wavelet Transforms Introduction to Theory and Applications, Massachusetts: Addison Wesley Longman.
- Rogers, J.K. & Cosman, P.C. (1998). Wavelet Zerotree Image Compression with Packetization. IEEE Signal Processing Letters, 5, 105-107.
- Shen K. & Delp, E.J. (1999). Wavelet Based Rate Scalable Video Compression. IEEE Transactions on Circuits and Systems for Video Technology, 9, 109 - 122.
- Sikora, T. (1997). The MPEG-4 Video Standard Verification Model. IEEE Transactions on Circuits and Systems for Video Technology, 7, 19-31.
- Sodagar, I., Lee, H.J., Hatrack, P. & Zhang, Y.Q. (1999). Scalable Wavelet Coding for Synthetic/Natural Hybrid Images. IEEE Transactions on Circuits and Systems for Video Technology, 9, 244-254.

- Stang, G. & Nguyen, T. (1996). Wavelets and Filter Banks, United States of America: Wellesley-Cambridge Press.
- Sun, X., Wu, F., Li, S. & Gao, W. (2002). In-Loop Deblocking Filter for Block Based Video Coding. ICSP 2002.
- Wang, Q. & Ghanbari, M. (1997). Scalable Coding of Very High Resolution Video Using the Virtual Zerotree. IEEE Transactions on Circuits and Systems for Video Technology, 7, 719-727.
- Xiong, Z., Ramchandran, K., Orchard, M.T. & Zhang, Y.Q. (1999). A Comparative Study of DCT and Wavelet Based Image Coding. IEEE Transactions on Circuits and Systems for Video Technology, 9, 692-695.
- Xu, D. & Bailey C., (1998). An Improved Three Step Search Block-Matching Algorithm for Low Bit-Rate Video Coding Applications. Computer Architecture Research Unit, School of Science and Technology University of Teesside, United Kingdom.
- Yeung, E. (1997). Image Compression Using Wavelets. IEEE International Conferences CCECE'97, pp. 241-244.
- Yin, C.Y. (2000). Image and Video Compression Using Embedded Zerotree Coding, Massachusetts: Ph.D. Thesis.
- Yu, P. & Venetsanopoulos, A. N. (1994). Hierarchical Finite-State Vector Quantization for Image Coding. IEEE Transactions on Communications, 42, 3020-3026.
- Zheng, W.X & Quan, Z.Y. (1996). Image Sequence Coding Using Wavelet Transform. IEEE Digital Signal Processing Workshop, pp.97-100.
- Zhou, Z. & Hung, D. (1998). Digital Design of Discrete Wavelet Transform for MPEG-4. IEEE International ASIC Conference (11th), pp. 333-336.
- Zhu, S. & Ma, K.K. (2000). A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation. IEEE Transactions on Image Processing, 9, 287-290.