PM-1 3 1/2" x 4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file / Votre référence

Our file / Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Image Compression
Using Multilevel Block Truncation Coding and
Pattern Matching
in the Wavelet Domain

by

Captain Luc Boileau, B.Sc. CD

A thesis submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements
for the degree of
Master of Engineering

Ottawa-Carleton Institute for Electrical Engineering
Faculty of Engineering
Department of Systems and Computer Engineering
Carleton University,
Ottawa, Ontario, Canada

27 September, 1995

© copyright

1995, Luc Boileau

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Sur titre - Votre référence

Sur titre - Notre référence

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-612-08973-8

Name MR. Luc Boileau

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Digital Image Compression in the Wavelet Domain

SUBJECT TERM

0 5 4 4

SUBJECT CODE

U·M·I

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION

General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0273
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523

Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language
General 0679
Ancient 0289
Linguistics 0290
Modern 0291
Literature
General 0401
Classical 0294
Comparative 0295
Medieval 0297
Modern 0298
African 0316
American 0591
Asian 0305
Canadian (English) 0352
Canadian (French) 0355
English 0593
Germanic 0311
Latin American 0312
Middle Eastern 0315
Romance 0313
Slavic and East European 0314

PHILOSOPHY, RELIGION AND THEOLOGY

Philosophy 0422
Religion
General 0318
Biblical Studies 0321
Clergy 0319
History of 0320
Philosophy of 0322
Theology 0469

SOCIAL SCIENCES

American Studies 0323
Anthropology
Archaeology 0324
Cultural 0326
Physical 0327
Business Administration
General 0310
Accounting 0272
Banking 0770
Management 0454
Marketing 0338
Canadian Studies 0385
Economics
General 0501
Agricultural 0503
Commerce-Business 0505
Finance 0508
History 0509
Labor 0510
Theory 0511
Folklore 0358
Geography 0366
Gerontology 0351
History
General 0578

Ancient 0579
Medieval 0581
Modern 0582
Black 0328
African 0331
Asia, Australia and Oceania 0332
Canadian 0334
European 0335
Latin American 0336
Middle Eastern 0333
United States 0337
History of Science 0585
Law 0398
Political Science
General 0615
International Law and Relations 0616
Public Administration 0617
Recreation 0814
Social Work 0452
Sociology
General 0626
Criminology and Penology 0627
Demography 0938
Ethnic and Racial Studies 0631
Individual and Family Studies 0628
Industrial and Labor Relations 0629
Public and Social Welfare 0630
Social Structure and Development 0700
Theory and Methods 0344
Transportation 0709
Urban and Regional Planning 0999
Women's Studies 0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture
General 0473
Agronomy 0285
Animal Culture and Nutrition 0475
Animal Pathology 0476
Food Science and Technology 0359
Forestry and Wildlife 0478
Plant Culture 0479
Plant Pathology 0480
Plant Physiology 0817
Range Management 0777
Wood Technology 0746
Biology
General 0306
Anatomy 0287
Biostatistics 0308
Botany 0309
Cell 0379
Ecology 0329
Entomology 0353
Genetics 0369
Limnology 0793
Microbiology 0410
Molecular 0307
Neuroscience 0317
Oceanography 0416
Physiology 0433
Radiation 0921
Veterinary Science 0778
Zoology 0472
Biophysics
General 0786
Medical 0760

EARTH SCIENCES

Biogeochemistry 0425
Geochemistry 0996

Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences 0768
Health Sciences
General 0566
Audiology 0300
Chemotherapy 0992
Dentistry 0567
Education 0350
Hospital Management 0769
Human Development 0758
Immunology 0982
Medicine and Surgery 0564
Mental Health 0347
Nursing 0569
Nutrition 0570
Obstetrics and Gynecology 0380
Occupational Health and Therapy 0354
Ophthalmology 0381
Pathology 0571
Pharmacology 0419
Pharmacy 0572
Physical Therapy 0382
Public Health 0573
Radiology 0574
Recreation 0575

Speech Pathology 0460
Toxicology 0383
Home Economics 0386

PHYSICAL SCIENCES

Pure Sciences
Chemistry
General 0485
Agricultural 0749
Analytical 0486
Biochemistry 0487
Inorganic 0488
Nuclear 0738
Organic 0490
Pharmaceutical 0491
Physical 0494
Polymer 0495
Radiation 0754
Mathematics 0405
Physics
General 0605
Acoustics 0986
Astronomy and Astrophysics 0606
Atmospheric Science 0608
Atomic 0748
Electronics and Electricity 0607
Elementary Particles and High Energy 0798
Fluid and Plasma 0759
Molecular 0609
Nuclear 0610
Optics 0752
Radiation 0756
Solid State 0611
Statistics 0463
Applied Sciences
Applied Mechanics 0346
Computer Science 0954

Engineering
General 0537
Aerospace 0538
Agricultural 0539
Automotive 0540
Biomedical 0541
Chemical 0542
Civil 0543
Electronics and Electrical 0544
Heat and Thermodynamics 0348
Hydraulic 0545
Industrial 0546
Marine 0547
Materials Science 0794
Mechanical 0548
Metallurgy 0743
Mining 0551
Nuclear 0552
Packaging 0549
Petroleum 0765
Sanitary and Municipal System Science 0790
Geotechnology 0428
Operations Research 0796
Plastics Technology 0795
Textile Technology 0994

PSYCHOLOGY

General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

The undersigned recommends to the Faculty of Graduate Studies and Research
acceptance of the thesis

Image Compression
Using Multilevel Block Truncation Coding and
Pattern Matching
in the Wavelet Domain

submitted by

Captain Luc Boileau, B.Sc. CD

in partial fulfillment of the requirements for
the degree of Master of Engineering

Dr. D.C. Coll, Thesis Supervisor

Dr. S.A. Mahmoud, Chair, Department of Systems
and Computer Engineering

Carleton University
14 November, 1995

Abstract

In many different fields, digitized images are replacing conventional analog images such as photograph and x-rays. The volume of data required to represent such images greatly slows transmission and makes storage prohibitively costly. One way to overcome these problems is by compressing the image data, that is, extracting the *visible elements* from the information contained in the images and encoding them.

This thesis documents the implementation of new coding schemes which take into account psychovisual features both in the spatial and wavelet domains. It demonstrates that the coding of images in the wavelet domain can be done more efficiently.

The image coding schemes introduced in this thesis are based on the concept of pattern matching. The binary coding scheme used is a pattern matching-based coding technique called *Visual Pattern Image Coding* (VPIC). Visual patterns, which are pre-defined subimages of limited spatial support derived from relevant psychophysical and physiological data, are matched against wavelet subimages and adequately encoded so as to maintain a high degree of visual fidelity and compression. Results reveal that it is slightly more efficient to code images in the wavelet domain than in the spatial domain. VPIC, however, turned out to be not good as reported in the literature.

Additionally, ternary and quaternary block truncation coding algorithms are developed and tested to overcome the low image fidelity characteristic of the VPIC technique. Test results show that multi-level, moment-preserving, quantization schemes have a higher level of performance than binary moment-preserving quantization schemes.

Acknowledgments

First, I would like to express my deep gratitude to my thesis supervisor Dr. David C. Coll for his guidance and intuition, both of which contributed significantly to this thesis. I would like to thank the Department of National Defence (DND) for its funding support, and a special thank to Mr. André Mainguy from the Communication Research Centre (CRC) for providing support in the theory of wavelet transforms.

I wish to give my special appreciation and gratitude to Nicole, the most important person in my life, for her consistent support and encouragement provided during the preparation of this thesis. I shall always be grateful to her for her countless sacrifices and patience. Without her support, this thesis would never have been. And finally, I would like to thank our children Mélanie, Katrine, Cindy and Junior for their understanding for the time spent away from them to do this work.

Table of Contents

Abstract.....	iii
Acknowledgements.....	iv
List of Tables	x
List of Figures.....	xiii
List of Appendices.....	xviii
CHAPTER 1 - Introduction	1
1.1 Thesis Topic.....	3
1.2 Achievements.....	5
1.3 Results Overview	7
1.4 Thesis Organization	8
CHAPTER 2 - Image Data Compression.....	10
2.1 General	10
2.2 Spatial Domain	12
2.2.1 Thresholding.....	12
2.2.2 Edge Detection	13
2.2.3 Gradient.....	13
2.2.4 Pattern Matching	15
2.2.5 Vector Quantization (VQ).....	18
2.2.6 Block Truncation Coding (BTC).....	20

2.2.7	Visual Pattern Image Coding (VPIC).....	21
2.2.8	Multi-Level Image Coding (MLIC).....	23
2.3	Wavelet Domain	23
2.3.1	General	23
2.3.2	Introduction to Wavelet Transform.....	27
2.3.3	Time-Frequency Analysis.....	28
2.3.4	The Continuous Wavelet Transform.....	29
2.3.5	The 1-D Discrete Wavelet Transform.....	31
2.3.5.1	The Analysis Stage.....	33
2.3.5.2	The Synthesis Stage.....	37
2.3.6	The 2-D Discrete Wavelet Transform	39
CHAPTER 3 - VPIC		42
3.1	General.....	42
3.2	Properties of Biological Vision.....	42
3.3	VPIC Pattern Design.....	44
3.3.1	Edge Pattern Design	45
3.3.2	VPIC Codebook Generation.....	48
3.3.3	Gradient Based Thresholding	53
3.4	VPIC Architecture	54
3.4.1	VPIC Coder	55
3.4.1.1	Uniform Block Coding.....	57
3.4.1.2	Edge Block Coding	57

3.4.1.3	VPIC Pattern Matching	58
3.4.2	VPIC Decoder	59
3.4.2.1	Postfiltering	60
3.4.3	Computing Compression Ratios in VPIC	62
3.4.4	Simulations and Results	63
CHAPTER 4 - Image Data Compression in the Wavelet Domain		76
4.1	Compression Advantages	76
4.1.1	High Resolution Subbands not Transmitted.....	76
4.1.2	Local Means not Transmitted.....	82
4.1.3	Residual not Used.....	88
4.2	Binary Coding.....	89
4.2.1	A 64K-Pattern WVPIC Coding Scheme With 4 x 4 Block Segmentation.....	91
4.2.1.1	Zero-Gradient Patterns	91
4.2.1.2	A 64K-Pattern WVPIC/MAE BTC Achitecture With 4 x 4 Block Segmentation	93
4.2.1.3	The BTC Oriented Gradient Coefficients in a 64K-Pattern Codebook.....	99
4.2.1.4	Result of the 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation	102
4.2.2	A Modified 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block segmentation.....	110

4.2.3	A 256K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block segmentation.....	117
4.2.4	A 28-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block segmentation.....	122
4.2.5	Analysis of the WVPIC/MAE BTC Coding Scheme With 4 x 4 Block segmentation.....	124
4.2.6	A 16-Pattern WVPIC/MAE BTC Coding Scheme With 2 x 2 Block segmentation.....	126
4.2.7	Analysis and Limitations of VPIC and WVPIC/MAE BTC Coding Schemes	131
4.3	Ternary Coding	133
4.3.1	The Three-Level WMME BTC Coding Scheme.....	135
4.3.1.1	The Three-Level WMME BTC Coder	138
4.3.1.2	The Three-Level WMME BTC Decoder	143
4.3.2	Results of the Three-Level WMME BTC Coding Scheme With 4 x 4 Block segmentation	145
4.3.3	Results of the Three-Level WMME BTC Coding Scheme With 2 x 2 Block segmentation	148
4.4	Quaternary Coding.....	152
4.4.1	The Four-Level WMME BTC Coding Scheme	152
4.4.1.1	The Four -Level WMME BTC Coder.....	154

4.4.1.2	The Four -Level WMME BTC Decoder	158
4.4.2	Results of the Four -Level WMME BTC Coding	
	Scheme With 4 x 4 Block segmentation	159
4.4.3	Results of the Four -Level WMME BTC Coding	
	Scheme With 2 x 2 Block segmentation	164
4.4.4	Results of the Four -Level WMME BTC Coding	
	Scheme With Variable Block segmentation.....	168
CHAPTER 5 - Conclusion.....		172
5.1	Summary of Contributions.....	172
5.2	Conclusion	172
5.3	Future Research	176
References		180

List of Tables

Table 2-1: Major First Generation Image Coding Techniques	10
Table 2-2: 6-Tap Daubechies Wavelet Filter Coefficients.....	36
Table 3-1: Input Parameters Used in VPIC Simulations.....	63
Table 3-2: Simulations Results for VPIC Coding Scheme Applied in the Spatial Domain.....	64
Table 4-1: Results of Discarding of Subbands.....	77
Table 4-2: Energy Distribution for a Level $M = 3$ Wavelet Decomposition of <i>Lena</i>	79
Table 4-3: Means and Variances for a Level $M = 3$ Wavelet Decomposition of <i>Lena</i>	82
Table 4-4: In-Depth Analysis of Segmented Subbands for a Level $M = 3$ Wavelet Decomposition of <i>Lena</i>	83
Table 4-5: Subband Variances	86
Table 4-6: Experimental Results of the 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image <i>Lena</i>	102
Table 4-7: Experimental Results of the Modified 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image <i>Lena</i>	114
Table 4-8: Experimental Results of the 256-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image <i>Lena</i>	120

Table 4-9:	Experimental Results of the 28-Pattern WVPIC/MAE BTC Coding	
	Scheme With 4 x 4 Block Segmentation Using Test Image <i>Lena</i>	122
Table 4-10:	In-Depth Analysis of 2 x 2 Segmented Subbands for a Level $M = 3$	
	Wavelet Decomposition of Test Image <i>Lena</i>	126
Table 4-11:	Experimental Results of the 16-Pattern WVPIC/MAE BTC Coding	
	Scheme With 2 x 2 Block Segmentation Using Test Image <i>Lena</i>	128
Table 4-12:	Three-Level WMME BTC Coding/Decoding Rules for Each	
	Transform Coefficient	137
Table 4-13:	Experimental Results of the Three-Level WMME BTC Coding	
	Scheme With 4 x 4 Block Segmentation Using Test Image <i>Lena</i>	145
Table 4-14:	Experimental Results of the Three-Level WMME BTC Coding	
	Scheme With 2 x 2 Block Segmentation Using Test Image <i>Lena</i>	148
Table 4-15:	Three-Level WMME BTC Coding/Decoding Format	152
Table 4-16:	Four-Level WMME BTC Coding/Decoding Format	153
Table 4-17:	Experimental Results of the Four-Level WMME BTC Coding	
	Scheme With 4 x 4 Block Segmentation Using Test Image <i>Lena</i>	159
Table 4-18:	Experimental Results of the Four-Level WMME BTC Coding	
	Scheme With 2 x 2 Block Segmentation Using Test Image <i>Lena</i>	165
Table 4-19:	Experimental Results Using Variable Block Segmentation and	
	Coding Techniques Using Test Image <i>Lena</i>	169
Table A-1:	Thesis Sources Codes Description for Unix-Based Programs	184
Table A-2:	Digital Image Processing Utilities for DOS-Based Programs	186

Table A-3:	Digital Image Statistical Algorithms for DOS Environment	189
Table A-4:	Digital Image Codebook/Pattern Generation Algorithms for DOS Environment	192
Table A-5:	Digital Image Coding Algorithms for DOS Environment	193

List of Figures

Figure 1-1: A 3-Level Wavelet Decomposition of Test Image <i>Lena</i>	3
Figure 2-1: Pattern Relationship Between Spatial and Wavelet Domains	18
Figure 2-2: VQ Block Diagram	20
Figure 2-3: BTC Block Diagram	21
Figure 2-4: A Simplified Block Diagram of VPIC	22
Figure 2-5: Decimation and Interpolation	31
Figure 2-6: Block Diagram of the Analysis Stage ($M = 3$) of a DWT	32
Figure 2-7: Octave Band Tree Structure and Its Frequency Band Splits	33
Figure 2-8: Block Diagram of the Analysis Stage ($M = 3$) of a DWT	34
Figure 2-9: Filter Responses for 6-Tap Daubechies QMF	36
Figure 2-10: Transfer Functions for 6-Tap Daubechies QMF	37
Figure 2-11: Transfer Functions for 6-Tap Daubechies QMF	37
Figure 2-12: Block Diagram of the Synthesis Stage ($M = 3$) of a DWT	38
Figure 2-13: Block Diagram of the Analysis Stage ($M = 3$) of a 2-D DWT	39
Figure 2-14: Spatial Orientation of the Wavelet Coefficients in 2-D	40
Figure 3-1: VPIC 14-Visual Pattern Code Vectors	47
Figure 3-2: A View of the VPIC 28-Pattern Codebook	51
Figure 3-3: GenCodBk Algorithm	52
Figure 3-4: VPIC Block Diagram	56
Figure 3-5: 0° VPIC Codebook Vectors	60

Figure 3-6: Frequency Response of $w[n]$	62
Figure 3-7: Frequency Response of $w[n]$ in Terms of $20 \log W(e^{j\omega}) $	62
Figure 3-8: Original 512 x 512 Test Image <i>Lena</i>	67
Figure 3-9: Reconstructed 512 x 512 Test Image <i>Lena</i> from Trial 7 (Spatial VPIC, PSNR = 28.28 dB)	68
Figure 3-10: Original 512 x 512 Test Image <i>Mandrill</i>	69
Figure 3-11: Reconstructed 512 x 512 Test Image <i>Mandrill</i> from Trial 7 (Spatial VPIC, PSNR = 21.80 dB)	70
Figure 3-12: Original 512 x 512 Test Image <i>Clown</i>	71
Figure 3-13: Reconstructed 512 x 512 Test Image <i>Clown</i> from Trial 7 (Spatial VPIC, PSNR = 27.93 dB)	72
Figure 3-14: Reconstructed 512 x 512 Test Image <i>Lena</i> with Postfiltering (Trial 7) (Spatial VPIC, PSNR = 27.98 dB)	73
Figure 4-1: Transform Coefficients of a Level $M = 3$ Wavelet Decomposition of <i>Lena</i>	78
Figure 4-2: Reconstructed Test Image of <i>Lena</i> for Trials 1, 10, 12 and 13	81
Figure 4-3: Local Gradient Uniform Quantizer	85
Figure 4-4: Subband Histograms for a Level $M = 3$ Wavelet Decomposition of <i>Lena</i>	87
Figure 4-5: Block Diagram of the Compressed/Decompressed Image Coding Process	90
Figure 4-6: The 64K-Pattern WVPIC/MAE BTC Block Diagram	94

Figure 4-7: Initialization of the WVPIC/MAE BTC Codebook	101
Figure 4-8: Reconstructed Test Image <i>Lena</i> for Trial 10 Input Parameters (PSNR = 25.18 dB, C = 10.7)	104
Figure 4-9: Effect of Tresholding <i>high</i> and <i>low</i> $\nabla_{\min}$ Settings	105
Figure 4-10: Reconstructed Test Image <i>Lena</i> for Trial 13 Input Parameters (PSNR = 28.32 dB, C = 12.7)	107
Figure 4-11: Reconstructed Test Image <i>Lena</i> for Trial 14 Input Parameters (PSNR = 6.32 dB, C = 13)	109
Figure 4-12: Block Diagram of the Modified 64K-Pattern WVPIC/MAE BTC Coding Scheme	113
Figure 4-13: Reconstructed Test Image <i>Lena</i> for Trial 7 Input Parameters (PSNR = 28.64 dB, C = 12.3)	116
Figure 4-14: The 256-Pattern WVPIC/MAE BTC Enhanced Texture Patterns	118
Figure 4-15: Reconstructed Test Image <i>Lena</i> for Trial 7 Input Parameters (PSNR = 27.61 dB, C = 13.2)	121
Figure 4-16: Reconstructed Test Image <i>Lena</i> for Trial 6 Input Parameters (PSNR = 27.23 dB, C = 15.3)	123
Figure 4-17: The Performance of the Different WVPIC/MAE BTC 4 x 4 Block Segmentation Implementations Using Test Image <i>Lena</i>	125
Figure 4-18: Reconstructed Test Image <i>Lena</i> for Trial 10 Input Parameters (PSNR = 30.22 dB, C = 7.2)	130

Figure 4-19: The Performance of the $2 \times 2/4 \times 4$ WVPIC/MAE BTC	
4×4 VPIC Coding Schemes.....	131
Figure 4-20: Ternary Coding of a Given 4×4 Image Block Wavelet	
Transform Coefficients	134
Figure 4-21: Transform Coefficients of an Edge Block Sorted from <i>min</i> to <i>max</i>	136
Figure 4-22: Three-Level WMME BTC Block Diagram.....	139
Figure 4-23: Wavelet Transform Coefficients Values of a Given $n \times n$ Image	
Block Where $n = 4$	141
Figure 4-24: Reconstructed Test Image <i>Lena</i> for Trial 15 Input Parameters	
(PSNR = 29.79 dB, C = 11.8)	147
Figure 4-25: Reconstructed Test Image <i>Lena</i> for Trial 12 Input Parameters	
(PSNR = 31.39 dB, C = 10.17)	150
Figure 4-26: Overall Three-Level WMME BTC Coding Scheme Performance	151
Figure 4-27: Transform Coefficients of an Edge Block Sorted from <i>min</i> to <i>max</i>	153
Figure 4-28: Four-Level WMME BTC Block Diagram	155
Figure 4-29: Reconstructed Test Image <i>Lena</i> for Trial 12 Input Parameters	
(PSNR = 31.88 dB, C = 7.74)	162
Figure 4-30: Reconstructed Test Image <i>Lena</i> for Trial 14 Input Parameters	
(PSNR = 30.50 dB, C = 13.9)	163
Figure 4-31: Three-Level and Four-Level WMME BTC Performance for	
4×4 Block Segmentation	164

Figure 4-32: Reconstructed Test Image <i>Lena</i> for Trial 11 Input Parameters (PSNR = 30.35 dB, C = 15.5)	167
Figure 4-33: Three-Level and Four-Level WMME BTC Performance for 2 x 2 Block Segmentation	168
Figure 4-34: Reconstructed Test Image <i>Lena</i> for Trial 1 Input Parameters (PSNR = 31.09 dB, C = 11.6)	170
Figure A-1: Schematic Diagram for Compression/Decompression Processes Used in this Thesis	198

List of Appendices

Appendix A: Source Code	184
Appendix B: Compression/Decompression Process.....	198
Appendix C: Source Code Examples.....	200

CHAPTER 1

Introduction

An image represented by, say, a rectangular array of 512 by 512 picture elements (pixels), each requiring 8 bits per pixel, contains over two million bits: a rather large number, and a problem for transmission and storage facilities. The goal of image compression is to reduce, as much as possible, the number of bits necessary to represent and reconstruct a faithful duplicate of the picture [1]. Compression can be achieved because the data originating from an image are not random. For example, adjacent samples in a monochrome image have similar grey values, thus exhibiting important spatial correlations, which may be exploited to reduce the size of the representation [2].

Recent progress in the study of the Human Visual System (HVS) has opened new vistas in image processing [3]. By taking advantage of the directional sensitivity of the neurones in the visual pathway, various image coding schemes using pattern matching has led to a new class of image compression methods [4]. By using the pattern matching approach in the spatial domain, the number of bits in an image can be reduced compared to performance attained by first generation coding techniques.

Image coding can also be done in the frequency or transform domain. Individual transform coefficients tend to be uncorrelated, with the signal energy distributed among

only a small set of transform coefficients, and hence are more easily and efficiently coded than the highly correlated pixels in the spatial domain. In addition, the human visual system is less offended by coarse quantization of high frequency components than of low frequency components, which are readily available in the transform domain.

Current compression standards such as JPEG, px64 and MPEG all use frequency domain coding [5]. They rely on the Discrete Cosine Transform (DCT) to produce a frequency domain version of the image. These techniques use the DCT applied to small blocks, typically 8 by 8 pixels. This block segmentation of the image produces frequency coefficients which are uncorrelated within the block, but which are correlated from block to block. Presentation of a specific "harmonic" in a spatially correct juxtaposition reveals an image or a set of correlated patterns in which features are clearly recognizable. A similar phenomenon occurs as a result of wavelet transformation. A 3-level wavelet decomposition of the familiar image *Lena* is shown in Figure 1-1. The image was decomposed using a Wavelet Transform based on a six-tap Daubechies Quadrature Mirror Filters (QMF) implemented as part of this research [30]. The visual correlation between the subband components in each subband is clearly identifiable. It seems reasonable then to consider improvements that can be made to existing compression algorithms by exploiting the apparent correlations that exist after transformation into the frequency domain.

Figure 1-1: A 3-Level Wavelet Decomposition of Test Image *Lena*.

1.1 Thesis Topic

This thesis examines two hypotheses. The first hypothesis that is explored is that high compression ratios can be achieved by utilizing the apparent correlation between subband frequency (wavelet) components. The subband "image", except for the zero frequency

components, is the output of a band-pass filter, and the largest outputs occur where the frequency component has most energy, that is, at edges. Calculations show that the statistical correlation between adjacent coefficients is very low (essentially zero), and yet spatial relations are very visible.

Typical compression algorithms, such as JPEG, which are based on a minimum Mean Squared Error (MSE) quantization criterion, discard low energy frequency components. These are usually the high frequency components which, typically, represent edges. Intuitively, this seems like the wrong thing to do. The human visual system "likes" sharp edges. In fact, in the printing industry, it is customary to emphasize edges using a technique called "unsharp masking" [6]. The edge information is contained in the subbands, and it seems reasonable to assume that edge information should be preserved and not discarded, as it often is in conventional minimum mean squared error compression algorithms.

The second hypothesis is that the edges are patterns, and that a pattern matching technique, such as the relatively new image coding technique called *Visual Pattern Image Coding* (VPIC) [4], could be used to represent the subband images.

A series of investigations was carried out on images decomposed using a six-tap QMF-based Wavelet Transform. After implementation of VPIC in the spatial domain, the investigations carried out the application of VPIC in the transform domain, and then went on to consider various multi-level VPIC combined with Block Truncation Coding (BTC) technique-based implementations in search of higher compression ratios and image quality than standard VPIC.

The multi-level VPIC/BTC image coding schemes developed as part of this thesis are newly designed, implemented, and published for the first time.

1.2 Achievements

In this thesis, several coding techniques based on pattern matching operating in the spatial and wavelet domains are introduced. The efforts spent to develop and implement the coding techniques presented in this thesis include:

- a. The successful C program implementation and testing of the Stanford Subband algorithm on a UNIX station at Carleton University. This algorithm was used for transforming the test images from the spatial domain into a wavelet domain.
- b. The development, testing and implementation of the VPIC algorithm introduced by Chen and Bovik;
- c. The development, testing and implementation of several types of multi-level Wavelet VPIC/BTC-based image coding algorithms;
- d. The development, testing and implementation of support utilities related to the processing of subband images in the UNIX and DOS environments.

These include:

- i) the capability to convert subband images from UNIX-based *big-indian* integer format to DOS-based *little-indian* integer format, and vice versa;
- ii) the capability to convert subband image transform coefficients from *integer* to *float* format, and vice versa;

- iii) the capability to disassemble and assemble subband images from one another for the processing of each subband image independently;
- iv) the capability to display byte, integer and float format subband images on UNIX platforms;
- e. The development, testing and implementation of statistical and analysis algorithms to learn about the statistical properties of subband images in the wavelet domain; and
- f. The development, testing and implementation of multi-size codebook generation algorithms required for the various coding techniques developed in this research.

All of the algorithms developed as part of this thesis were written in C or C++ for a UNIX work-station or an IBM/PC Pentium. The compilation of enough information to develop and implement the VPIC algorithm which served as the basis in this research was a difficult task. The scientific paper dealing with the VPIC coding scheme was found to be incomplete in description causing implementation delays, and thorough testing was required to ensure full functionality of the algorithm as published. Key pieces such as the codebook code vectors computation was missing. Additional information on the visual performance of the VPIC algorithm was never published, hence no basis related to the visual performance of VPIC was available. The implementation of the image coding techniques presented in this thesis resulted in the production of **ninety nine thousand** lines of source code.

1.3 Results Overview

The implementation of the standard VPIC in the spatial domain was performed successfully with results comparable to those published by Chen and Bovik. In the wavelet domain, modified versions of the standard VPIC and Moment Absolute Error (MAE) BTC quantization schemes were integrated to form a more versatile pattern-based image coding architecture. The implementation of such a coding scheme in the wavelet domain, called WVPIC for Wavelet VPIC, was also a success. The overall results obtained in terms of visual quality and compression ratio were comparable when using 4×4 block truncation to that obtained by standard VPIC. With the 2×2 block scheme, the results were even better.

With WVPIC, a relatively high compression ratio was easy to achieve while a high Peak Signal-to-Noise Ratio (PSNR) was not. In addition, *blockiness* and *stripe* artifacts were very obvious, especially with the 4×4 design. The visual quality of the reconstructed images could not be improved even with more edge blocks detected and coded. Furthermore, post-processing filtering is required to filter the blockiness artifacts around the uniform patterns, thus increasing the processing time of the system.

As a result, another coding scheme involving WVPIC was developed and implemented which permitted better edge preserving characteristics. This coding scheme combined a modified version of WVPIC with multi-level Minimum Maximum Error Block Truncation Coding (MME BTC), which provided a much clearer and sharper reconstructed image at comparable compression ratios. The overall complexity of the multi-level MME BTC in the wavelet domain is less complex than standard VPIC, and

can be implemented easily in any hardware system. Its compression performance can be enhanced further by incorporating other coding techniques, such as entropy coding.

1.4 Thesis Organization

The material presented in this thesis is as follows. Chapter 2 contains background information on image data compression. The first section provides a general overview of digital image coding. This includes a definition of the digital image coding and the many first and second generation image coding schemes that have been developed over the past few decades. The next section provides various details inherent to image coding applied in the spatial domain. This is followed by an introduction to four popular image coding techniques which are based on a concept called *pattern matching*. The final section in Chapter 2 introduces what is becoming a very popular transform, the *wavelet transform*. An overview of the wavelet transform used in this thesis is presented in its one- and two-dimensional forms.

Chapter 3 describes the *Visual Pattern Image Coding* (VPIC) technique in detail, which is the basic coding scheme used in this thesis. The first section provides theoretical basis of VPIC. This includes some background information on VPIC as well as some justification from the properties of biological vision. The next section of this chapter presents an overview of VPIC pattern design, and a description of how the VPIC codebook is generated. Also, some notion of gradient thresholding is provided. In the last section, an in-depth description of the VPIC architecture developed and implemented as part of this thesis is presented.

Chapter 4 provides implementations of the various binary and multi-level image pattern matching coding algorithms developed in this thesis. The first section provides some of the advantages that pertain to the coding of images in the wavelet domain as oppose to spatial domain. In the second section, the development of several versions of standard VPIC applied in the wavelet domain are presented in detail, including the results obtained in terms of Peak-Signal-Noise Ratio (PSNR) and compression ratio. The algorithms presented in this section are based on what is called the WVPIC/MAE BTC architecture, developed and implemented as part of this thesis. In the third section, Ternary Coding is introduced. It provides a thorough description of the Three-Level WVPIC Minimum Maximum Error (WMME) BTC coding architecture developed and implemented in this research, including the results achieved. The fourth and final section provides a thorough description of the Four-Level WMME BTC coding architecture developed and implemented, including results.

Chapter 5 presents a summary of the results obtained with all the coding schemes introduced in this thesis. A comparison is made from all the performance achieved by the coding algorithms. A summary and conclusion is also presented, as well as areas of future work which could contribute to this field are discussed.

CHAPTER 2

Image Data Compression

2.1 General

The digital representation of an image requires a very large number of bits. The goal of image coding is to reduce this number as much as possible, while retaining enough information to reconstruct a faithful duplicate of the original picture. Many first generation image coding techniques, such as Pulse-Code Modulation, Predictive Coding, and Delta Modulation, were based on statistical signal and source coding theory. Lossless and lossy techniques which operated in the spatial or space domain were developed. These techniques were either fixed or adaptive. The following is a list of the major first generation image coding techniques developed [2], [7], [8], [9]:

Spatial Methods	Transform Methods
Pulse-Code Modulation (PCM)	Karhunen-Loève Transform (KLT)
Predictive Coding	Discrete Fourier Transform (DFT)
Differential Pulse-Code Modulation (DPCM)	Discrete Cosine Transform (DCT)
Delta Modulation	Walsh-Hadamard Transform (WHT)
Interpolation Coding	
Bit-Plane Coding	

Table 2-1: Major First Generation Image Coding Techniques.

Hybrid methods which combine spatial and transform techniques were also developed in an attempt to achieve higher compressions. As well, codes generated by these techniques were often entropy coded, i.e. represented by Huffman codes. The compression ratios attained by using the above techniques reached a saturation level of around 10:1 during the late 1970s. At that point in time, the upper bound given by the entropy of the source was still not reached. A major drawback of early compression methods is that they do not take into account what the human eye sees [10].

Some of the second generation image coding techniques developed are Vector Quantization (VQ) and Block Truncation Coding (BTC) as described in references [11] to [17]. In addition to using coding theory as their foundation, they rely heavily on the way the Human Visual System (HVS) processes images. As we will see later in this thesis, coding schemes matched to the HVS by imitating its known functions, in the processing of contours, textures and pattern matching (edges) allow for higher compression ratios than the early image compression techniques. Thus, second generation techniques use an initial step where the image data are converted into some sort of messages, such as pre-defined patterns, and then these patterns are coded as per first generation coding theory. Typical compression ratios achieved by these methods are as high as 70:1, however, image visual quality remains an important problem to investigate.

Since pattern matching and edge preserving in the transform domain is the main focus of this thesis, some of the pattern detection concepts used traditionally in the spatial domain by second generation compression algorithms will be introduced in the sections to follow. These include thresholding, edge detection, gradient computing and pattern

matching. Also, an introduction to VQ and BTC image coding schemes will be presented which will help the reader understand the introduction of a relatively new coding technique called *Visual Pattern Image Coding* (VPIC) developed by Bovik and Chen [4].

VPIC is an hybrid image coding method somewhat between VQ and BTC. It is based primarily on pattern matching and edge preserving, and is a two-level coding technique. Since some multi-level image coding algorithms are developed in this thesis, some notions about multi-level coding are also presented.

As discussed in Chapter 1, this investigation consists mainly of the application of VPIC in the transform domain. Consequently, a special kind of wavelet transform was chosen to be the main transform, and therefore, section 2.3 is dedicated to an introduction of wavelet transforms.

2.2 Spatial Domain

2.2.1 Thresholding

Thresholding can be defined as being the process of identifying a pixel or a group of pixels which exceed a pre-specified or adaptively determined value (threshold) [53]. One of the simplest cases of thresholding is in the situation where a single threshold value is considered. Each pixel in an image is examined. If a pixel has a value greater than or equal to a pre-defined threshold, the corresponding output pixel is identified as a high, otherwise it is a low. This operation can be defined as follow:

$$\mathbf{R}_i = \begin{cases} \mathbf{H}, & I_i \geq T \\ \mathbf{L}, & I_i < T \end{cases}$$

Here, I_i is the pixel, R_i is the output pixel, T the threshold value, H the high or above-or-equal-to-threshold result value, and L the low or below-threshold result value. This kind of thresholding is often used to separate an object like an edge or subimage from its background. So far, it has been assumed that the threshold T is simply scalar. In this thesis, a special type of thresholding is used, and it is based on the notion of the *gradient* of subimages or blocks of pixels. This notion is discussed further in section 2.2.3.

2.2.2 Edge Detection

Edge detection using thresholding is one of the commonest approaches used in second generation compression techniques [18]. Edge detection relies on the observation that discontinuities in gray levels or brightness or colour are often associated with physical boundaries between objects, or boundaries between different parts of the same object. Consequently, dividing an image based on these discontinuities should result in the detection of edges or visually important patterns, which, in fact, correlate strongly with complete objects, and/or with important parts of objects.

2.2.3 Gradient

Many edge detectors estimate the direction of the edge as well as its strength. For example, gradients are inherently vectors, indicating magnitude and direction of change. Gradients measure how quickly pixel values change when moving across an image. If a

particular part of the image is smooth with gently changing gray level, the gradients will be small. In contrast, if the image contains patterns, the gradients will be large [45].

From a mathematical point of view, the gradient might be considered as the *first derivative* of the image. In an image $I(x,y)$, the gradient magnitude ∇ and direction θ of the gradient are related to the x and y coordinates, where,

$$\nabla = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2} \quad 2-1$$

$$\theta = \tan^{-1} \left(\frac{\frac{\partial I}{\partial y}}{\frac{\partial I}{\partial x}} \right) \quad 2-2$$

The approximation of gradients is inherently noisy in areas where neighbourhood pixels have very similar values, such as uniform grey level regions. One way to alleviate the noise problem is by computing the gradient on a group of pixels in the form of blocks instead of computing the gradient using just two adjacent pixels. Another way to reduce noise artifacts is by performing some kind of gradient thresholding. For example, gradient thresholding applied on image blocks differentiates uniformly distributed local images from nonuniformly distributed ones which reflect *edge blocks*, i.e., blocks containing edges. By computing the gradients on edge blocks only, the overall noise artifacts in an image are vastly reduced.

2.2.4 Pattern Matching

Patterns or image features are identifiable components of images, and can be detected in the spatial or transform domains. There are a variety of feature extraction techniques, and one of the most important feature extraction techniques is based on *shape* and *orientation*. Shape and orientation offer a very direct correspondence between an image block and a pre-defined pattern. A very popular approach that uses the notion of shape and orientation is *template matching* or *pattern matching*, which can be applied once the local image is converted to binary (two level) format. For example, let's partition a 512 x 512 image of *Lena* into blocks of 4 x 4 pixels. This amounts to a total number of 16,384 blocks. A block chosen at random, say, block $b[4328]$, has the following values:

$$b[4328] = \begin{bmatrix} 223 & 206 & 103 & 17 \\ 221 & 202 & 88 & 19 \\ 220 & 198 & 63 & 25 \\ 223 & 180 & 36 & 21 \end{bmatrix}$$

It can be challenging to determine the kind of pattern $b[4328]$ represents by a simple examination. However, by computing its binary format, the job of determining its local pattern becomes trivial. The binary format of an image block is calculated by determining the polarity of what is called its *residual*. The *residual* $r[4328]$ is computed as:

$$r[4328] = b[4328] - u[4328]$$

where $u[4328]$ denotes the mean of $b[4328]$.

$$r[4328] = \begin{bmatrix} 223 & 206 & 103 & 17 \\ 221 & 202 & 88 & 19 \\ 220 & 198 & 63 & 25 \\ 223 & 180 & 36 & 21 \end{bmatrix} - 127.8125$$

$$r[4328] = \begin{bmatrix} 95.1875 & 78.1875 & -24.8125 & -110.8125 \\ 93.1875 & 74.1875 & -39.8125 & -108.8125 \\ 92.1875 & 70.1875 & -64.8125 & -102.8125 \\ 95.1875 & 52.1875 & -91.8125 & -106.8125 \end{bmatrix}$$

The conversion of the *residual* to its equivalent binary format (or bit map) is accomplished as per the following: any residual value greater than or equal to 0 becomes a “+”, otherwise, it becomes a “-”.

$$bitmap[block] = \begin{cases} +, r_{i,j} \geq 0 \\ -, r_{i,j} < 0 \end{cases}$$

Thus, $bitmap[4328]$ becomes,

$$bitmap[4328] = \begin{bmatrix} + & + & - & - \\ + & + & - & - \\ + & + & - & - \\ + & + & - & - \end{bmatrix}$$

Examining $bitmap[4328]$, it is now easy to see that this 4 x 4 block of information corresponds to an edge with a vertical orientation. In hexadecimal format, this pattern corresponds to $0xcccc_{16}$, and it is determined as followed. Each row in the 4 x 4 block corresponds to a hexadecimal value where a “+” sign coincides to a binary bit “1”, and a

“-” sign coincides to a binary bit “0”. And so, the first row of *bitmap*[4328] corresponds to the binary number 1100_2 , which in turn corresponds to the hexadecimal value $0xc_{16}$. Doing the same for the remaining rows in the block, one obtains a pattern that corresponds to the hexadecimal number $0xcccc_{16}$.

Sometimes, it can be easier to code patterns in the transform domain than in the spatial domain. For example, patterns that are harder to identify due to surrounding pixel intensities in the image domain tend to stand out in the wavelet domain. Patterns such as large bar shapes are displayed as edges in the wavelet domain. A coding algorithm working on 4×4 image blocks might perceive an 6×6 bar shaped pattern as an uniform block if and only if the whole bar shaped pattern happens to fit exactly in a 4×4 block. As a result, the bar shaped pattern is reconstructed as an uniform block, not an edge block. If it does not happen to fit exactly in a 4×4 block, then it is coded as two edge patterns instead because the first 4×4 block reflects the front of the bar shaped pattern, and the next 4×4 block reflects the rear.

Looking at the pattern circled in Figure 2-1 (a), it can be seen how it appears in the wavelet domain; it is transformed into an edge pattern. The pattern circled in Figure 2-1 (a) has a minimum width of seven pixels, and the pattern circled in Figure 2-1 (b) has a maximum width of two pixels. The pattern in the wavelet domain will be coded as one edge pattern only, as opposed to an uniform pattern or two edge patterns.

Figure 2-1: Pattern Relationship Between Spatial and Wavelet Domains.

2.2.5 Vector Quantization (VQ)

In the following sections, three second generation compression techniques are presented. They are Vector Quantization (VQ), Block Truncation Coding (BTC) and Visual Pattern Image Coding (VPIC). These techniques all exploit pattern matching schemes, and are similar in that the image to be coded is first partitioned into a large number of smaller $n \times n$ subimages or blocks. Each $n \times n$ block is then coded either by extracting certain intrinsically high-information quantities which are transmitted in reduced form (as in BTC), or by making a comparison between the block and a set of preordained vectors or typical subimages, the index of the closest vector being transmitted instead of the original block (as in VQ). VPIC is a mixture of both the BTC and VQ compression techniques, and its main advantage is its ability to preserve edge patterns using relatively small size codebooks.

The VQ approach [11], [12], [13], is based on a *codebook*, usually generated by iterative procedures, which contains side information on typical vectors that represent patterns. VQ generally requires three major steps; vector formation, codebook

generation, and quantization [19]. In the coder, Hamming or Euclidean distances or a minimum distortion based rule, can be used to select the best matched code vector to represent a given subimage. In effect, it represents the minimum measured distortion between the image block and the code vector. In the decoder, VQ is usually realized by simple Look-Up-Table (LUT) operations. Compression is obtained by using a LUT-codebook with relatively few code vectors compared to the number of possible image vectors.

The bit rate of a VQ scheme is $R = \log_2 N_c/n$ bits per pixel (bpp) where N_c represents the dimension of the codebook, and n represents the dimension of each block. The key element in VQ is the generation of the codebook, either *local* or *global*, and its searching structure which must be inherently efficient. The main disadvantages of this approach are the substantial number of comparison computations per pixel (code vector search) required at the encoding end, and the extensive codebooks which are either image dependent or domain dependent (computed from a typical set of images).

In addition, the fact that the code vectors greatly affect the visual quality of the coded images, the method and coding complexity in generating visually efficient codebooks are very demanding. A typical block diagram of the basic VQ structure is shown in Figure 2-2:

Figure 2-2: VQ Block Diagram.

2.2.6 Block Truncation Coding (BTC)

BTC, as described in [1], [14], [15], [16], [17], [19], and [20] is essentially a 2-level (1-bit) moment-preserving quantizer which preserves certain statistical moments of small non-overlapping blocks of pixels of the input image. Some of the minimum distortion schemes used in BTC are the mean and standard deviation of each block [14], Mean Squared Error (MSE) [15], Mean Absolute Error (MAE) [17], and Minimize Absolute Error [20].

The advantages of BTC are pixel resolution, edge preservation, and low computational complexity. The data transmitted by a BTC coder are a binary plane (bit map) which defines the quantization level for each pixel, and two 8-bit parameters for each block, e.g., the mean of the block and its standard deviation as per [14]. Decoding is the simple process of placing the appropriate reconstruction value at each pixel location as per the bit map. A block diagram of the BTC scheme is presented in Figure 2-3.

The BTC algorithm used in this thesis is a derivative of the MAE method described in [17]. The pixels in each block are quantized into high-mean and low-mean, minimizing the mean absolute error in each block. The output data set includes a bit map specifying the high and low state of each pixel, high-mean, and low-mean for each block.

Figure 2-3: **BTC Block Diagram.**

Differentiation between uniform and edge image blocks in this work is performed through the use of gradient thresholding, which is based on the local gradient rather than the local mean. The key points in MAE BTC are the moment-preserving quantizers, source coding of the bit map and the reconstruction levels to further reduce the bit rate than in standard BTC.

2.2.7 Visual Pattern Image Coding (VPIC)

In 1990, a new image coding scheme consistent with the properties of biological vision was published by Bovik and Chen [4]. This compression method is based on a pattern matching approach called Visual Pattern Image Coding (VPIC), and performs well when used in the spatial domain. The authors demonstrated that considerable image compression can be attained. VPIC operates by coding the image using a small set of independent visual patterns which are localized subimages containing visually important information.

Although VPIC is closely related to BTC and VQ, there are important differences. First, there is no training phase required in codebook generation since the visual patterns used in VPIC are derived from relevant psychophysical and physiological properties of the HVS [16], and are thus adaptable to any kind of images. Secondly, the assignment of patterns to local image blocks is not based on a standard MSE, but on bit map correlation matching and on the orientation of the image patterns. It is computationally efficient. According to Bovik and Chen [4], VPIC offers the following:

- a. excellent image quality agreeing with perception;
- b. compression ratios comparable to or exceeding those offered by VQ; and
- c. vastly reduced coding complexity (nearly two orders of magnitude faster than standard VQ and BTC).

The authors claim that VPIC is feasible for real-time applications without requiring special-purpose hardware, and performs at or above VQ in terms of compression ratio and image quality. Illustrated at Figure 2-4 is a block diagram of VPIC. The internal functionality of the VPIC architecture will be presented in more detailed in Chapter 3.

Figure 2-4: A Simplified Block Diagram of VPIC.

2.2.8 Multi-Level Image Coding (MLIC)

In this section, a brief introduction to MLIC is presented since it will be presented in more detailed in Chapter 4. The VPIC and BTC coding techniques briefly introduced in sections 2.2.6 and 2.2.7 are basically *two-level* moment-preserving quantizer that adapts to local properties of the image. They have the advantages of preserving edges and having low complexity.

Some scientific images such as satellite remote sensing, Synthetic Aperture Radar (SAR), and medical imagery contain very-high resolution artifacts coinciding with sharp edges and sudden single-pixel-width variations in gray-level which can be crucial in the image application and must be preserved [21]. The fact that these images require massive storage space and substantial transmission times when communicating such images over data networks, a way to compress such images while maintaining high fidelity must be utilized.

As we will see later in this thesis, a MLIC coding achitecture is required to code highly edges busy images with satisfactory fidelity. And so, instead of using two-level patterns in the coding process, multi-level patterns can be used to represent each edge block detected in an image more adequately.

2.3 Wavelet Domain

2.3.1 General

As signal energy tends to be compacted into a few transform coefficients via a given transform, storage and/or transmission of digitized images is much more efficient in the

transform domain than in the original data (spatial) domain [22]. Which transform should be used is another question? The Discrete Fourier Transform (DFT) and the Discrete Cosine Transform (DCT) are two examples of well established tools in signal and image processing. The advantage that the DCT has over the DFT is that the DCT generates real values only [40]. However, the main drawbacks of the DCT are blocking effects and mosquito noise well documented in [23]. These drawbacks are mainly due to quantization errors.

Additionally, it might be of interest to know how well the DFT and DCT mimic the Human Visual System (HVS). As the HVS is generally considered to be an optimal image processor [41], it makes sense to use a transform that processes images as close as possible to the HVS. In the physiology of human vision [24], evidence has been gathered showing that the retinal image is *decomposed* into several spatially oriented frequency channels. Thus the processing of images is carried out in the frequency domain in the form of subbands.

This is where wavelet transforms come into play. Subband coding, using a wavelet transform implementation, is an attractive image coding method since its hierarchical structure is somewhat similar to that of the HVS [8]. Additionally, a wavelet transform has the potential capability for reducing the blocking effect and the mosquito noise which are the two main disadvantages of conventional transform coding such as the DCT [23], [25].

One practical form of a wavelet transform is the Discrete Wavelet Transform (DWT). The DWT is usually treated as subband coding, since wavelet transforms can be realized using bandpass filter banks.

A wavelet transform is better able to *zoom in* or *zoom out* on very short-lived high frequency phenomena, such as transients in signals or singularities in functions. It is essentially a tool that *cuts* up data or functions into different frequency components (subbands), and then analyses each component with a resolution matched to its scale [26], [27].

The concept of subband coding can be better appreciated by assigning higher processing priority to subbands with more energy content. For example, the low order frequency subbands contain most of the total energy in an image [28]. Consequently, by discarding the high order subbands in the reconstructing process, it is expected (as seen later) that very little or negligible reconstruction error occurs once the analysis-synthesis operation is done. Typically, a compression ratio of 4:1 can be achieved when high order subbands are discarded prior to the encoding process.

In this work, a wavelet transform is used as the main transform vehicle for mapping the images from the spatial domain to the wavelet domain. A wavelet transform is used for the following reasons:

- a. Images involve essentially two types of signals: First, *stationary* signals, that is, signals whose properties do not evolve in space. Such properties are reflected in image regions which have uniform distributions. For such signals, the natural stationary transform is the well-known Fourier Transform.

Second, *non-stationary* signals whose properties do evolve in space. Such properties are reflected in regions which have non-uniform distributions such as edges and visual patterns in an image. Here, an analysis adapted to non-stationary signals requires more than the Fourier Transform. An approach is to introduce time dependency in the Fourier Transform while preserving linearity. This can be done by using the Short-Time Fourier Transform (STFT) or a wavelet transform;

- b. Another reason for using wavelet transforms is to perform an analysis on whether or not the wavelet subbands can be encoded more efficiently than the original image using pattern matching coding techniques. Once the subbands are obtained through the use of a wavelet transform, different pattern matching coding techniques can be used for each subband, thus taking advantage of the properties of each subband;
- c. According to Mutsumi *et al.* [23], orthonormal wavelet transforms can be used to resolve the blockiness effect and mosquito noise otherwise obtained by using the DCT for example. As part of this work, this can be verified;
- d. Wavelet transforms allow for natural multifrequency channel decompositions to images in a way similar to the human perception system [28]. The time-frequency characteristic of a wavelet transform are a natural match to some of the properties of the HVS [29]. It is therefore logical to use a transform that naturally mimics the structure of the visual system;

- e. As stated earlier, a compression ratio of 4:1 can be achieved initially by discarding higher frequency subbands which contribute very little to the visual quality of the overall image once reconstructed. This is to be verified; and
- f. From an efficiency point of view, a Discrete Wavelet Transform (DWT) is computationally less intense than the Discrete Cosine Transform or the Fast Fourier Transform. An FFT of length N (where N is a power of 2), requires roughly $N \log_2 N$ operations, while a DWT of length N requires approximately N operations [40]. So it makes sense to use a transform algorithm that is computationally fast.

2.3.2 Introduction to Wavelet Transform

Wavelet transforms have apparently been used in the 1930s and 1940s by other researchers in a somewhat disguised form [26], but really gained popularity a couple of decades later. In the early 1980s, the wavelet transform family was shown to be as powerful and versatile as the Fourier Transform.

The French mathematician, Yves Meyer, recognized the wavelet theory to be part of the field of Digital Signal Processing (DSP), and produced a family of wavelets that he proved were most efficient for modeling complex phenomena [30]. This work was then improved upon by other well known researchers such as Mallat [28], Chiu [30], Mathieu [31], Daubechies [32] and many others. Since 1988, there has been a lot of research done in this area where the wavelet transforms found their way in applications such as image decomposition, reconstruction, analysis and compression.

The basic idea behind wavelet theory is to decompose a given signal into elementary building blocks called *wavelets* which have the property that:

- a. high frequency elements correspond to sharp time localization; while
- b. low frequency elements have sharp frequency localization.

2.3.3 Time-Frequency Analysis

Wavelet theory provides various techniques such as multiresolution processing used in computer vision, and subband coding for image compression. It provides an alternative to the classical Short-Time Fourier Transform (STFT) which was used to measure the local frequency content of a signal. The basic difference between the STFT and a wavelet transform is as follows. The STFT uses sines and cosines (waves) multiplied by sliding windowed signals which map a signal into a two-dimensional function in a time-frequency plane. Its main drawback is once a window is defined for the STFT, the time-frequency resolution is fixed over the entire time-frequency plane. Thus components of a signal can be analyzed with good time resolution or frequency resolution, but not both.

A wavelet transform uses short windows at high frequencies and long windows at low frequencies by means of *translations* and *dilations*. It overcomes the resolution limitation of the STFT by using a basis function (basic wavelet) that allows for the resolution to vary in time and frequency in the time-frequency plane at the same time. The resolution variations are controlled by the scaling parameter in the wavelet function. This produces a multiresolution analysis of a given signal.

2.3.4 The Continuous Wavelet Transform

In wavelet theory, the window equivalent of the STFT is an oscillating function and is called a *basic wavelet* $h(t)$. Any basic wavelet can be implemented as a filterbank. It is represented by the impulse response $h(t)$ of the filter bank. Based on certain predefined conditions of the basic wavelet $h(t)$, it can be implemented to have a rapid decay at infinity, smooth behavior, and to satisfy the fundamental condition (equ. 2-1), which means that in some weak sense $h(t)$ is oscillating.

$$\int_{-\infty}^{\infty} h(t) dt = 0 \quad 2-1$$

A wavelet transform amounts to projecting a signal using a family of elementary functions such that these functions are obtained by *translating* and *dilating* a single basic function. The Continuous Wavelet Transform (CWT) is more specifically the *convolution* of some signal $x(t)$, with a translated and dilated wavelet:

$$\text{CWT}_x(\tau, a) = \frac{1}{\sqrt{|a|}} \int x(t) h\left(\frac{t-\tau}{a}\right) dt \quad 2-2$$

All impulse responses of the filter bank are defined as *scaled* versions of the basic wavelet $h(t)$, where a is the *scale* factor and τ is the *translating* factor. The constant $1/\sqrt{|a|}$ is simply used for energy normalization [28]. The scale factor a has the same notion as the scale in maps. Very large scales mean global views, while very small

scales mean detailed views. Thus as a changes, the basic wavelet covers different frequency ranges or is tuned to different bandwidths. Referring to equation 2-2, a large scale parameter corresponds to *contracted* signals (small frequencies), while a small scale parameter corresponds to *dilated* signals (high frequencies) [30]. The translating factor τ dictates the rate at which the image is analyzed.

Another notion related to scaling is the *resolution* of a signal. The resolution is also related to its frequency content. The resolution, in a filter implementation, is controlled by the filter itself. For example, when filtering a signal through a lowpass filter, the scale of the signal stays the same, but its resolution is reduced. The reduction in resolution of the signal is essentially caused by the loss of high frequency details.

Now let's see the effects of scaling and resolution of a discretized signal when applied through a lowpass filter followed by downsampling. Because a discretized signal is passed through a lowpass filter, a reduction in the resolution of the signal takes place (Figure 2-5[a]). Increasing the scale involves downsampling (keeping every other output sample of the filter, and discarding the others). This process is called *decimation* which refers to the convolution of a signal with the impulse response of a lowpass filter followed by downsampling (Figure 2-5[b]). On the other hand, decreasing the scale of a signal is done through upsampling (inserting a zero between every sample). This however does not alter the resolution because it can be undone. This is called *interpolation* which is essentially the convolution of a signal being upsampled first with the impulse response of a lowpass filter (Figure 2-5[c]). These notions are fundamental to the implementation of a wavelet transform.

As a convention related to decimation and interpolation shown in Figure 2-5, the downsampling and upsampling is performed on the order of two. This simply doubles or halves the scale of the signal by a factor of two.

Figure 2-5: Decimation and Interpolation.

2.3.5 The 1-D Discrete Wavelet Transform

The practical implementation of a Discrete Wavelet Transform (DWT) is actually much simpler than the mathematical theory which supports it. Daubechies [32] discovered that a wavelet transform can be implemented with a specially designed pair of Finite Impulse Response (FIR) filters cascaded together in a subband coding arrangement as shown in Figure 2-6.

These specially FIR filters are called a *Quadrature Mirror Filter* (QMF) pair due to their unique design. A FIR filter is simply a method of filtering a discretized signal with a known set of discretized samples representing the signal [42], [43]. A FIR filter performs the dot product (sum of products) or convolution between the filter coefficients and the discrete samples. QMF filters have frequency responses such that they separate

the high-frequency and low-frequency components of the input signal. The dividing point of the frequency response of the QMF filter pair is usually *halfway* between 0Hz (DC) and $\pi/2$ (Nyquist frequency). The Nyquist frequency corresponds to half the data sampling rate.

Figure 2-6: Block Diagram of the Analysis Stage ($M = 3$) of a DWT.

The particular wavelet function used depends on the frequency response of the filters. A DWT is normally performed in two stages, namely the *analysis* stage and the *synthesis* stage. Both stages are composed of filter banks designed to operate in decimation or interpolation fashions, respectively. These two stages are usually implemented based on what is called an *octave band subband tree structure* [22]. The aim is to split only the lower half of the spectrum into two equal subbands at a given resolution. Thus the detail or higher frequency subbands of the signal are never decomposed further at any resolution. Figure 2-7 illustrates such structure with a signal decomposition of $M = 3$ iterations.

Figure 2-7: Octave Band Tree Structure and Its Frequency Band Splits.

As described earlier, this multiresolution (coarse-to-fine) signal decomposition design was first proposed in 2-D by Burt and Adelson [33] in 1983 for vision and image coding problems. First the low **L** and high **H** signal bands are obtained by way of filtering. The low band provides the coarser version of the original signal, and the high band contains the detail information. The process is repeated M times on only the lower spectral half component of the higher-level node (through decimation) in the tree. This scheme is the heart of what is called *subband coding*, a very popular image compression technique. As it shall be seen, it is easy to implement a wavelet transform using this concept.

2.3.5.1 The Analysis Stage

Figure 2-8 shows a block diagram of the *analysis* stage of a one-dimensional Discrete Wavelet Transform implemented with QMF FIR filters. Although Figure 2-8 is similar to

Figure 2-6, it also shows the relationship with Figure 2-7, that is the outputs **LLL**, **LLH**, **LH**, and **H**. One important point to note is that the QMF FIR filter pairs, to be considered as *regular* filters, must meet certain basic conditions as described by [22], [26], [30], [32], [34] and [35].

Figure 2-8: Block Diagram of the Analysis Stage ($M = 3$) of a DWT.

This particular implementation (Figure 2-8) which denotes the *analysis* stage is also called the *decomposition* stage of a subband coding design scheme in certain literatures. Here, the analysis implementation illustrates a $M = 3$ iteration implementation. As discussed earlier, the analysis filter bank is constituted of decimation filter arrangements as shown in Figure 2-5[b]. In Figure 2-8, the output of each QMF filter pair is downsampled by two. The output of the QMF filter pair produces the *wavelet coefficients* and are fed into another identical QMF filter pair. This process can be repeated recursively, which results in the spectrum of an original signal being divided into various spectral *subbands* with successively coarser measurements in time. With

each iteration, the width of each spectral subband narrows by two and decreases by two in frequency. In all, applying subdivision on the resulting lower subband repeatedly, one can realize the wavelet transform with $M + 1$ subbands by M stages.

As mentioned above, the same wavelet coefficients are used in both at the input of the lowpass and highpass filters (these filters are actually bandpass filters). The lowpass filter wavelet coefficients represent the coarser coefficients or coarser components $c(M,k)$ of the original signal in a given iteration of resolution 2^M . The highpass filter wavelet coefficients represent the detail coefficients or detail components $d(M,k)$ of the original signal in a given iteration of resolution 2^M . Their mathematical forms are written as follows.

$$c_{1,k} = \sum_n c_{0,n} g(n - 2k) \quad 2-3$$

$$d_{1,k} = \sum_n c_{0,n} h(n - 2k) \quad 2-4$$

The M parameter represents the multiresolution iteration, and is accomplished by the decimation process described earlier. The k parameter controls the translation of the wavelet function in time. The sequences $c_1 = (c_{1,k})$, $d_1 = (d_{1,k})$ are uniquely determined from c_0 by the application of the lowpass and highpass filters. From the next iteration, one gets the next set of sequences $c_2 = (c_{2,k})$, $d_2 = (d_{2,k})$,

$$c_{2,k} = \sum_n c_{1,n} g(n - 2k) \quad 2-5$$

$$d_{2,k} = \sum_n c_{1,n} h(n - 2k) \quad 2-6$$

using the same filters. This process is repeated recursively from one iteration to the other. It must be understood that in each iteration, the current highpass portion coefficients $d(M,k)$ correspond to the difference between the previous lowpass portion coefficients $c(M,k)$ and the current one, that is a passband. Each of these approximated sequences can be considered as a sub-signal of the original one.

In this thesis, the filters g and h used are 6-tap Daubechies filters. Their corresponding impulse responses $g(n)$ and $h(n)$ are given in Table 2-2.

n	$g(n)$	$h(n)$
0	0.23523360	0.02490875
1	0.57055846	0.06041610
2	0.32518250	-0.09546721
3	-0.09546721	-0.32518250
4	-0.06041610	0.57055846
5	0.02490875	-0.23523360

Table 2-2: 6-Tap Daubechies Wavelet Filter Coefficients.

and their filter impulse responses are plotted in Figure 2-9.

Figure 2-9: Filter Responses for 6-Tap Daubechies QMF.

The transfer functions $G(\omega)$ and $H(\omega)$ are plotted in Figures 2-10 and 2-11.

Figure 2-10 denotes the transfer functions $G(\omega)$ and $H(\omega)$ in terms of amplitude (dB)

versus frequency, and Figure 2-11 shows the transfer functions $G(\omega)$ and $H(\omega)$ in terms of log-amplitude versus the frequency linearly.

Figure 2-10: Transfer Functions for 6-Tap Daubechies QMF.

Figure 2-11: Transfer Functions for 6-Tap Daubechies QMF.

2.3.5.2 The Synthesis Stage

The synthesis stage is the reverse implementation of the DWT. All incorporated filter pairs are preceded by an upsampling of two, which constitutes the interpolation scheme introduced in Figure 2-5[c]. The coarse and detail wavelet coefficients generated in the

analysis stage are combined, at each resolution, to eventually reconstruct the original signal. Figure 2-12 illustrates a block diagram of the synthesis stages of a DWT.

The whole process works down the various branches of the tree combining both types of coefficients into approximation signals with higher levels of detail.

Figure 2-12: Block Diagram of the synthesis Stages ($M = 3$) of a DWT.

All coefficients are cascaded through the Daubechies lowpass and highpass filters. In between each filter stage, the coefficients are summed up to produce wavelet coefficients with higher definitions for the next higher level of resolution.

The final set of coefficients $c(M,k)$ is the reconstruction of the original signal data points. One important point to understand is that due to downsampling and upsampling performed in both the analysis and synthesis stages, scaling is automatically aligned so to match the original input signal. The discrete mathematical representation of the reconstruction data points $c(M,k)$ is defined as,

$$c_{0n} = \sum_{k=-\infty}^{\infty} [c_{M,k} g(n - 2k) + d_{M,k} h(n - 2k)] \quad 2-7$$

that is, as the weighted sum of impulse responses, where the weights are the inner products of the signal with the impulse responses g and h . Within certain conditions, the synthesis filters and the analysis filters are designed to be identical.

2.3.6 The 2-D Discrete Wavelet Transform

Figure 2-13 represents a 3-stage ($M = 3$) subband implementation of a two-dimensional discrete wavelet transform for image analysis.

Figure 2-13: Block Diagram of the Analysis Stage ($M = 3$) of a 2-D DWT.

In each stage, the process for decomposition is composed of two steps. First, one splits the horizontal elements of the input image into two subbands of equal frequency width and keeps one vertical element out of two. Second, one splits the vertical components of the input image into two subbands of equal frequency width and keeps

one horizontal element out of two. As a result, the input image V_0 is divided into 4 subimages, namely V_1^{LL} , V_1^{LH} , V_1^{HL} and V_1^{HH} .

By repeating the same decomposition process on the resulting lower subimage V_1 , one can realize a wavelet decomposition with $3M + 1$ subbands. The spatial orientations of the resulting subbands are shown in Figure 2-14.

Figure 2-14: Spatial Orientation of the Wavelet Coefficients in 2-D.

The original dimensions of the test images used in this work are 512 x 512 pixels. At a given resolution, each detail image corresponds to a particular spatial orientation tuning, that is, the vertical, horizontal and diagonal directions. The subimage **LL3**, shown at the top left corner, has the lowest frequency content. The **LL3** subimage is usually referred as the DC. The subimage **HH1**, shown at the bottom right, has the

highest diagonal frequency content. The subimage **HL2** and **LH2** have the highest horizontal and vertical frequency content, respectively. The number of samples in each subimage at the $(M + 1)$ stage is $1/4$ of the M th stage.

CHAPTER 3

VPIC

3.1 General

As stated in section 2.2.4, Visual Pattern Image Coding (VPIC) was introduced by Bovik and Chen in 1990 [4]. VPIC is an image data compression technique based on pattern matching, and is somewhat a cross-breed between Vector Quantization (VQ) and Block Truncation Coding (BTC). The main difference between VPIC and other pattern matching compression techniques is in the implementation of its relatively small codebook requirement and the code vectors which represent visually meaningful subimages or image blocks, which match biologically-derived visual response characteristics of the human visual system.

3.2 Properties of Human Vision

Experimental studies in animals have shown that there are specific nerves that *fire* only when excited by a particular color, a particular form, a vertical line, an angled line, a circular blob, or upon perception of motion in a specific direction [36]. The discovery of these specialized groups of nerves led naturally to the concept of *pattern* or *feature* detection [37]. The fact that some nerves *fire* only when presented with specific features leads to feature-detection models, in which a scene is made up of individual features.

The visual system consists of various receptor fields, including the retina, lateral geniculate body, and visual cortex. Within the visual cortex are different types of cells that respond to structures of different sizes, contrasts, and orientations. Some large size cells respond (tune) to large size objects of low spatial frequency. Other small size cells respond to smaller size objects of higher spatial frequency. Some cells respond to high contrast and other to low contrast.

Cells responding to low spatial frequency stimuli provide visual information about the gross features of an object, and those responding to high spatial frequency features provide information on the fine details of an object [37]. As a result, many researchers believe that the HVS functions as a multiscale model where discrete *channels* in the system process the visual information within restricted ranges of spatial frequencies. These information channels represent the activity performed independently by those so called *size-selective* cell's discussed above [37].

In the study of the functional architecture of biological visual processing, certain *low-level* psychophysical and physiological properties of vision can be regarded as being accurately modeled. For example, the concepts of image *smoothness* (low-frequency or uniform regions) and *detail* (high-frequency or edge, bars or visual features) are recast as localized *visual continuity* and *visual discontinuity* constraints (features), respectively [4], [44]. And therefore, this is where *pattern matching* image coding techniques come into play.

3.3 VPIC Pattern Design

In this version of VPIC, the input image is segmented into image blocks having dimensions of 4 x 4 pixels. From a discrete point of view, a 4 x 4 image block $b_{i,j}$ has the following form:

$$b_{i,j} = \begin{bmatrix} I_{i,j} & I_{i,j+1} & I_{i,j+2} & I_{i,j+3} \\ I_{i+1,j} & I_{i+1,j+1} & I_{i+1,j+2} & I_{i+1,j+3} \\ I_{i+2,j} & I_{i+2,j+1} & I_{i+2,j+2} & I_{i+2,j+3} \\ I_{i+3,j} & I_{i+3,j+1} & I_{i+3,j+2} & I_{i+3,j+3} \end{bmatrix}$$

where $I_{i,j}$ denotes an individual pixel intensity in an image block $b_{i,j}$. An image block $b_{i,j}$ is limited to,

$$b_{i,j} = [I_{n,m} : 4i \leq n \leq 4i + 3, 4j \leq m \leq 4j + 3] \quad , 0 \leq i, j \leq 4 \quad 3-1$$

Based on the concepts of visual continuity and discontinuity, two types of blocks are defined, namely *uniform* and *edge* blocks. Uniform blocks are subimages whose internal distribution (pixel values) is consistent, hence contain very little information. In that case, it is said that any variation within the block falls below the Weber threshold¹,

¹ Weber's Law [2]: If the luminance of f_0 of an object is just noticeably different from the luminance f_s of its surround, then their ratio is

$$\left| \frac{f_s - f_0}{f_0} \right| = \text{constant}$$

and the pixel values are presumed constant. As a result, uniform image blocks contain strictly low frequency information. Edge blocks are subimages whose internal distribution has a high degree of intensity variation, containing high frequency information which is visually significant. In VPIC, significant importance is given to edge patterns since they are considered to be the most important attributes of an image as far as the HVS is concerned.

3.3.1 Edge Pattern Design

In VPIC, the size of a pattern (image block) is finite, and the frequency content of the edge patterns is constrained to allow for occurrences of singles edges only. As a result, codebook generation is much less complex and the codebook are smaller than the ones used in standard VQ schemes. Edge blocks or edge patterns are detected based on the natural measure of variation reflected by the notion of gradient as discussed in Chapter 2.

The discretized local gradient magnitude $\nabla b_{i,j}$ which corresponds to the contrast, and the discretized local gradient orientation $\theta b_{i,j}$ which corresponds to the orientation of the intensity change within each image block are given respectively by,

$$|\nabla b|_{i,j} = \sqrt{(\nabla_x b_{i,j})^2 + (\nabla_y b_{i,j})^2} \quad 3-2$$

$$\theta b_{i,j} = \tan^{-1} \left(\frac{\nabla_y b_{i,j}}{\nabla_x b_{i,j}} \right) \quad 3-3$$

The discrete gradient magnitude ∇ of a given 4×4 block is computed as the oriented (horizontally and vertically) average finite-difference of neighbouring pixel values within an image block. The partial derivative approximations in the x and y directions are computed as follows:

$$\nabla_x b_{i,j} = \frac{\partial I}{\partial x} = \text{avg}[I_{n,m}: 4i+2 \leq n \leq 4i+3, 4j \leq m \leq 4j+3] - \text{avg}[I_{n,m}: 4i \leq n \leq 4i+1, 4j \leq m \leq 4j+3] \quad 3-4$$

$$\begin{aligned} \nabla_x b_{i,j} &= \frac{1}{8} [I_{i+2,j} + I_{i+2,j+1} + I_{i+2,j+2} + I_{i+2,j+3} + I_{i+3,j} + I_{i+3,j+1} + I_{i+3,j+2} + I_{i+3,j+3}] - \\ &\quad \frac{1}{8} [I_{i,j} + I_{i,j+1} + I_{i,j+2} + I_{i,j+3} + I_{i+1,j} + I_{i+1,j+1} + I_{i+1,j+2} + I_{i+1,j+3}] \\ &= \text{avg of lower half} - \text{avg of upper half} \end{aligned} \quad 3-5$$

$$\nabla_y b_{i,j} = \frac{\partial I}{\partial y} = \text{avg}[I_{n,m}: 4i \leq n \leq 4i+3, 4j+2 \leq m \leq 4j+3] - \text{avg}[I_{n,m}: 4i \leq n \leq 4i+3, 4j \leq m \leq 4j+1] \quad 3-6$$

$$\begin{aligned} \nabla_y b_{i,j} &= \frac{1}{8} [I_{i,j+2} + I_{i,j+3} + I_{i+1,j+2} + I_{i+1,j+3} + I_{i+2,j+2} + I_{i+2,j+3} + I_{i+3,j+2} + I_{i+3,j+3}] - \\ &\quad \frac{1}{8} [I_{i,j} + I_{i,j+1} + I_{i+1,j} + I_{i+1,j+1} + I_{i+2,j} + I_{i+2,j+1} + I_{i+3,j} + I_{i+3,j+1}] \\ &= \text{avg of right half} - \text{avg of left half} \end{aligned} \quad 3-7$$

where $\nabla_x b_{i,j}$ represents the partial derivative in the x direction, and $\nabla_y b_{i,j}$ represents the partial derivative in the y direction, at location i, j .

The code vectors in the VPIC codebook consist of four basic edge patterns with respective orientations $0^\circ, 45^\circ, 90^\circ, 135^\circ$. The remaining complementary 45° increments, namely $180^\circ, 225^\circ, 270^\circ, 315^\circ$ are separately identified by transmitting the *oriented polarity* of the edge pattern. Figure 3-1 shows the 14 basic VPIC code vectors or edge patterns p_j used in this thesis which were defined over the above orientations for 4×4 image blocks.

As discussed in [4], it is not a requirement that all patterns presented in Figure 3-1 be used to code images; a subset or another set with slightly different orientations can be sufficient, however, the quality of the reconstructed image may suffer somewhat.

Figure 3-1: VPIC 14-Visual Pattern Code Vectors.

In any of the vectors shown in Figure 3-1, a positive element indicates that the corresponding pixel has a positive value, and a negative element indicates a negative pixel value. Elements of the same polarity have the same value, therefore, there can only be two different values per pattern; one representing the positive polarity, and the other, the negative polarity.

3.3.2 VPIC Codebook Generation

The code vectors used in the VPIC codebook reflect the patterns shown in Figure 3-1. As discussed in the preceding paragraph, two values a and b of the code vectors called the *oriented gradient coefficients* are computed such that each pattern satisfies the following two conditions:

- a. zero mean; and
- b. unit gradient magnitude ($|\nabla p_j| = 1$).

For example, the oriented gradient coefficients a and b of the 4×4 VPIC pattern p_3 are calculated as follows. The bit map of p_3 is illustrated below:

$$p_3 = 0x0013_{16} = \begin{bmatrix} - & - & - & - \\ - & - & - & - \\ - & - & - & + \\ - & - & + & + \end{bmatrix}$$

Let a represent the oriented gradient coefficient of a positive element in the bit map denoted by p_3 , and b the oriented gradient coefficient of a negative element. Then the partial derivatives in the x and y directions of pattern p_3 , respectively $\nabla_x p_3$ and $\nabla_y p_3$ are calculated as:

$$\begin{aligned} \nabla_x p_3 &= \frac{\text{bottom 8 pixels}}{8} - \frac{\text{top 8 pixels}}{8} \\ &= \frac{1}{8} (3a + 5b - 8b) = \frac{3}{8} (a - b) \end{aligned}$$

$$\begin{aligned}
 \nabla_y p_3 &= \frac{\text{right 8 pixels}}{8} - \frac{\text{left 8 pixels}}{8} \\
 &= \frac{1}{8} (3a + 5b - 8b) = \frac{3}{8} (a - b)
 \end{aligned}
 \tag{3-9}$$

The gradient magnitude of pattern p_3 is:

$$\begin{aligned}
 |\nabla p|_3 &= \sqrt{\nabla_x p_3^2 + \nabla_y p_3^2} \\
 |\nabla p|_3 &= \sqrt{2 \left[\frac{3}{8} (a - b) \right]^2} \\
 &= \frac{3\sqrt{2}}{8} (a - b)
 \end{aligned}
 \tag{3-10}$$

The normalized value of a for pattern p_3 is:

$$|\nabla p|_3 = 1 = \frac{3\sqrt{2}}{8} (a - b)
 \tag{3-11}$$

thus,

$$a = \frac{8}{3\sqrt{2}} + b
 \tag{3-12}$$

To find the normalized value of parameter b , one must use the 0-mean of pattern p_3 .

$$3a + 13b = 0
 \tag{3-13}$$

thus, substituting equ. 3-12 into equ. 3-13 gives,

$$\begin{aligned}
3\left(\frac{8}{3\sqrt{2}} + b\right) + 13b &= 0 \\
\frac{8}{\sqrt{2}} + 16b &= 0 \\
b &= -\frac{\sqrt{2}}{4} = -0.3535534
\end{aligned}$$

and, substituting the value b into equ. 3-12 gives,

$$\begin{aligned}
a &= \frac{8}{3\sqrt{2}} + b = \frac{8}{3\sqrt{2}} - \frac{\sqrt{2}}{4} \\
a &= \frac{13\sqrt{2}}{12} = 1.5320647
\end{aligned}$$

The next step is to initialize a vector v_3 with the values a and b . This vector v_3 is the code vector in the VPIC codebook that represents pattern p_3 . For example, the oriented gradient coefficients $a = 1.5320647$ and $b = -0.3535534$ which are related to p_3 are used to initialize the vector v_3 based on p_3 's bit map.

$$p_3 = \begin{bmatrix} - & - & - & - \\ - & - & - & - \\ - & - & - & + \\ - & - & + & + \end{bmatrix} \Rightarrow v_3 = \begin{bmatrix} -0.3535534 & -0.3535534 & -0.3535534 & -0.3535534 \\ -0.3535534 & -0.3535534 & -0.3535534 & -0.3535534 \\ -0.3535534 & -0.3535534 & -0.3535534 & 1.5320647 \\ -0.3535534 & -0.3535534 & 1.5320647 & 1.5320647 \end{bmatrix}$$

It can be seen that while v_3 is initialized to the values of a and b , the vector v_3 preserves the orientation of the pattern p_3 . This mathematical process must be done for every VPIC pattern defined in Figure 3-1.

A view of the VPIC codebook that represents all the pre-defined patterns listed in Figure 3-1 is shown in Figure 3-2. In reality, each oriented gradient coefficient is stored

coding schemes [4]. This problem can be reduced to some degree by applying a smoothing filter (lowpass filter) to the mean intensity subimage prior to adding the edge blocks. In turn this process minimizes the degrading of details (edges) in the image, and the overall visual distortion is thereby decreased. In this thesis, a spatial 1-D low-pass 3-tap FIR filtering process was applied along each dimension where each pixel in the mean intensity image $u_{i,j}$ is replaced by a weighted average $w[n]$ (filter weights) of its neighborhood pixels, that is,

$$x_{i,j} = \sum_{n=-1}^1 w_n \cdot u_{i,j+n} \quad , i,j = 1..3 \quad 3-17$$

was performed along the x-direction, and

$$y_{i,j} = \sum_{n=-1}^1 w_n \cdot u_{j,i+n} \quad , i,j = 1..3 \quad 3-18$$

was performed along the y-direction. In both cases, the filter weights $w[n]$ were defined as,

$$w[n] = \left[\frac{1}{4}, \frac{1}{2}, \frac{1}{4} \right]$$

Other weighting coefficients can be used, but this option was not pursued. The above low-pass FIR filter was used because of its maximally flat frequency response (see Figures 3-6, 3-7).

The *GenCodbk* algorithm which generates the VPIC codebook of 4 x 4 code vectors was implemented in C++ using the Symantec compiler on a Pentium 90Mhz PC system. The Symantec compiler was chosen because of its internal Disk Operating System (DOS) Extender (DOSX) which takes advantage of the extended memory features of the ix86 family of Intel processors. By using DOSX, the 64K byte limit on array sizes and the 640K memory barrier associated with IBM personal computers were overcome. Array size is limited only by memory, and memory up to 4 Giga-bytes was addressable. This feature of the compiler proved essential to the efficient implementation of C++ algorithms for this thesis.

3.3.3 Gradient-Based Thresholding

The use of thresholding based on the notion of gradient allows for the effective differentiation of uniform and edge image blocks. In VPIC, a *minimum gradient threshold* $\nabla_{\min}$ and a *maximum gradient* $\nabla_{\max}$ are used.

The minimum gradient threshold $\nabla_{\min}$ denotes the minimum perceivable gradient magnitude that can occur in an image, in accordance to Weber's law and is responsible for the differentiation between edge image blocks and uniform image blocks. In addition, this threshold is constant throughout the coding process.

The maximum gradient $\nabla_{\max}$ represents the maximum gradient magnitude, and all edge blocks with greater gradient magnitudes are truncated to $\nabla_{\max}$. The use of minimum threshold and maximum gradient inherently limits a certain range of gradient magnitudes which can be quantized much more optimally. The question that comes to mind is will

there be any visual degradation by coding to the maximum gradient? According to Levine [52], there are very few negative side effects in truncating high valued gradients to a maximum gradient $\nabla_{\max}$, and that is for two reasons:

- a. large gradient values occur infrequently in natural images; and
- b. the eye is not very sensitive to the magnitude of edge gradients.

3.4 VPIC Architecture

As stated earlier, the VPIC image coding scheme has been designed somewhat similarly to that of Vector Quantization (VQ) and Block Truncation Coding (BTC) image coding schemes. Respectively, the major differences from these three coding methods are:

- a. unlike VQ, VPIC is concerned with a small set of visually significant patterns (codebook), based on a simple biologically determined geometric model, which is defined *independently* of the images to be decoded. Each pattern is a finite subimage (block) that is visually meaningful to a normal human observer. Similar to VQ, each edge block is *matched* to only one of the codebook vectors, and the index of the selected vector is transmitted to the decoder; and
- b. instead of transmitting the low and high means of every image blocks as in BTC, the gradient and the mean of only the edge blocks are transmitted. As for the uniformly distributed image blocks, only their local means are transmitted.

As a result, the architecture of VPIC is very simple, and as such, the coding is computationally fast.

3.4.1 VPIC Coder

Figure 3-4 depicts a detailed block diagram of the VPIC architecture. The first step that takes place in the VPIC coder is the segmentation of the image into $n \times n$ image blocks. The mean intensity and the residual of each block are then computed and saved in memory buffers u and r , respectively. The residual r is computed as:

$$r = b - u \quad 3-14$$

where u represents the mean of a $n \times n$ image block, and b a $n \times n$ image block [46].

The local gradient and orientation are then computed using the discrete approximation equations 3-2 and 3-3. The next step is the detection of edges or patterns. This is done as so. If,

$$\nabla b_{i,j} \leq \nabla_{\min} \quad 3-15$$

then the image block is determined to be an uniform block, otherwise, it is an edge block.

In both cases, uniform or edge block, the local mean $u_{i,j}$ is quantized and transmitted to the decoder.

3.4.1.1 Uniform Block Coding

In the case where an uniform block type is detected, only its mean $u_{i,j}$ is quantized and transmitted as u^* . An uniform block does not carry any visually important information other than background information, thus pattern matching is not performed and the local gradient is not needed at the decoder. Consequently, only the quantized mean of an uniform block is encoded as $n_{uu} + 1$ bits. The additional flag bit is needed to indicate to the decoder the *type* of image block. When *blocktype* = 0, $b_{i,j}$ is of type uniform, and when *blocktype* = 1, $b_{i,j}$ is of type edge.

Additionally, the mean of an uniform block requires a greater precision than the mean of an edge block [4], and therefore, the quantized mean of an uniform block is transmitted with greater precision. This is achieved by encoding the local mean $u_{i,j}$ with more bits (i.e. 8-bits per mean value). In here, we are only talking about the mean of an image block. As we will see later, an edge block overall requires greater coding precision than an uniform block.

3.4.1.2 Edge Block Coding

Edge blocks require more attention in the encoding process than uniform blocks. In the case where an edge block is detected, three things must take place:

- a. the local mean is quantized and transmitted;
- b. the local gradient is quantized and transmitted; and
- c. pattern matching is performed, and the index of the matched pattern transmitted.

The local mean $u_{i,j}$ of an edge block is quantized to $n_{ue} + 1$ bits (with less precision than n_{uu}) and transmitted as u^* . As mentioned earlier, the additional flag bit identifies the *blocktype* of the image block to the decoder. Additionally, the local gradient is also quantized, compressed to n_g bits, and transmitted as g . This last value is transmitted for edge image blocks only. According to Chen and Bovik, the local gradient is far less critical than the number of oriented patterns used in the matching process. In fact, they claim that very good results are obtained without transmitting the gradient at all! Therefore, the attribute n_g can be reduced considerably to attain further compression ratios.

3.4.1.3 VPIC Pattern Matching

Pattern matching is performed only on edge blocks for reasons mentioned in section 3.4.1. For faster results, this process is performed in two steps in VPIC. Step one consists in using the pre-computed orientation of the edge block to map it to a subset of VPIC codebook vectors shown in Figure 3-1. In this case, each group contains four or fewer pre-defined image patterns.

The actual pattern matching is done in the second step. It is based on binary correlation, in which an edge block is matched to the codebook vectors in the pre-selected VPIC subset whose *polarity distribution* is the closest in the Hamming sense. The polarity distribution is the distribution of (+) and (-) elements in the bit map corresponding to the image block. The index of the selected codebook vector v_j is encoded in j as $n_p + 1$ bits where the additional flag bit indicates to the decoder the *oriented polarity* of the pattern as described in section 3.3.1. For example, a bit "0" tells

the decoder that the pattern to be decoded is located in the first half of the codebook, namely vectors between 0° and 135° . A bit “1” indicates that the pattern is located in the second half, namely vectors between 180° and 315° .

3.4.2 VPIC Decoder

One of the greatest advantages offered by VPIC is the decoding efficiency since the coded image blocks are represented either as uniform or edge blocks. At the decoder, an image block is reconstructed based on the quantized values of the transmitted local means in the case of uniform blocks, and on a visual pattern index and quantized values of local gradients and means in the case of edge blocks.

The reconstruction of uniform blocks simply requires the substitution of the local means transmitted by the coder. Each pixel value in a $n \times n$ reconstructed uniform block is assigned with the value of the mean transmitted by the coder.

The reconstruction of edge blocks is a bit more complex, in that for each pixel of a $n \times n$ reconstructed edge block a table look-up is required, as well as a single multiplication and addition. The codebook contains $n \times n$ vectors describing both the edge pattern orientations and edge gradient magnitudes called *oriented gradient coefficients* as described in section 3.3.2. These values are used in the reconstruction process for adequate representation of detected edge blocks. The normalized vectors in the VPIC codebook for the 0° subset is shown in Figure 3-5.

An edge block is reconstructed by adding the mean transmitted by the coder to the product of the gradient and the pre-computed *oriented gradient coefficients*. The vector v_j , consisting of *oriented gradient coefficients* reflects the orientation of the pattern, a zero

coding schemes [4]. This problem can be reduced to some degree by applying a smoothing filter (lowpass filter) to the mean intensity subimage prior to adding the edge blocks. In turn this process minimizes the degrading of details (edges) in the image, and the overall visual distortion is thereby decreased. In this thesis, a spatial 1-D low-pass 3-tap FIR filtering process was applied along each dimension where each pixel in the mean intensity image $u_{i,j}$ is replaced by a weighted average $w[n]$ (filter weights) of its neighborhood pixels, that is,

$$x_{i,j} = \sum_{n=-1}^1 w_n \cdot u_{i,j+n} \quad , \quad i, j = 1..3 \quad 3-17$$

was performed along the x-direction, and

$$y_{i,j} = \sum_{n=-1}^1 W_n \cdot u_{j,i+n} \quad , \quad i, j = 1..3 \quad 3-18$$

was performed along the y-direction. In both cases, the filter weights $w[n]$ were defined as,

$$w[n] = \left[\frac{1}{4}, \frac{1}{2}, \frac{1}{4} \right]$$

Other weighting coefficients can be used, but this option was not pursued. The above low-pass FIR filter was used because of its maximally flat frequency response (see Figures 3-6, 3-7).

Figure 3-6: Frequency Response of $w[n]$.

Figure 3-7: Frequency Response of $w[n]$ in Terms of $20 \log |W(e^{j\omega})|$.

3.4.3 Computing Compression Ratios in VPIC

To code a 4×4 uniform block, $n_u = n_{uu} + 1$ bits per block (bpb) are required. To code a 4×4 edge block, $n_e = n_{ue} + n_g + n_p + 2$ bits per block (bpb) are required. Consequently, n_e is always greater or equal to n_u ($n_e \geq n_u$). In an image coded with N_u uniform and N_e edge blocks, the exact compression ratio can be computed as per equ. 3-19 and 3-20.

$$C_{VPIC} = \frac{128 (N_u + N_e)}{(N_u n_u + N_e n_e)} \quad 3-19$$

or as,

$$C_{VPIC} = \frac{\text{image total no. of bytes} \cdot 8}{(N_u n_u + N_e n_e)} \quad 3-20$$

3.4.4 Simulations and Results

A VPIC algorithm has been developed, as part of this thesis, and implemented in the C++ programming environment using the Symantec C++ compiler on an IBM Pentium. The VPIC coder/decoder algorithm consists of roughly 4,000 lines of code which is totally modularized and reasonably commented.

To verify the accuracy of the algorithm against the published version, several trials were conducted on three different images, namely *Lena*, *Mandrill*, and *Clown*. All images are 512 x 512 in size. A performance analysis of the VPIC algorithm was performed with several input parameters as listed in Table 3-1.

VPIC (Spatial Domain) - Simulations									
Compression Parameters	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	Trial 9
Uniform mean bpb (n_{uu})	5	3	8	5	8	8	8	8	8
Block type (flag bit)	1	1	1	1	1	1	1	1	1
Edge mean bpb (n_{ue})	3	5	8	3	8	8	8	8	8
Gradient bpb (n_e)	0	0	0	8	8	8	8	8	8
Pattern index bpb (n_p)	4	4	4	4	4	4	4	4	4
Pattern polarity (flag bit)	1	1	1	1	1	1	1	1	1
V_{min}	30	30	20	30	30	30	20	10	20
V_{max}	255	255	90	255	90	255	90	90	90
Postfiltering	no	no	no	no	no	no	no	no	yes
Total: (n_u, n_e)	(6, 9)	(4, 11)	(9, 14)	(6, 17)	(9, 22)	(9, 22)	(9, 22)	(9, 22)	(9, 22)

Table 3-1: Input Parameters Used in VPIC Simulations

For each test image, Table 3-2 shows the number of uniform blocks N_u and edge blocks N_e detected, and the resulting compression ratios (C) and Peak Signal-to-Noise Ratios (PSNR) attained by VPIC. In the following paragraphs, discussion is concentrated on *Lena* only since the results pertaining to the other two types of test images are very similar.

VPIC (Spatial Domain) - Compression Results				
	Output Parameters	Lena	Mandrill	Clown
Trial 1	Number of uniform blocks (N_u)	14,790	12,306	14,154
	Number of edge blocks (N_e)	1,594	4,078	2,230
	Compression (C)	20:1	18.30:1	19.56:1
	Signal-to-noise-ratio (PSNR)	27.13 dB	21.64 dB	26.59 dB
Trial 2	Number of uniform blocks (N_u)	14,790	12,306	14,154
	Number of edge blocks (N_e)	1,594	4,078	2,230
	Compression (C)	26.79:1	21.36:1	25.15:1
	Signal-to-noise-ratio (PSNR)	25.25 dB	21.29 dB	25.05 dB
Trial 3	Number of uniform blocks (N_u)	14,790	12,306	14,154
	Number of edge blocks (N_e)	1,594	4,078	2,230
	Compression (C)	13.78:1	13.13:1	13.60:1
	Signal-to-noise-ratio (PSNR)	27.62 dB	21.91 dB	27.14 dB
Trial 4	Number of uniform blocks (N_u)	14,790	12,306	14,154
	Number of edge blocks (N_e)	1,594	4,078	2,230
	Compression (C)	17.86:1	14.24:1	16.77:1
	Signal-to-noise-ratio (PSNR)	27.47 dB	21.55 dB	26.93 dB
Trial 5	Number of uniform blocks (N_u)	14,790	12,306	14,154
	Number of edge blocks (N_e)	1,594	4,078	2,230
	Compression (C)	12.35:1	10.25:1	11.74:1
	Signal-to-noise-ratio (PSNR)	27.99 dB	21.80 dB	27.49 dB
Trial 6	Number of uniform blocks (N_u)	14,790	12,306	14,154
	Number of edge blocks (N_e)	1,594	4,078	2,230
	Compression (C)	12.35:1	10.25:1	11.74:1
	Signal-to-noise-ratio (PSNR)	27.96 dB	21.79 dB	27.48 dB
Trial 7	Number of uniform blocks (N_u)	13,761	9,708	12,752
	Number of edge blocks (N_e)	2,623	6,676	3,632
	Compression (C)	11.39:1	8.70:1	10.58:1
	Signal-to-noise-ratio (PSNR)	28.28 dB	21.80 dB	27.93 dB
Trial 8	Number of uniform blocks (N_u)	11,183	5,492	10,141
	Number of edge blocks (N_e)	5,201	10,892	6,243
	Compression (C)	9.52:1	6.99:1	8.93:1
	Signal-to-noise-ratio (PSNR)	28.02 dB	21.00 dB	26.75 dB
Trial 9	Number of uniform blocks (N_u)	13,761	9,708	12,752
	Number of edge blocks (N_e)	2,623	6,676	3,632
	Compression (C)	11.39:1	8.70:1	10.58:1
	Signal-to-noise-ratio (PSNR)	27.98 dB	21.71 dB	27.61 dB

Table 3-2: Simulations Results for VPIC Coding Scheme Applied in the Spatial Domain.

Comparing the PSNRs between trials 1 and 2 demonstrates that the parameter n_{uu} requires a greater coding precision than that of the parameter n_{ue} . While the compression ratio of trial 2 is higher than that of trial 1, as expected the trial 2 PSNR is lower than that of trial 1. In trial 3, although the means of the uniform and edge blocks are fully coded, the PSNR obtained is slightly better than in trial 1 and the compression ratio is much lower. This only proves that the mean of uniform block requires a greater coding precision than that of edge block.

Examining n_g from trials 1 and 4, the gradient information is not transmitted in one case (trial 1), and fully coded in the other case (trial 4). The fact that the PSNR has little increase when the gradient of edge blocks are fully coded, (no degradation due to $n_g = 8$) shows that transmission of gradient information is not a priority.

Results from trials 5 and 6 show that truncating local gradients (see section 3.3.3) that are above the maximum gradient magnitude to $\nabla_{\max}$ seems to have little negative effect on the PSNR and no effect at all on the compression ratio. This stable performance is attributed to the fact that limiting the range of local gradients on edges allows for the bulk of the quantized gradient values to be coded with much more precision.

Two parameters are modified between trials 7 and 8. First, $\nabla_{\min}$ is set to 20 in trial 7, and second $\nabla_{\min}$ is set to 10 in trial 8. Because $\nabla_{\min}$ in trial 7 is decreased down to 20 (compared to previous settings of 30), there are 2,623 edge blocks detected instead of 1,594 in the previous trials, and there are even more edge blocks detected in trial 8 ($N_e = 5,201$ edge blocks). Given that there are more edge blocks detected, one would expect large increase in PSNRs. The reality is that there is not a large increase. The increase is only minimal, from 27.99 dB in trial 5, to 28.28 dB in trial 7, and to 28.02 dB in trial 8.

According to Chan and Chan [38], adding more edge blocks do not have contributive result due to the limited spatial support of the visual patterns proposed by Chen and Bovik [4]. In addition, Chan and Chan speculate that much of the *textured pixels* are smoothed out by 2-level quantization, which renders the reconstructed image unnatural.

Consequently, one might conclude that there comes a point where detection of additional edge blocks does not contribute very much to a better quality of the reconstructed image. In fact, the compression ratio decreases as a result of detecting/coding larger numbers of edge blocks because more bits are used in the coding process. Figure 3-8 and 3-9 illustrate the original and trial 7 reconstructed 512 x 512 test image of *Lena*.

Figure 3-8: Original 512 x 512 Test Image *Lena*.

Figure 3-9: Reconstructed 512 x 512 Test Image *Lena* from Trial 7 (Spatial VPIC, PSNR = 28.28 dB).

Figure 3-10 and 3-11 illustrate the original and trial 7 reconstructed 512 x 512 test image of the *Mandrill*.

Figure 3-10: **Original 512 x 512 Test Image *Mandrill*.**

Figure 3-11: Reconstructed Test Image *Mandrill* for Trial 7 (Spatial VPIC, PSNR $\approx$ 21.80 dB).

Figure 3-12 and 3-13 illustrate the original and trial 6 reconstructed 512 x 512 test image of the *Clown*.

Figure 3-12: Original 512 x 512 Test Image *Clown*.

Figure 3-13: Reconstructed Test Image *Clown* for Trial 7 (Spatial VPIC, PSNR = 27.93 dB).

As we can see from the appearance of the reconstructed images in Figures 3-9, 3-11 and 3-13 that the blockiness aspect is still quite visual. And yet these results are at VPIC's highest performance set up. A popular technique used to decrease the effect of blockiness is by applying a lowpass filter to the reconstructed image.

In trial 9, a postfiltering process is applied with $\nabla_{\min}$ set back to 20. The settings in trial 9 are otherwise the same as in trial 7. The purpose of filtering is to smooth out the block effect in an reconstructed image by applying a lowpass filter as discussed in section 3.4.2.1. This means that the low-frequency content is preserved while lowering the high-frequency artifacts, since block effects have high-frequency content to a certain degree. Figure 3-14 illustrates the reconstructed image of *Lena* with postfiltering.

Figure 3-14: Reconstructed Test Image *Lena* with Postfiltering (trial 9) (Spatial VPIC, PSNR = 27.98 dB).

Comparing the PSNRs between trial 7 and 9, the PSNR is decreased somewhat due to postfiltering. This phenomenon is in fact discussed by Lindley [39]. The visual result of lowpass filtering is a slight blur of the image, thus lower PSNR. This blur results because any sharp pixel transitions are averaged with their surroundings as the high-frequency content is attenuated. However, the final result is still a blockiness effect as seen in Figure 3-14.

The results obtained in this chapter are somewhat hard to compare due to insufficient tests data available from the Chen and Bovik article. Chen and Bovik did not documented anywhere the PSNR performance by their algorithm, nor did they performed any kind of large-scale results. But the one thing is for sure, although VPIC is a great approach to image compression, it does not have the capability to preserve image fidelity to a very high standard in the spatial domain at least. An article written by K.W Chan and K.L. Chan [38] does in fact confirm the moderate PSNR performance of VPIC.

The VPIC coding scheme in the spatial domain can be summarized as follows:

- a. moderate PSNR performance;
- b. poor compression performance;
- c. simple architecture;
- d. small codebook;
- e. universal codebook;
- f. produces blockiness effects at high compression ratios;
- g. no training phase required when designing visual patterns; and,
- h. expensive codebook search operations are eliminated.

As we will see in Chapter 4, the VPIC coding technique just described will be implemented in the wavelet domain. Although the VPIC scheme has low PSNR performance in the spatial domain, we estimated that it still be worth while to spend further efforts in implementing VPIC in the wavelet domain in the hope to increase its PSNR.

CHAPTER 4

Image Data Compression in the Wavelet Domain

4.1 Compression Advantages

There are several advantages in coding an image which has been transformed initially. For example, by quantizing various statistically calculated values of small blocks of transform coefficients, such as the mean and variance, a signal can be almost fully recovered once the reverse transform is performed [25]. Also, the transform coefficients with large variances can be retained as significant features for pattern coding, and the transform coefficients with small variances, hence low energy content, can be discarded due to their low contributions to the visual quality of the reconstructed image. In Wavelet Subband Coding for instance, each subband can be coded more efficiently than the entire full band image by exploiting different encoding techniques mixed with adaptive block size schemes.

In the following sections, some of the notions used in the coding techniques modified or developed as part of this thesis, and which were used for the purpose of reducing the amount of information transmitted, are discussed in more detail.

4.1.1 High Resolution Subbands not Transmitted

As discussed earlier in this thesis, one way to reduce the amount of information in an

2

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

image once it has been mapped into a given transform domain is to discard low energy transform coefficients. This strategy is viable due to the fact that most transforms have the capability to compact signal energy into fewer transform coefficients [25]. Needless to say, some information is invariably lost, but its visual impact can be minimized by utilizing some adaptive processing schemes.

In the wavelet domain, discarding of low energy transform coefficients tends to be a relatively simple task. The higher order subbands in the wavelet domain have low energy transform coefficients. Therefore, by discarding the transform coefficients in the high order subbands, compression of data can be easily achieved.

Figure 4-1 shows the transform coefficients for each subband of a level $M = 3$ wavelet decomposition of the image *Lena*. A 6-tap Daubechies filter was used for this wavelet decomposition. As discussed in Chapter 2, a level $M = 3$ decomposition produces a total of 10 subbands. The lowest frequency subband, the DC subband, is denoted as subband 0. The highest frequency subband is denoted as subband 9.

The results of discarding various subbands of *Lena* are summarized in Table 4-1.

Trial	Discarded Subbands (only)	C	bpp	PSNR (dB)	% of Wavelet Coefficients Set to Zero
1	9	1.33 : 1	6.02	40.66	25
2	8	1.33 : 1	6.02	37.97	25
3	7	1.33 : 1	6.02	34.79	25
4	6	1.07 : 1	7.5	37.40	6.25
5	5	1.07 : 1	7.5	35.28	6.25
6	4	1.07 : 1	7.5	30.80	6.25
7	3	1.02 : 1	7.8	34.37	1.56
8	2	1.02 : 1	7.8	33.05	1.56
9	1	1.02 : 1	7.8	28.32	1.56
10	7 - 9	4 : 1	2	32.68	75
11	6, 8, 9	2.3 : 1	3.5	34.10	56.3
12	4 - 9	16 : 1	0.5	27.46	94
13	1 - 9	64 : 1	.125	23.89	98

Table 4-1: Results of Discarding of Subbands.

Various subbands are set to zero and then the image is reconstructed from the remaining transform coefficients. The first thing to note is as lower order subbands are discarded one at the time in trials 1 to 9, the PSNR tends to decrease. As well, the compression

Figure 4-1: Transform Coefficients of a Level $M = 3$ Wavelet Decomposition of *Lena*.

ratio C decreases. Therefore, even though there is less data discarded starting from trials 1 to 9, the PSNR still keeps declining. The only explanation left which can describe this

kind of behavior is that lower subbands contains more relevant information, thus more energy per transform coefficient. Hence, if high energy coefficients are set to zero, the resulting reconstructed image does have a visually poor quality. Therefore, lower order subbands are better than higher order subbands at preserving the PSNR. And so, discarding subband 1 for example (see trial 9), the lowest level of PSNR is produced, hence a low visual quality of the reconstructed image is achieved.

In general, the DC subband has the highest energy content, and the high order subbands have the lowest energy content, especially the diagonal subbands (3, 6 and 9) as shown in Figure 4-1 and Table 4-2.

Energy Per Subband										
subband	0	1	2	3	4	5	6	7	8	9
energy	50,549,616	390,323	128,518	93,607	874,172	298,201	175,950	1,342,969	603,956	290,309

Table 4-2: Energy Distribution for a Level $M = 3$ Wavelet Decomposition of *Lena*.

When examining these figures, let's not forget to take into account the proportion of each subband. For example, there seems to be more energy in subband 8 than in subband 5. This is not true; In reality, there is more energy in subband 5 because its dimension is four times smaller than subband 8. Thus, by discarding higher order subbands, one should still expect a high PSNR since these subbands have the lowest energy content, thus little contributive effects to visual quality as measured by the MSE.

In trial 10 of Table 4-1, subbands 7, 8 and 9 (highest subbands) are set to zero. This means that 75% of the wavelet coefficients are discarded, and the image is reconstructed from the remaining 25% coefficients. As a result, a PSNR of 32.68 dB is

obtained. So this confirms the fact that eliminating high order subbands still produces an excellent level of visual quality for the reconstructed image.

With the image *Lena*, it just so happens that subband 6 contains less relevant information than subband 7. This can be confirmed by examining the PSNRs of trials 3 and 4. Therefore, by discarding subband 6 instead of subband 7 (see trial 11), one should expect an higher PSNR than in trial 10. However, the compression ratio should be less since subband 6 contains a lot less data than subband 7.

Setting too many subbands to zero has a profound impact on the PSNR even though the compression ratio can be relatively high. In trial 12 for example, subbands 4 to 9 are set to zero which accounts for 94% of the coefficients eliminated. A PSNR of 27.46 dB and a relatively high compression of 16 to 1 is achieved. In trial 13, only the DC is reconstructed and 98% of the coefficients are discarded. Still a fairly reasonable PSNR value of 23.89 dB is produced. The compression of 64 to 1 is very impressive, but the visual appearance is very blocky. Figure 4-2 shows the reconstructed images for trials 1, 10, 12, and 13.

As shown, figures 4-2 (a) and (b) have no degradation, and figure 4-2 (c) has a little bit of degradation. The reconstructed image for trial 13 has lots of artifacts. It was discovered toward the end of this research that these artifacts are mainly caused by the incapacity of the wavelet transform used in this work to reproduce artifact-free images when using the reference signal (DC subband) only [47]. This deficiency should be addressed in later research work.

Figure 4-2: Reconstructed Test Image of *Lena* for Trials 1, 10, 12 and 13.

In all, this wavelet transform, and many others, has very good compaction attributes, although it does not produced images that are 100% artifact-free. As a general rule, it is fairly safe to discard the high order wavelet subbands of an image before the coding process takes place. It usually does not have a dramatic impact on the visual quality of the reconstructed image, as seen from the above experiments.

In this work subbands 1 to 3, containing most of the energy in the image, will never be coded in order to preserve a certain acceptable level of visual image fidelity.

However, with a more efficient wavelet transform, as discussed above, the lower order subbands can be coded to achieve greater compression ratios with artifact-free results.

4.1.2 Local Means not Transmitted

In this section, some facts are presented in an attempt to justify the concept of not transmitting the local mean of each block in the wavelet domain. Table 4-3 lists the mean and variance of each subband for a level $M = 3$ wavelet decomposition of the image *Lena* [48], [49], [50]. Table 4-3 also includes the mean and variance for the original image *Lena* in the spatial domain for comparison purposes. As expected, the sum of the subband means and variances respectively is equal to the mean and variance of the original image (spatial domain). For example, the sum of the variances of subbands 0 to 9 is roughly equal to the variance of the original. The same goes for the mean.

Also, it is worth noting that the mean of the lowest frequency subband, the DC subband, is equal to the mean of the original image, while the remaining subbands have

Subband	Dynamic Range of each pixel		Statistical data		
	Minimum Range	Maximum Range	Mean	Variance	2nd Moment
0	- 4.116	230.000	99.051	2530.070	12341.215
1	- 51.340	54.051	-0.242	95.235	95.294
2	- 36.078	40.957	0.076	31.371	31.377
3	- 28.979	37.690	0.094	22.845	22.853
4	- 55.214	81.836	-0.070	53.350	53.355
5	- 42.590	42.153	0.023	18.200	18.201
6	- 30.351	32.947	0.001	10.739	10.739
7	- 58.314	60.203	-0.069	20.487	20.492
8	- 25.042	32.367	0.012	9.216	9.216
9	-17.296	21.389	0.009	4.430	4.430
Sum			98.885	2795.943	
Original	0.000	248.000	99.051	2795.355	12610.153

Table 4-3: Means and Variances for a Level $M = 3$ Wavelet Decomposition of *Lena*.

essentially zero mean. This observation can be used positively during the coding process when using VPIC in the wavelet domain. Knowing the fact that the mean of any subband is zero, there is no need to transmit the mean. This allows for greater compression to be achieved than the one obtained in the spatial domain (see Chapter 3).

However, further investigation needs to be done. The fact that the overall mean of each subband is zero does not necessarily imply that the local mean of a subimage ($n \times n$ block) is zero. Further analysis of a level $M = 3$ wavelet decomposition of *Lena* is shown in Table 4-4. Here, the statistical information tabulated is presented for each subband once it has been segmented into 4×4 blocks.

For example, Table 4-4 shows that subband 1 has a dimension of 64×64 pixels, and 256 blocks of size 4×4 . The minimum and maximum gradient values encountered throughout the 256 blocks in subband 1 was 0 and 19.83 respectively. Knowing

			dynamic range gradient values		dynamic range mean values		dynamic range block mean values					block type qty	
subband	blocks	dim	min	max	min	max	$\mu_{uniform}^{min}$	$\mu_{uniform}^{max}$	μ_{edge}^{min}	μ_{edge}^{max}	V_{min}	Nu	Ne
1	256	64 x 64	0	19.83	-9.92	8.77	-9.92	8.77	0	0	20	256	0
							-9.92	8.12	-6.60	8.77	10	238	18
							-9.92	4.48	-7.50	8.77	5	177	79
							-1.33	2.87	-9.92	8.77	2	99	157
							-1.33	0.81	-9.92	8.77	1	61	195
							-1.03	0.38	-9.92	8.77	.5	22	234
2	256	64 x 64	0	10.46	-2.71	2.55	-2.71	2.55	0	0	20	256	0
							-2.71	2.55	0	2.49	10	255	1
							-2.61	2.43	-2.71	2.55	5	238	18
							-1.97	1.08	-2.71	2.55	2	180	76
							-1.97	1.08	-2.71	2.55	1	129	127
							-1.97	0.61	-2.71	2.55	5	67	189
3	256	64 x 64	0	11.05	-3.92	5.92	-3.92	5.92	0	0	20	256	0
							-3.92	5.92	-1.69	1.29	10	254	2
							-3.92	2.86	-2.35	5.92	5	235	21
							-1.15	1.18	-3.92	5.92	2	186	70
							-0.58	0.56	-3.92	5.92	1	138	118
							-0.42	0.44	-3.92	5.92	5	86	170

Table 4-4: In-depth Analysis of Segmented Subbands for Level $M = 3$ Wavelet Decomposition of *Lena*.

			dynamic range gradient values		dynamic range mean values		dynamic range block mean values					block type qty	
subband	blocks	dim	min	max	min	max	$\mu_{uniform}^{min}$	$\mu_{uniform}^{max}$	μ_{edge}^{min}	μ_{edge}^{max}	∇_{min}	Nu	Ne
4	1024	128 x 128	0	26.59	-8.56	5.85	-8.86	5.85	-6.27	0	20	1021	3
							-8.86	5.85	-7.56	5.22	10	974	50
							-3.35	5.30	-8.86	5.85	5	854	170
							-3.35	4.00	-8.86	5.85	2	630	394
							-2.58	0.93	-8.86	5.85	1	391	633
							-1.05	0.79	-8.86	5.85	.5	136	888
5	1024	128 x 128	0	13.62	-4.68	4.23	-4.68	4.23	0	0	20	1024	0
							-4.68	4.23	-1.59	3.80	10	1019	5
							-2.72	3.05	-4.68	4.23	5	976	48
							-1.67	2.56	-4.68	4.23	2	829	195
							-1.05	1.11	-4.68	4.23	1	561	463
							-0.86	1.09	-4.68	4.23	.5	255	769
6	1024	128 x 128	0	9.08	-4.16	8.66	-0.86	0.57	-4.68	4.23	.25	74	950
							-4.16	8.66	0	0	20	1024	0
							-4.16	8.66	0	0	10	1024	0
							-2.22	8.66	-4.16	1.92	5	1002	22
							-2.21	8.66	-4.16	3.38	2	874	150
							-2.21	1.95	-4.16	8.66	1	668	356
7	4096	256 x 256	0	26.75	-10.30	7.08	-2.21	0.88	-4.16	8.66	5	326	698
							-2.21	0.88	-4.16	8.66	.25	103	921
							-0.48	0.56	-4.16	8.66	.25	103	921
							-10.30	7.08	-9.87	4.00	20	4090	6
							-8.00	5.46	-10.30	7.08	10	4034	62
							-6.06	5.46	-10.30	7.08	5	3798	298
8	4096	256 x 256	0	13.20	-3.99	6.35	-5.01	3.02	-10.30	7.08	2	2843	1253
							-4.01	2.54	-10.30	7.08	1.5	2277	1819
							-4.01	2.32	-10.30	7.08	1.3	1945	2151
							-2.21	1.82	-10.30	7.08	1	1400	2696
							-1.25	1.82	-10.30	7.08	.5	456	3640
							-1.25	1.57	-10.30	7.08	.25	136	3960
9	4096	256 x 256	0	13.20	-3.99	6.35	-3.99	6.35	0	0	20	4096	0
							-3.99	5.71	-2.99	6.35	10	4088	8
							-3.37	4.01	-3.99	6.35	5	4032	64
							-2.41	1.88	-3.99	6.35	2	3325	771
							-2.07	1.88	-3.99	6.35	1.5	2756	1340
							-2.07	1.88	-3.99	6.35	1.3	2434	1662
9	4096	256 x 256	0	7.84	-2.94	2.34	-2.07	1.88	-3.99	6.35	1.2	2259	1837
							-2.07	1.88	-3.99	6.35	1.1	2047	2049
							-2.07	1.86	-3.99	6.35	1	1826	2270
							-1.56	1.32	-3.99	6.35	.5	662	3434
							-1.06	1.32	-3.99	6.35	.25	182	3914
							-2.94	2.34	0	0	20	4096	0
9	4096	256 x 256	0	7.84	-2.94	2.34	-2.94	2.34	0	0	10	4096	0
							-2.94	2.34	-2.13	0.52	5	4089	7
							-2.07	1.95	-2.94	2.34	2	3785	311
							-2.07	1.60	-2.94	2.34	1	2328	1768
							-2.07	1.60	-2.94	2.34	.85	1923	2173
							-2.07	1.60	-2.94	2.34	.75	1621	2475
9	4096	256 x 256	0	7.84	-2.94	2.34	-2.07	1.60	-2.94	2.34	.65	1340	2756
							-2.07	1.60	-2.94	2.34	.5	892	3204
							-2.07	1.60	-2.94	2.34	.25	242	3854
							-2.07	0.85	-2.94	2.34	.25	242	3854
							-2.07	0.85	-2.94	2.34	.25	242	3854
							-2.07	0.85	-2.94	2.34	.25	242	3854

Table 4-4: In-depth Analysis of Segmented Subbands for Level $M = 3$ Wavelet Decomposition of *Lena* (con't).

the minimum and maximum gradient magnitudes in each subband helps determine the optimum value that should be set for the *maximum gradient* $\nabla_{\max}$ for quantizing purposes.

The local gradients are quantized as per the following uniform quantizer characteristics shown in Figure 4-3. $\nabla_{\min}$ is the lower gradient limit, and $\nabla_{\max}$ is the upper gradient limit. The local gradients in between $\nabla_{\min}$ and $\nabla_{\max}$ are quantized to 2^{ng} levels, where ng is the number of bits used to quantize local gradients.

Figure 4-3: Local Gradient Uniform Quantizer.

In Table 4-4, for every subband except the DC subband, the minimum and maximum subband means are given. For example, the smallest and the highest mean values out of the 256 blocks in subband 1 are -9.92 and 8.77 respectively for $\nabla_{\min}$ set to 20. By examining these values, it is easy to conclude that the mean of each block in any subband is not necessary zero, as opposed to the overall mean of zero for each subband as presented in Table 4-3. And so, can one still afford not to transmit the mean

of each block as performed in standard VPIC? Looking at these figures, it might not be a good strategy not to transmit the mean of each block. However, not shown in Table 4-4, is that the mean of most blocks in subbands 1 to 9 tends to be very close to zero. Only in a few extreme cases is the mean of a block is not close to zero, and these are the values shown in Table 4-4.

To be able to adequately make a decision on whether or not the local means should be transmitted to the decoder, one must have more statistical information about the mean of each local block and the variance of the means in each subband. And so, a program was designed to compute the mean of each block and the variance of the means of each subband for the image *Lena* with a level $M = 3$ wavelet decomposition. Table 4-5 depicts the variances of the means for each subband of Figure 4-1 for 4 x 4 block segmentation.

subband	dynamic range mean values		variance of the means
	min	max	
1	-9.92	8.77	4.45
2	-2.71	2.55	0.49
3	-3.92	5.92	0.79
4	-8.86	5.85	2.10
5	-4.68	4.23	0.44
6	-4.16	8.66	0.34
7	-10.30	7.08	1.21
8	-3.99	6.35	0.39
9	-2.94	2.34	0.16

Table 4-5: Subband Variances.

In Table 4-5, it would be too long to list the mean of every block for each subband. Instead, the minimum and maximum mean values of each subband is listed.

The last column of Table 4-5 shows the variance of the means for each subband. The variance of the mean gives us an idea of how much the mean of each block spreads out from the origin. This can be graphically presented in the form of subband histograms as shown in Figure 4-4. Here, the mean of each subband is given and the variance of the means of each subband, including its standard deviation, are listed.

Figure 4-4: Subband Histograms for $M = 3$ Wavelet Decomposition of *Lena*.

Except for subband #1, all other subbands (subbands 2 to 9) have variances close to zero. This means that most of the image blocks have a mean close to zero, and so, this

justifies the fact of not transmitting the local means to the decoder. As we will see in later on in the chapter, the omission of transmitting the mean to the decoder results in a PSNR decrease of at most 1 dB. The decrease in PSNR is mostly due to the local means of subband #1, containing most of the energy, not being transmitted.

It was mentioned earlier that the *minimum gradient threshold* $\nabla_{\min}$ determines whether a block is to be an uniform or an edge block. The number of uniform and edge blocks detected is indicated by N_u and N_e . The sum of N_u and N_e accounts for the total amount of blocks in an image or a subband. By varying the values of $\nabla_{\min}$, the means of uniform and edge blocks change since different ones are detected. This is confirmed by the smallest mean value for uniform blocks $\mu_{\text{uniform}}^{\min}$, the highest mean value for uniform blocks $\mu_{\text{uniform}}^{\max}$, the smallest mean value for edge blocks $\mu_{\text{edge}}^{\min}$ and the highest mean value for edge blocks $\mu_{\text{edge}}^{\max}$ in each subband which are computed. In turn, this can help to determine the best $\nabla_{\min}$ setting for each subband to obtain optimal compression/ fidelity of reconstruction.

4.1.3 Residual not Used

Referring back to the block diagram of standard VPIC at Figure 3-3, the residual r of each block is computed prior to assigning patterns to image blocks in the spatial domain. This allows for the mean of each image block to be coded separately. The calculation of the residual r in the spatial domain is a must because of the high potential average intensity differences that occur between image blocks [46]. As a result, each residual r has zero mean, and is used as the basis for pattern matching.

In the wavelet domain, it was earlier mentioned that most of the local means have zero mean. As a result, the residual $r = b - u$ does not have to be computed because the transform coefficients have very small values, in fact, very similar to *residual* values. Consequently, each block in a given subband is assumed to be residual ready and can be used as the basis for pattern matching. In addition, this strategy is cost effective since the overall execution of the algorithm is faster because no *residual* needs to be computed.

4.2 Binary Coding

In the spatial domain, the VPIC coding scheme is basically considered as a 2-level or binary image coding technique. It is binary because the basis functions with which the image is reconstructed are 2-level, binary patterns. The values of the gradient magnitude ∇_b of the edge block and the mean $\mu_{i,j}$ of each block are encoded and transmitted to the decoder to allow the appropriate values to be assigned to the binary pattern.

In this section, a modified version of standard VPIC that is adapted to wavelet subband images is presented. This modified version is called Wavelet Visual Pattern Image Coding (WVPIC). The processing of each subband, from this point forward in this thesis, is illustrated in the following block diagram at Figure 4-5.

Initially the image composed of raw data is mapped into the wavelet domain using a 6-tap Daubechies filters implementation as discussed in Chapter 2. The wavelet transform coefficients of each subband are then compressed and encoded through the WVPIC coding scheme, and transmitted to the decoder.

Figure 4-5: Block Diagram of the Compressed/Decompressed Image Coding Process.

At the decoder stage, the compressed transform coefficients of each subband are decoded by the decoder part of the WVPIC architecture to produce the decompressed or reconstructed wavelet transform coefficients. At this point, the transform coefficients are converted back into the spatial domain using the inverse discrete wavelet transform algorithm. Once all the reconstructed transform coefficients of each subband are mapped into the spatial domain, then every subband is inserted in their proper position in the overall image picture.

In the following sections, four different versions of WVPIC are presented and analyzed. These versions simply encompass various implementations of WVPIC with different block sizes and codebook sizes. At the end of each major section, a discussion of each WVPIC version performance is presented.

4.2.1 A 64K-Pattern WVPIC Coding Scheme With 4 x 4 Block Segmentation

Coding 4 x 4 blocks as binary images produces a possibility of 65,536 (64K) distinct patterns. The first implementation of the WVPIC algorithm was designed with a codebook which was constructed such that all possible 16-bit patterns (4 x 4 blocks) were represented, a total of 65,536 pre-defined vectors. A *full* codebook simply guarantees that all detected edge patterns can be reconstructed as faithfully as possible. In later stages of the optimization process of the WVPIC algorithm, different and reduced codebook sizes will be implemented to increase compression ratios.

During the early stages of the development of WVPIC one major problem arose. *Overflow* and *underflow* transform coefficients were detected once the reconstruction process was done. Extensive study revealed that certain vectors in the codebook (a few thousand) were the cause of these *overflow / underflow* problems. These code vectors are called *zero-gradient patterns*.

4.2.1.1 Zero-Gradient Patterns

When generating the 64K-Pattern codebook, it was discovered that it was impossible to compute the *oriented gradient coefficients* for certain visual patterns among the 65,536 possible patterns. It was discovered that any visual pattern which had the same number of positive and negative elements between the top and bottom halves, and the same number of positive and negative elements between the left and right halves produces a zero (0) gradient magnitude. It was decided that such visual pattern is to be called a *zero-gradient pattern*. *Zero-gradient patterns* were causing erroneous subimage pixel values (in the

form of overflow and underflow pixel values) that could only be noted during the decoding process.

Let's take a moment to explain what is meant by an *overflow* or *underflow* problem. In an image, which is composed of a number of picture elements (pixels) each requiring 8 bits per pixel, the pixel values must be between zero and 255 only. An *overflow* pixel value is one that is greater than 255, and an *underflow* pixel value is one that is less than zero.

As an example of a *zero-gradient pattern*, let us choose a pattern p , say, $p = 0x0ff0_{16}$. The corresponding bit map of pattern p_{0x0ff0} is:

$$p_{0x0ff0_{16}} = \begin{bmatrix} - & - & - & - \\ + & + & + & + \\ + & + & + & + \\ - & - & - & - \end{bmatrix}$$

Then, the partial gradient magnitudes in the x and y directions of pattern p_{0x0ff0} are computed as:

$$\begin{aligned} \nabla_x p &= \frac{\text{bottom 8 pixels}}{8} - \frac{\text{top 8 pixels}}{8} \\ &= \frac{1}{8} (4a + 4b - 4a - 4b) = 0 \end{aligned} \tag{4-1}$$

$$\begin{aligned} \nabla_y p &= \frac{\text{right 8 pixels}}{8} - \frac{\text{left 8 pixels}}{8} \\ &= \frac{1}{8} (4a + 4b - 4a - 4b) = 0 \end{aligned} \tag{4-2}$$

The gradient magnitude of pattern p_{0x0ff0} is:

$$|\nabla p| = \sqrt{\nabla_x p^2 + \nabla_y p^2} = 0 \quad 4-3$$

In all, any pattern (edge block) detected in an image which corresponds to a *zero-gradient pattern* in the codebook cannot be coded adequately with standard VPIC because overflow and underflow problems occur. Hence, whenever *zero-gradient patterns* are involved in an image, adequate encoding using standard VPIC is impossible, or else a poorer visual quality image is reconstructed as a result.

4.2.1.2 A 64K-Pattern WVPIC/MAE BTC Architecture With 4 x 4 Block Segmentation

The solution to the *zero-gradient pattern* problem is that zero-gradient detected patterns must be coded with a modified version of WVPIC. In this thesis, it was decided that zero-gradient detected patterns be coded using a coding technique known as Mean Absolute Error (MAE) BTC discussed in Chapter 2. However, for any other patterns, WVPIC is the main coding scheme used for the encoding process. Figure 4-6 shows a detailed block diagram of the modified version of 64K-Pattern WVPIC/MAE BTC. This whole encoding/decoding process is performed for each and every subband involved.

The 64K-Pattern WVPIC/MAE BTC architecture differs from standard VPIC in the encoding of the zero-gradient detected blocks. As shown in Figure 4-6, three additional blocks are added (identified in gray color) in the coder to code zero-gradient

Figure 4-6: The 64K-Pattern WVPIC/MAE BTC Block Diagram.

detected edge blocks independently from non zero-gradient edge blocks. These are, respectively:

- a. *compute BTC coefficients a & b ;*
- b. *BTC coefficients a & b quantization; and*
- c. *encode a & b coefficients (n_g bits).*

Once a zero-gradient edge pattern is detected, the computing of its BTC equivalent coefficients a and b is handled by the first block. The a coefficient represents the *high mean*, and the b coefficient represents the *low mean* of the edge block. Their mathematical expressions is written as follows:

$$a = \text{avg}(+res) = \frac{\sum +res}{\text{total number of } +res} \quad 4-4$$

$$b = \text{avg}(-res) = \frac{\sum -res}{\text{total number of } -res} \quad 4-5$$

where $+res$ (res for residual) denotes a positive pixel element, and $-res$ denotes a negative pixel element from the edge block. And so, a is equal to the average of the positive pixel elements and b is equal to the average of the negative pixel elements in an edge block.

For example, let block $b[119]$ be real data detected as an edge pattern in subband 1. The actual wavelet coefficients for $b[119]$ are listed below:

$$b[119] = \begin{bmatrix} 13.385997 & 14.633750 & -18.905514 & -5.953508 \\ 18.135355 & -22.061888 & -2.669317 & 8.104279 \\ -11.847732 & 10.452658 & 7.018513 & 2.890158 \\ -2.7336777 & -0.427074 & 1.343743 & -11.472719 \end{bmatrix}$$

The mean μ of block 119, $\mu[119]$ is:

$$\mu[119] = -0.00686$$

The residual r of block 119, ($r[119] = b[119] - \mu[119]$) is:

$$r[119] = \begin{bmatrix} 13.392683 & 14.640436 & -18.898829 & -5.946822 \\ 18.142040 & -22.055202 & -2.662631 & 8.110965 \\ -11.841045 & 10.459344 & 7.025199 & 2.896844 \\ -2.726991 & -0.420388 & 1.350429 & -11.466033 \end{bmatrix}$$

The BTC coefficients a and b are calculated as such:

$$\begin{aligned} a &= \text{avg}(+res) \\ &= \frac{\sum +res}{\text{total number of } +res} \\ &= \frac{(13.392683 + 14.640436 + 18.142040 + 8.110965 + 10.459344 + 7.025199 + 2.896844 + 1.350429)}{8} \\ &= 9.502243 \end{aligned}$$

$$\begin{aligned}
b &= \text{avg}(-res) \\
&= \frac{\sum -res}{\text{total number of } -res} \\
&= \frac{(-18.898829 - 5.946822 - 22.055202 - 2.662631 - 11.841045 - 2.726991 - 0.420388 - 11.466033)}{8} \\
&= -9.502243
\end{aligned}$$

Thus, the *high mean* and the *low mean* are computed as $a = 9.502243$ and $b = -9.502243$. Note that both the high and low means are equal with opposite sign values. This will come handy in a later section.

The next step consists in quantizing uniformly the BTC coefficients a and b , which is handled by the second block in Figure 4-6. Then, the next block, “encode a & b coefficients”, is responsible to encode the BTC coefficients which has been previously quantized as n_{btc} bits in two different binary buffers called btc_a and btc_b in binary format to be transmitted to the decoder.

One more thing that differentiates this achitecture from standard VPIC is the coding of the address of the pattern being correlated which is transmitted to the decoder. In standard VPIC, an additional bit is sent as part of the address code indicating to the decoder the orientation of the pattern (see chapter 3). In WVPIC with a 64K-Pattern codebook, the orientation information is not needed since the address transmitted by the coder refers directly to the code vector in the codebook. However, the decoder needs another kind of information for it to work properly, and that is, whether or not the edge block being decoded is a zero-gradient or non zero-gradient edge block. So in standard VPIC the additional bit dedicated to the orientation information is dedicated to indicating

the *edgetype* information in WVPIC. A bit “0” indicates that the edge block being coded is a non zero-gradient edge block, and a bit “1” a zero-gradient edge block.

In this version, coding a 4 x 4 uniform block is the same as standard VPIC. However, to code a 4 x 4 edge block, more bits are required because of the additional two buffers used to encode the BTC coefficients. The bits per block (bpb) required to code a 4 x 4 uniform and edge block are as follows:

$$\text{a. uniform block: } n_u = n_{uu} + 1 \text{ bpb} \quad 4-6$$

$$\text{b. edge block: } n_e = n_{ue} + (n_{btc} * 2) + n_g + n_p + 2 \text{ bpb} \quad 4-7$$

where the number of bits required to code an uniform block n_u is equal to the number of bits required to code the mean of an uniform block n_{uu} , plus one additional bit indicating to the decoder whether the block being processed is an uniform or edge block. The number of bits required to code an edge block n_e is equal to the sum of the number of bits required to code the mean n_{ue} , the BTC coefficients a and b ($n_{btc} * 2$), the gradient n_g , the correlated pattern address n_p , and two additional bits indicating the *blocktype* and the *edgetype* of the edge block.

At the decoder stage, zero-gradient edge blocks are also decoded independently from non zero-gradient edge blocks. In fact, the zero-gradient edge blocks and the non zero-gradient edge block oriented gradient coefficients have different destinations. A reconstructed image block $b_{i,j}^*$ constituting of a non zero-gradient block is reconstructed as (eqn. 3-16):

$$b_{i,j}^* = u_{i,j}^{**} + [g_{i,j}^{**} \cdot v_j] \quad 4-8$$

where $\mu_{i,j}^{**}$ represents the transmitted mean and $g_{i,j}^{**}$ the transmitted gradient of the image block being reconstructed. The code vector v_j represents the oriented gradient coefficients of the pattern denoted by j .

A reconstructed image block $b_{i,j}^*$ constituting of a zero-gradient block is reconstructed as:

$$b_{i,j}^* = u_{i,j}^{**} + [a|b \cdot \pm v_{i,j}] \quad 4-9$$

where the BTC coefficient a is multiplied strictly with the positive oriented gradient coefficients $+v_j$, and the BTC coefficient b is multiplied with the negative oriented gradient coefficients $-v_j$ only. Then, the result is added to the transmitted mean $\mu_{i,j}^{**}$ of the image block.

4.2.1.3 The BTC Oriented Gradient Coefficients in a 64K-Pattern Codebook

In developing the codebook to represent all possible visual patterns of a 4 x 4 block, the zero-gradient pattern problem must be taken into account because no *oriented gradient coefficients* can be computed for any *zero-gradient pattern*. This fact is again caused by the gradient value “0” produced anytime the gradient magnitude of a *zero-gradient pattern* is being computed. Now, the *zero-gradient patterns* have zero gradient

magnitudes, and it is impossible to generate the corresponding *oriented gradient coefficients* in the codebook. Another solution must be implemented.

The solution is a simple one. For non zero-gradient patterns, the corresponding *oriented gradient coefficients* are computed as in standard VPIC and properly initialized in the codebook as vectors as per section 3.3.2. For zero-gradient patterns, *BTC oriented gradient coefficients* are computed instead. The codebook *BTC oriented gradient coefficients* in the code vectors are computed based on the following two rules:

- a. for any positive element in the zero-gradient pattern bit map, its corresponding codebook vector element is set to “1”; and
- b. for any negative element in the zero-gradient pattern bit map, its corresponding codebook vector element is set to “-1”.

Therefore, there could only be a maximum of two different codebook *BTC oriented gradient coefficients*, “1” and “-1”. For example, the pattern p_{0xc972} is a zero-gradient pattern and its corresponding bit map looks like this:

$$p_{0xc972_{16}} = \begin{bmatrix} + & + & - & - \\ + & - & - & + \\ - & + & + & + \\ - & - & + & - \end{bmatrix}$$

Now, observing the above rules for computing the codebook *BTC oriented gradient coefficients* of a zero-gradient pattern, the bit map becomes,

$$\begin{bmatrix} + & + & - & - \\ + & - & - & + \\ - & + & + & + \\ - & - & + & - \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & 1 \\ -1 & -1 & 1 & -1 \end{bmatrix}$$

The next step consists in initializing the codebook properly as shown in Figure 4-7. Once the bit map of the image block is properly converted to “+1” and “-1”, each line is saved in parallel (as shown above) in the code vector memory. It can be seen that a code vector is 16 bits long for a 4 x 4 image block.

Figure 4-7: Initialization of the WVPIC/MAE BTC Codebook.

And so at the decoder stage, when a zero-gradient pattern is reconstructed, the codebook vector element of “1” is simply replaced by the value of the high mean a , and the codebook vector element of “-1” is simply replaced by the value of the low mean b .

Then the transmitted mean of the subimage is added as per eqn. 4-7. No multiplication operations are involved.

4.2.1.4 Results of the 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation

Using the transform coefficients of *Lena*, several experiments were carried out using the WVPIC/MAE BTC achitecture and the 64K-pattern codebook discussed above. The results are summarized in Table 4-6.

In trials 1 to 9, only one subband is processed by WVPIC/MAE BTC at a time. For these trials, the PSNR values are very high, but the compression ratios are not. The compression ratios are very low due to the large number of bits needed to transmit the pattern addresses and the WVPIC/MAE BTC parameters set to their maximum precision without coarse quantization. At this point, it was judged not necessary to set the parameters for optimum compression since further optimization of WVPIC/MAE BTC needs to be done.

trial	compressed subband (only)	discarded subband (only)	PSNR (dB)	C	bpp	WVPIC input parameters
1	1	-	30.33	1.03 : 1	7.8	$\nabla_{\min} = 5$ $n_{uu} = 8$ $n_g = 8$ $\nabla_{\max} = 20$ $n_{ue} = 8$ $n_p = 8$ $n_{btc} = 8$
2	2	-	33.44	1.03 : 1	7.8	same
3	3	-	34.77	1.03 : 1	7.8	same
4	4	-	32.46	1.08 : 1	7.4	same

Table 4-6: Experimental Results of the 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena*.

trial	compressed subband (only)	discarded subband (only)	PSNR (dB)	C	bpp	WVPIC input parameters
5	5	-	35.84	1.08 : 1	7.4	same
6	6	-	37.52	1.08 : 1	7.4	same
7	7	-	35.71	1.3 : 1	6.2	same
8	8	-	38.22	1.3 : 1	6.2	same
9	9	-	40.59	1.3 : 1	6.2	same
10	1 - 9	-	25.18	10.7 : 1	0.75	same
11	4 - 9	-	28.57	7.4 : 1	1.1	same
12	7 - 9	-	33.40	3.3 : 1	2.4	same
13	4 - 6	7 - 9	28.32	12.7 : 1	0.63	same
14	4 - 6	7 - 9	27.98	13 : 1	0.62	$\nabla_{\min} = 8$ $n_{uu} = 8$ $n_g = 8$ $\nabla_{\max} = 20$ $n_{ue} = 8$ $n_p = 8$ $n_{btc} = 8$
15	4 - 6	7 - 9	28.91	11.8 : 1	0.68	$\nabla_{\min} = 2$ $n_{uu} = 8$ $n_g = 8$ $\nabla_{\max} = 20$ $n_{ue} = 8$ $n_p = 8$ $n_{btc} = 8$
16	4 - 6	7 - 9	29.16	10.7 : 1	0.75	$\nabla_{\min} = 1$ $n_{uu} = 8$ $n_g = 8$ $\nabla_{\max} = 20$ $n_{ue} = 8$ $n_p = 8$ $n_{btc} = 8$
17	4 - 6	7 - 9	25.84	9.6 : 1	0.83	$\nabla_{\min} = .5$ $n_{uu} = 8$ $n_g = 8$ $\nabla_{\max} = 20$ $n_{ue} = 8$ $n_p = 8$ $n_{btc} = 8$

Table 4-6: Experimental Results of the 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena* (con't).

In trial 10, all the subbands are compressed except the DC subband. In fact, in this thesis, the DC subband is always transmitted without being compressed, otherwise

the visual quality of the reconstructed image would not be acceptable to the human eye.

The information content of the DC subband is too important.

The results of trial 10 are not too good. The PSNR of 25.18 dB for a compression of 10.7 : 1 is not that good. In Figure 4-8, a picture of the reconstructed image *Lena* for trial 10 is shown.

Figure 4-8: Reconstructed Test Image *Lena* for Trial 10 Input Parameters (PSNR = 25.68 dB, C = 10.7).

As one can see, lots of striped artifacts can be found. In fact, these artifacts are seen in any reconstructed WVPIC/MAE BTC images where the PSNR is less than 29 to 30 dB.

It is conjectured that these striped artifacts are caused by the coarse quantization of the transform coefficients and the actual wavelet transform used, which is not optimized for image compression. Figure 4-9 illustrates the effects of thresholding using 2-level quantization scheme such as VPIC. In Figure 4-9 [a], the values of a given image block are shown. When $\nabla_{\min}$ is set to an high threshold value, the image block is detected as an uniform block, and is reconstructed as shown in Figure 4-9 [b], that is, the transform coefficients of the uniform block are reconstructed as the mean of the image block, which was transmitted by the coder. When $\nabla_{\min}$ is set to a low threshold value the image block is detected as an edge block, and is thus reconstructed as an edge block, as per Figure 4-9 [c]. The transform coefficients, in that case, are assigned positive and negative values regardless of their original values due to equations 4-8 and 4-9. Thus, the more edge blocks detected, the more quantization that takes place, creating more artifacts.

Another explanation for the presence of artifacts in the reconstructed image is related to the *type* of wavelet transform used. As Villasenor et al. explain [47], the filter bank used to emulate the wavelet transform should be carefully researched so that it should not result in a decomposition/reconstruction that leads to artifacts in the image reconstructed from the reference signal alone (see trial 13 of Table 4-1).

The artifacts, which would be otherwise canceled by the information in the detail signals if lossless coding were used, can only be partially canceled when the detail signals undergo the severe compression necessary to achieve reasonable overall coding rates. As

a result, is it preferable to use a wavelet transform that leads to a relatively artifact-free reconstruction when using the reference signal (DC subband) only.

Figure 4-9: Effect of Thresholding *high* and *low* $V_{\min}$ Settings.

In trial 11, only subbands 4 to 9 are processed by WVPIC/MAE BTC. The PSNR of 28.57 dB is better, but compared to standard VPIC applied in the spatial domain, the compression ratio is still too low. In trial 12, only subbands 7 to 9 are processed resulting in an excellent PSNR of 33.40 dB while its compression ratio is quite low.

In Figure 4-10, a picture of the reconstructed image *Lena* for trial 13 is shown.

Figure 4-10: Reconstructed Test Image *Lena* for Trial 13 Input Parameters (PSNR = 28.32 dB, C = 12.7).

This time, only subbands 4 to 6 are compressed (in trial 13), while the higher order subbands 7 to 9 are simply discarded. As expected, the compression ratio is only moderate because subbands 1 to 3 are left untouched (not compressed), and they contribute most in terms of the energy content. The PSNR of 28.32 dB is similar to that of trial 11. The PSNR result of trial 13 is similar to trial 11 because subbands 7 to 9, which are discarded, do not excessively contribute due to their low energy content.

Note that most of the artifacts have disappeared. Those left are probably caused by the wavelet transform used as discussed earlier and the fact that subbands 7 to 9 have been discarded. Also note that the blockiness effects are gone.

Even though the $\nabla_{\min}$ is set to the same value in trials 10 and 13, the stripe artifacts found in trial 10 (see Figure 4-8) are not as intense here because subbands 1 to 3 are not processed through the coder. Most of the artifacts generated in trial 10 are caused by subbands 1 to 3 because the transform coefficients in subbands 1 to 3 contain most of the energy content among all transform coefficients in the test image.

Increasing the minimum threshold $\nabla_{\min}$, fewer edge blocks are detected, and hence the visual quality of the reconstructed image is poorer. This can be verified by examining trial 14 which has a low PSNR of 6.32 dB with a $\nabla_{\min}$ of 8. Its reconstructed image is shown at Figure 4-11. Here, even though the PSNR is low, the artifacts are not as intense compared to Figure 4-10 (see the nose areas). This proves again that when the $\nabla_{\min}$ is set to a high value, fewer stripe artifacts are manifested because fewer transform coefficients are reproduced as *large* transform coefficient amplitudes. Again, note that the blockiness effect is gone.

Figure 4-11: **Reconstructed Test Image *Lena* for Trial 14 Input Parameters (PSNR = 27.98 dB, $C = 13$).**

In trials 15 to 17 the minimum threshold $\nabla_{\min}$ is decreased gradually. If $\nabla_{\min}$ is decreased, more edge blocks are detected resulting in lower compression ratios and higher PSNR values. However, there comes a point when decreasing the $\nabla_{\min}$, that the PSNR of the reconstructed image starts declining. In Table 4-6, the turning point can be seen by examining the $\nabla_{\min}$ and PSNR in trial 17. While its $\nabla_{\min}$ is lower than the ones in

trials 15 and 16, the PSNR in trial 17 is lower. Therefore, one can conclude that detection of a larger number of edge blocks does not necessarily improve the visual quality of the reconstructed image.

The performance of this version of WVPIC/MAE BTC, as described in the previous sections, is not overly impressive. The results obtained in trial 15 are slightly better than the best performance produced by standard VPIC in spatial domain (see Table 3-2, trial 6). However, the PSNR values and compression ratios obtained are comparable to the ones obtained in standard VPIC applied in the spatial domain. In the next few sections, efforts are aimed at increasing the compression ratios by modifying somewhat the basic WVPIC/MAE BTC architecture.

4.2.2 A Modified 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation

In this version, two notions introduced in sections 4.1.2 and 4.1.3 are implemented. First, the local mean of any block is not transmitted, and second, the residual of each block is not pre-computed for reasons discussed in sections 4.1.2 and 4.1.3.

Additionally, it was briefly mentioned in section 4.2.1.2 that BTC coefficients a and b are equal, but have opposite signs. This is true for any BTC coefficients that are computed. Given that the number of positive residual components is equal to the number of negative residual components for any *zero-gradient pattern*, then using equations 4-4 and 4-5,

$$\begin{aligned}
 u &= a + b \\
 &= \frac{\sum +res}{8} + \frac{\sum -res}{8} \\
 &= 0
 \end{aligned}
 \tag{4-10}$$

And so the end result is that,

$$a = -b \tag{4-11}$$

Consequently, there is no need to transmit both coefficients to the decoder. Only one BTC coefficient needs to be transmitted to the decoder. In this research, it was decided that the BTC coefficient a is coded and transmitted only. At the decoder end, the decoded BTC coefficient is simply made equal to a and negated. Thus,

$$b_{\text{decoder}} = -a_{\text{transmitted}} \tag{4-12}$$

In Figure 4-12, a detailed block diagram of the modified 64K-Pattern WVPIC/MAE BTC implementation with a 64K-Pattern codebook is shown. Compared to the 64K-Pattern WVPIC/MAE BTC architecture presented at Figure 4-6, this architecture is clearly simpler. Here, once the transform coefficients have been segmented into 4 x 4 subimages, the gradient of each block is computed based on the values of the transform coefficients directly. The transform coefficients are represented by b . The local mean and residual are not computed since the mean of any block is generally zero or very close to zero, as per sections 4.1.2 and 4.1.3. And so, where the residual r was used, the actual transform coefficients b is now used instead.

Once the gradient of each block is computed, the gradient magnitudes or BTC coefficients a are encoded and transmitted for every edge block detected. As well, the pattern matching process is performed for every edge block detected. This time, however, the transform coefficients b are used for bit map conversion, not the residual r .

Because the local means are not computed earlier in the algorithm, no mean quantization and encoding is performed. Thus the infrastructure used for such purpose in the previous WVPIC/MAE BTC achitecture is not implemented here. The binary buffer μ^* used to store all encoded local means in the previous WVPIC/MAE BTC achitecture, is only used to indicate the *type* of each block to the decoder in this implementation. In this version, the μ^* binary buffer is called the *blocktype* buffer. When $\mu_i^* = 0$, the block being processed is of type uniform, and when $\mu_i^* = 1$, the block is an edge block.

At the decoder, the decoding process is identical to the previous WVPIC/MAE BTC implementation, except for two things. First, once the BTC coefficient a is decoded for a zero-gradient edge block only, the BTC coefficient b is created as per eqn. 4-12. Second, when a uniform block is being decoded, as identified by the *blocktype* buffer, the reconstructed transform coefficients of the block b^* are simply initialized to "0". Thus, a local mean of zero is created for each uniform block being reconstructed.

With all of these modifications implemented, the bits per block (bpb) required to code $n \times n$ uniform and edge blocks is as follows:

$$\text{a. uniform block: } n_u = N_u + N_e \text{ bpb} \quad 4-13$$

$$\text{b. edge block: } n_e = n_{btc} + n_g + n_p + 1 \text{ bpb} \quad 4-14$$

where n_u is equal to the sum of the number of uniform blocks N_u detected and the number of edge blocks N_e detected. The number of bpb required to code an edge block n_e is equal to the sum of the number of bits required to code the BTC coefficient n_{brc} , the gradient n_g , the address of the correlated pattern n_p selected to represent the edge block, and one bit to indicate to the decoder the *edgetype* of the edge block. As with the previous WVPIC/MAE BTC architecture, the *edgetype* bit tells the decoder whether the block being processed is a zero-gradient or non zero-gradient pattern.

A summary of the experiments carried out with the modified 64K-Pattern WVPIC/MAE BTC coding scheme with 4 x 4 image segmentation is listed in Table 4-7. In this version of WVPIC/MAE BTC, the compression ratio is considerably increased while the PSNRs have decreased by no more than one dB as compared to the previous WVPIC/MAE BTC algorithm developed in section 4.2.1.2. This can be verified by comparing trials 10 to 13 from Table 4-6 and trials 1, 3, 4, 6 from Table 4-7 respectively. In trial 5, one can see that the compression is 46 to one, however the reconstructed visual quality of the image (PSNR = 24.49 dB) is very poor.

trial	compressed subband (only)	discarded subband (only)	PSNR (dB)	C	bpp	WVPIC input parameters
1	1 - 9	-	24.45	31 : 1	0.23	$\nabla_{\min} = 5$ $n_{brc} = 8$ $n_p = 8$ $\nabla_{\max} = 20$ $n_g = 8$
2	1 - 3	-	27.06	1.05 : 1	7.6	same
3	4 - 9	-	28	12.9 : 1	0.62	same

Table 4-7: Experimental Results of the Modified 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena*.

trial	compressed subband (only)	discarded subband (only)	PSNR (dB)	C	bpp	WVPIC input parameters
4	7 - 9	-	32.4	3.8 : 1	2.1	same
5	1 - 6	7 - 9	24.49	46 : 1	0.17	same
6	4 - 6	7 - 9	28	15 : 1	0.53	same
7	4 - 6	7 - 9	28.64	12.3 : 1	0.65	$\nabla_{\min} = 1$ $n_{bt} = 8$ $n_p = 8$ $\nabla_{\max} = 20$ $n_g = 8$

Table 4-7: Experimental Results of the Modified 64K-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena* (con't).

Comparing trials 6 and 7, a slightly higher compression ratio is obtained from trial 7 than trial 6 due to the minimum threshold $\nabla_{\min}$ being decreased to one. This notion was discussed in the previous section. The reconstructed image obtained in trial 7 is shown at Figure 4-13. Still, artifacts are the main problem, especially when the compression has increased.

Overall, in this version of WVPIC/MAE BTC where the mean and 50% of the BTC coefficient information are not transmitted to the decoder while using the transform coefficient directly instead of the residuals, the compression ratios obtained are much better compared to the first version of WVPIC/MAE BTC developed in section 4.2.1.2. In this version of WVPIC/MAE BTC, performance in terms of compression ratio has surpassed the one achieved by standard VPIC in the spatial domain. The fidelity of the reconstructed images through the PSNR parameter is down by one dB at most, and artifact problems are still a major problem.

Figure 4-13: Reconstructed Test Image *Lena* for Trial 7 Input Parameters (PSNR = 28.64 dB, $C = 12.3$).

A drawback of the modified WVPIC/MAE BTC is the size of the codebook used. The dimension of a 64K-pattern codebook can be evaluated as followed. There are 65,536 possible patterns that can be addressed in a 16-bit word. Thus there are 65,536 code vectors in the codebook. Each vector of *oriented gradient coefficients* is saved in *float* format at no cost. In a 32-bit computer environment, such as the Pentium or Unix

systems, a *float* value uses exactly four bytes of memory. Each vector represents a 4 x 4 image pattern, and requires 16 *oriented gradient coefficients*. Thus, the size of a 64K-Pattern codebook is calculated as:

$$65,536 \text{ possible patterns} * 16 \text{ bits per pattern} * 4 \text{ bytes} = 4,194,304 \text{ bytes}$$

Now let's see what happen when the size of the codebook is decreased, that is, the number of code vectors contained in the codebook is reduced.

4.2.3 A 256-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation

To increase the compression ratios of WVPIC/MAE BTC further, the number of coded vectors used to represent all possible edge blocks detected in the coding and decoding processes can be reduced. The key point for using a reduced code vector space in a codebook is to choose a set of code vectors which can represent all possible patterns detected as accurately as possible.

In a recent IEEE publication [38], K. W. Chan and K. L. Chan proposed an enhanced set of visual patterns that cover textured structures not taken into account by the standard set of VPIC visual patterns recommended by Bovik and Chen [4]. They speculate that due to the limited set of spatial visual patterns proposed by Chen and Bovik, the visual quality of images coded by VPIC cannot be further improved. As discussed in section 3.4.4, adding more edge patterns by decreasing V_{min} , does not

improve the visual reconstructed quality of the image. This concept was further confirmed by [38].

The set of enhanced visual patterns proposed in [38] is a 256-member set of textured patterns which includes the 28 visual patterns originally proposed by Bovik and Chen. This set, as in standard VPIC, is directly mapped by WVPIC/MAE BTC via an integrated codebook. The extended set of textured patterns is constructed to preserve horizontal, vertical, and diagonal straight edges as in standard VPIC. In addition, it preserves shifted edges, thin lines, stripes (two to three pixels wide), corners, tilted corner edges and some other irregular but structured patterns.

The first 128 extended visual patterns are illustrated in Figure 4-14.

Figure 4-14: The 256-Pattern WVPIC/MAE BTC Enhanced Texture Patterns.

The remaining 128 are the complements of the ones illustrated. There are of course both types of patterns, the *zero-gradient patterns* and *non zero-gradient patterns*, in the proposed 256 patterns. And so, the codebook for such a set of visual patterns is generated in the same way as explained in the previous sections.

The main difference between the 64K-Pattern and 256-Pattern codebooks, is the resulting size of the codebooks. Instead of using a codebook of size 4,194,304 bytes as in the case of a 64K-Pattern codebook, a codebook of size 16,384 bytes is used instead. This is a tremendous reduction in memory size requirements. Also, by using a reduced pattern codebook, a tremendous reduction in the number of bits required to address all 256 pre-defined coded vectors is needed. Instead of using a 16-bit word for addressing all 65,536 coded vectors, only a 8-bit word is required to address 256 coded vectors. This means that the bits requirements to represent all of the encoded addresses in j (transmitted by the coder to the decoder) is reduced by half.

The achitecture of the 256-Pattern WVPIC/MAE BTC encoder is identical to the Modified 64K-Pattern WVPIC/MAE BTC achitecture shown in Figure 4-12. The only difference between the two architectures is the size of the codebook used. Table 4-8 is a list of the experimental results carried out using the 256-Pattern WVPIC/MAE BTC achitecture with 4 x 4 block segmentation on the image *Lena*. In trial 1, all the subbands are processed through the 256-Pattern WVPIC/MAE BTC algorithm, resulting in a PSNR value of 24.03 dB and a compression ratio of 34.2 to 1. As opposed to trial 1 in Table 4-7, the compression ratio obtained here is better while the PSNR suffered a bit by a difference of 0.42 dB. The visual quality of the reconstructed image is however unacceptable.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	-	24.03	34.2 : 1	.234	$\nabla_{\min} = 5$ $n_{btc} = 8$ $n_p = 8$ $\nabla_{\max} = 20$ $n_g = 8$
2	1 - 3	-	26.57	1.1 : 1	7.27	same
3	4 - 9	-	27.53	13.4 : 1	.597	same
4	7 - 9	-	32.61	3.9 : 1	2.05	same
5	1 - 6	7 - 9	24.03	49.1 : 1	.163	same
6	4 - 6	7 - 9	27.56	15.2 : 1	.526	same
7	4 - 6	7 - 9	27.61	13.2 : 1	.606	$\nabla_{\min} = 1$ $n_{btc} = 8$ $n_p = 8$ $\nabla_{\max} = 20$ $n_g = 8$

Table 4-8: Experimental Results of the 256-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena*.

Comparing trials 2 to 4 from Table 4-7 and Table 4-8, a slightly higher compression ratio is obtained with the 256-Pattern scheme because the address word length needed to address all 256 possible code vectors is eight bits instead of the 16 bits required for the 64K-Pattern version. A maximum difference of 0.49 dB in PSNR is manifested between trials 2 to 4. Of course, the visual quality of the reconstructed image of *Lena* in trial 4 (PSNR of 32.61) is excellent, however the compression is quite low.

By just discarding the same subbands (subbands 7 to 9), a slightly higher PSNR and compression ratio are obtained (see trial 10 from Table 4-1). Surprisingly, the coding of the high order subbands (subbands 7 to 9) only is not recommended. The compression ratio of 49.1 to 1 obtained in trial 5 is quite impressive, however the PSNR is way too low.

Figure 4-15 illustrates the reconstructed picture of *Lena* in trial 7.

Figure 4-15: Reconstructed Test Image *Lena* for Trial 7 Input Parameters (PSNR = 27.61, C = 13.2).

Artifacts are still a major problem due to coarse quantization. In trials 6 and 7, the compression ratios obtained are slightly better than the ones listed in Table 4-7, while the PSNR values are a bit lower.

Overall, the 256-Pattern WVPIC/MAE BTC coding scheme has better compression performance and slightly lower PSNRs than its predecessor, the modified 64K-Pattern coding scheme. Although the compression ratio of the 256-Pattern WVPIC/MAE BTC achieved the best performance up to this point, the PSNR has declined a bit. Artifact problems are very obvious still.

4.2.4 A 28-Pattern WVPIC Coding Scheme With 4 x 4 Block Segmentation

The 28-Pattern WVPIC/MAE BTC coding scheme is implemented with the same set of visual patterns developed and proposed by Chen and Bovik [4]. The experimental results obtained by using the 28-Pattern implementation is listed in Table 4-9.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	-	23.55	36 : 1	.222	$\nabla_{\min} = 5$ $n_{btc} = 8$ $n_p = 8$ $\nabla_{\max} = 20$ $n_g = 8$
2	1 - 3	-	26.03	1.1 : 1	7.27	same
3	4 - 9	-	27.16	13.6 : 1	.588	same
4	7 - 9	-	32.43	3.9 : 1	2.05	same
5	1 - 6	7 - 9	23.58	50.8 : 1	.158	same
6	4 - 6	7 - 9	27.23	15.3 : 1	.523	same
7	4 - 6	7 - 9	27.09	13.7 : 1	.584	$\nabla_{\min} = 1$ $n_{btc} = 8$ $n_p = 8$ $\nabla_{\max} = 20$ $n_g = 8$

Table 4-9: Experimental Results of the 28-Pattern WVPIC/MAE BTC Coding Scheme With 4 x 4 Block Segmentation on Test Image *Lena*.

Comparing Table 4-8 and Table 4-9, a higher compression ratio is produced overall with a slightly lower PSNR average than that achieved by the 256-Pattern coding scheme. A maximum increase of 0.01 compression ratio is obtained using the 28-Pattern version (see trial 6) , and a decrease of 0.33 in PSNR is manifested between the 256-Pattern and the 28-Pattern algorithms.

Illustrated at Figure 4-16, is the reconstructed test image *Lena* for trial 6.

Figure 4-16: Reconstructed Test Image *Lena* for Trial 6 Input Parameters (PSNR = 27.23, C = 15.3).

Similarly to Figure 4-15, artifacts are very obvious due to coarse quantization of the image data.

4.2.5 Analysis of the WVPIC/MAE BTC Coding Schemes with 4 x 4 Block Segmentation

By examining Tables 4-6, 4-7, 4-8 and 4-9, it is found that the best combination of compressed and discarded subbands is when subbands 1 to 3 are left untouched, subband 4 to 6 compressed through the WVPIC/MAE BTC coding scheme with a minimum threshold $\nabla_{\min}$ set to one, and subbands 7 to 9 are discarded completely. These set ups correspond to trials 16 in Table 4-6, and trials 7 in Tables 4-7, 4-8, and 4-9. Compression ratios of 10.7 to 15.3 with PSNRs of 29.16 to 27.23 are produced. While the compression ratios attained are not too bad, the PSNRs obtained are still too low.

Decreasing the minimum threshold $\nabla_{\min}$ to detect more edge blocks does not have a tremendous impact on the visual quality of the reconstructed image as seen in each version. As well, by decreasing the $\nabla_{\min}$, more edge blocks are detected, which increases the number of bits required to code the subbands because edge blocks require more bits to code than uniform blocks. As discussed in section 4.2.1.4, contributive effects are not steady as the $\nabla_{\min}$ is decreased.

In all, by completely discarding the local mean information, not using the residuals of each $n \times n$ block, using only one BTC coefficient instead of two, and reducing the number of patterns to represent all possible detected edge blocks from 65,536 to 28, an overall compression ratio of roughly 15 to 1 was achieved for the best

input set up, and the visual quality of the reconstructed images increased by only 0.88 dB compared to the best PSNR achieved in the spatial domain.

The overall performances of the different versions of WVPIC coding schemes are presented in Figure 4-17. The algorithm which has the highest PSNR is the 64K-Pattern WVPIC/MAE BTC. However, the 64K-Pattern version has the lowest performance in term of compression. On the other hand, the 28-pattern implementation has the highest compression as expected, but the lowest PSNR.

Figure 4-17: The Performance of the Different WVPIC/MAE BTC 4 x 4 Block Segmentation Implementations Using Test Image *Lena*.

As a result, the performance of WVPIC/MAE BTC applied to wavelet coefficients is comparable to the performance obtained by standard VPIC in the spatial domain in term of PSNR. However, the compression performance obtained in the wavelet domain is better.

At this point, it was decided that it would be a good idea to use WVPIC/MAE BTC coding scheme on 2 x 2 blocks instead of 4 x 4 blocks. The visual quality of the reconstructed image should be better while the compression ratio should decrease.

4.2.6 A 16-Pattern WVPIC/MAE BTC Coding Scheme With 2 x 2 Block Segmentation

The WVPIC/MAE BTC coding scheme designed to work on 2 x 2 image blocks has exactly the same architecture as the coding schemes developed to work on 4 x 4 image blocks, which was introduced in the previous sections. A statistical analysis of a level $M = 3$ wavelet decomposition of *Lena* was performed using 2 x 2 image block segmentation. The resulting statistical information, which is tabulated in Table 4-10, is presented for each subband.

Examining the range of the gradient magnitudes, the best setting for the maximum gradient $\nabla_{\max}$ is 80. The information shown in Table 4-10 can also be used to help determine the proper minimum gradient threshold $\nabla_{\min}$.

			dynamic range gradient values		dynamic range mean values		dynamic range block mean values					block type qty	
subband	blocks	dim	min	max	min	max	$\mu_{\text{uniform}}^{\text{min}}$	$\mu_{\text{uniform}}^{\text{max}}$	$\mu_{\text{edge}}^{\text{min}}$	$\mu_{\text{edge}}^{\text{max}}$	$\nabla_{\min}$	Nu	Ne
1	1024	64 x 64	0	80.96	-17.20	22.23	-17.20	22.23	0	0	81	1024	0
							-17.20	22.23	-9.88	7.05	70	1021	3
							-17.20	22.23	-12.09	7.05	60	1018	6
							-16.76	22.23	-17.20	21.31	50	1008	16
							-16.76	22.23	-17.20	21.31	40	984	40
							-16.76	22.23	-17.20	21.31	30	954	70
							-16.10	22.23	-17.20	21.31	20	860	164
							-11.73	15.19	-17.20	22.23	10	689	335
							-7.15	5.55	-17.20	22.23	5	518	506
							-2.61	3.11	-17.20	22.23	1.5	208	816
							-1.57	3.11	-17.20	22.23	1.4	192	832

Table 4-10: In-depth Analysis of 2 x 2 Segmented Subbands for Level $M = 3$ Wavelet Decomposition of Test Image *Lena*.

			dynamic range gradient values		dynamic range mean values		dynamic range block mean values					block type qty	
subband	blocks	dim	min	max	min	max	$\mu_{uniform}^{min}$	$\mu_{uniform}^{max}$	μ_{edge}^{min}	μ_{edge}^{max}	∇_{min}	Nu	Ne
2	1024	64 x 64	0	50.82	-9.76	15.03	-9.76	15.03	0	0	51	1024	0
							-9.76	15.03	0	9.86	40	1019	5
							-9.73	15.03	-9.76	9.86	30	1011	13
							-9.73	8.62	-9.76	15.03	20	978	46
							-9.73	8.62	-9.76	15.03	10	883	141
							-8.78	8.62	-9.76	15.03	5	756	268
							-8.78	8.62	-9.76	15.03	2	482	542
							-8.78	8.62	-9.76	15.03	1.8	443	581
							-1.58	2.32	-9.76	15.03	1.7	425	599
							-0.78	1.55	-9.76	15.03	0.5	103	921
3	1024	64 x 64	0	39.27	-10.72	12.46	-10.72	12.46	0	0	40	1024	0
							-10.72	12.46	0	6.56	30	1022	2
							-10.09	12.46	-10.72	12.01	20	1003	21
							-10.09	12.46	-10.72	12.01	10	916	108
							-10.09	12.46	-10.72	12.01	5	773	251
							-1.54	4.24	-10.72	12.01	1.5	458	566
							-1.54	0.94	-10.72	12.01	1	313	711
4	4096	128 x 128	0	82.32	-24.89	19.61	-24.89	19.61	0	0	83	4096	0
							-24.89	19.61	-13.50	7.09	70	4092	4
							-24.89	19.61	-14.28	9.26	60	4081	15
							-24.89	18.28	-20.29	19.61	50	4056	40
							-24.89	18.09	-20.76	19.61	40	4029	67
							-24.89	18.09	-22.48	19.61	30	3954	142
							-24.89	18.09	-22.48	19.61	20	3776	320
							-24.89	16.44	-22.48	19.61	10	3398	698
							-11.64	12.03	-24.89	19.61	5	2826	1270
							-1.91	3.29	-24.89	19.61	0.5	194	3902
5	4096	128 x 128	0	47.79	-10.93	15.72	-1.45	1.33	-24.89	19.61	0.25	52	4044
							-10.93	15.72	0	0	48	4096	0
							-8.90	11.05	-10.93	15.72	30	4077	19
							-8.07	11.05	-10.93	15.72	20	4018	78
							-7.98	8.22	-10.93	15.72	10	3759	337
							-7.98	4.76	-10.93	15.72	5	3333	763
							-2.84	2.50	-10.93	15.72	1	1025	3071
6	4096	128 x 128	0	30.35	-11.40	11.50	-1.39	1.74	-10.93	15.72	0.5	296	3800
							-11.40	11.50	0	0	31	4096	0
							-11.40	11.50	-4.57	5	20	4076	20
							-8.22	11.50	-11.40	6.88	10	3896	200
							-8.22	11.50	-11.40	6.88	5	3507	589
							-3.73	11.50	-11.40	10.52	1	1185	2911
7	16384	256 x 256	0	85.90	-23.55	23.10	-1.00	1.25	-11.40	11.50	0.25	113	3983
							-23.55	23.10	0	0	86	16384	0
							-23.55	23.10	-15.16	0	70	16382	2
							-18.03	15.18	-23.55	23.10	30	16244	140
							-18.03	14.38	-23.55	23.10	20	15958	426
							-18.03	12.51	-23.55	23.10	10	15014	1370
							-14.71	11.04	-23.55	23.10	5	12924	3460
							-7.15	5.25	-23.55	23.10	1	1987	14397
							-1.97	3.89	-23.55	23.10	0.25	129	16255

Table 4-10: In-depth Analysis of 2 x 2 Segmented Subbands for Level $M = 3$ Wavelet Decomposition of Test Image *Lena* (con't).

			dynamic range gradient values		dynamic range mean values		dynamic range block mean values					block type qty	
subband	blocks	dim	min	max	min	max	$\mu_{\text{uniform}}^{\text{min}}$	$\mu_{\text{uniform}}^{\text{max}}$	$\mu_{\text{edge}}^{\text{min}}$	$\mu_{\text{edge}}^{\text{max}}$	∇_{min}	Nu	Ne
8	16384	256 x 256	0	43.27	-12.78	13.43	-12.78	13.43	0	0	44	16384	0
							-12.78	11.37	-8.6	13.43	15	16204	180
							-12.78	7.57	-9.87	13.43	10	15810	574
							-12.78	5.48	-9.87	13.43	5	14126	2258
							-6.62	5.48	-12.78	13.43	4	12882	3502
							-6.28	5.48	-12.78	13.43	3	10626	5758
9	16384	256 x 256	0	21.72	-7.87	4.67	-7.87	4.67	0	0	22	16384	0
							-7.87	4.67	-4.24	4.67	9	16271	113
							-7.87	4.65	-4.24	4.67	7	16096	288
							-7.87	4.65	-4.24	4.67	5	15443	941
							-4.44	4.65	-7.87	4.67	4	14521	1863
							-4.42	3.99	-7.87	4.67	3	12571	3813
							-4.42	3.63	-7.87	4.67	2	8720	7664

Table 4-10: In-depth Analysis of 2 x 2 Segmented Subbands for Level $M = 3$ Wavelet Decomposition of Test Image *Lena* (con't).

In a 2 x 2 segmentation environment, there can be only $2^4 = 16$ possible patterns that can be represented by a 2 x 2 subimage. And so, the total size of a 16-Pattern WVPIC/MAE BTC in a 2 x 2 block segmentation environment is:

$$16 \text{ possible patterns} * 4 \text{ bits per pattern} * 4 \text{ bytes} = 256 \text{ bytes}$$

Some tests using the 16-Pattern WVPIC/MAE BTC achitecture with 2 x 2 block segmentation have been carried out using the image *Lena* as the main test image. The experimental results are presented at Table 4-11.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	-	27.08	9.1 : 1	.879	$\nabla_{\text{min}} = 5$ $n_{\text{bic}} = 8$ $n_p = 4$ $\nabla_{\text{max}} = 20$ $n_g = 8$
2	1 - 3	-	29.69	1.04 : 1	7.69	same
3	4 - 9	-	30.44	6.7 : 1	1.19	same

Table 4-11: Experimental Results of the 16-Pattern WVPIC/MAE BTC Coding Scheme With 2 x 2 Block Segmentation Using Test Image *Lena*.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
4	7 - 9	-	34.9	3.2 : 1	2.5	same
5	4 - 6	7 - 9	27.91	14.5 : 1	.550	$\nabla_{\min} = 40$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 8$
6	4 - 6	7 - 9	28.19	14.4 : 1	.555	$\nabla_{\min} = 30$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 8$
7	4 - 6	7 - 9	29.41	13.2 : 1	.608	$\nabla_{\min} = 10$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 8$
8	4 - 6	7 - 9	29.87	11.8 : 1	.677	$\nabla_{\min} = 5$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 8$
9	4 - 6	7 - 9	30.13	9.6 : 1	.835	$\nabla_{\min} = 2$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 8$
10	4 - 6	7 - 9	30.22	7.2 : 1	1.1	$\nabla_{\min} = .5$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 8$
11	4 - 6	7 - 9	28.35	11.55 : 1	.692	$\nabla_{\min} = 2$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_g = 0$
12	4 - 6	7 - 9	29.74	9.6 : 1	.835	$\nabla_{\min} = 2$ $n_{btl} = 8$ $n_p = 4$ $\nabla_{\max} = 20$ $n_g = 8$
13	4 - 6	7 - 9	14.60	10.7 : 1	.749	$\nabla_{\min} = 2$ $n_{btl} = 4$ $n_p = 4$ $\nabla_{\max} = 20$ $n_g = 4$
14	4 - 6	7 - 9	28.38	10.5 : 1	.765	$\nabla_{\min} = 2$ $n_{btl} = 4$ $n_p = 4$ $\nabla_{\max} = 40$ $n_g = 8$

Table 4-11: Experimental Results of the 16-Pattern WVPIC/MAE BTC Coding Scheme With 2 x 2 Block Segmentation Using Test Image *Lena* (con't).

The overall PSNR achieved by the 16-Pattern WVPIC/MAE BTC coding scheme applied on 2 x 2 subimages is roughly two dB better than in the 4 x 4 versions. The compression obtained in both schemes are somewhat comparable, although the compression performance of the 2 x 2 structure is slightly lower than that of the 4 x 4 structure. These results were to be expected. The higher PSNR values obtained in this version are attributed to the smaller image blocks of 2 x 2. Because quantization is

performed on smaller image blocks the difference between the reconstructed image and the original image (the reconstructed error) is smaller. The lower compression achievements are attributed to the larger number of image blocks to be coded.

Figure 4-18, the reconstructed test image *Lena* for trial 10 is shown. As we can see, there are no more artifacts present in this picture. This is to be expected since the overall error, as discussed above, is smaller.

Figure 4-18: Reconstructed Test Image *Lena* for Trial 10 Input Parameters (PSNR = 30.22 dB, $C \approx 7.2$).

4.2.7 Analysis and Limitations of VPIC and WVPIC/MAE BTC Coding Schemes

The performances of the VPIC (operating in the spatial domain), Modified 64K-Pattern WVPIC/MAE BTC coding schemes operating on 4×4 image blocks and the 16-Pattern WVPIC/MAE BTC coding scheme using 2×2 image blocks are summarized in Figure 4-19. The Modified 64K-Pattern WVPIC/MAE BTC architecture was chosen to represent all other WVPIC/MAE BTC versions developed to operate on 4×4 image blocks because it has the best overall performances in term of compression and PSNR characteristics.

Figure 4-19: The Performance of the 2×2 / 4×4 WVPIC/MAE BTC and 4×4 VPIC Coding Schemes.

As one can see in Figure 4-19, the 2×2 scheme has a better PSNR for most compression performance than that of the WVPIC/MAE BTC operating on 4×4 image blocks. In fact, the 2×2 WVPIC/MAE BTC coding scheme is the only architecture that produces artifact-free reconstructed images. Also shown in Figure 4-19, is the

performance of standard VPIC. The VPIC coding scheme produces lower PSNR outputs and has a broader range of compression. Overall, the visual quality of the reconstructed image is better with the 2 x 2 implementation with slightly lower compression ratios than VPIC.

In this section, the implementation of VPIC in the wavelet domain has been demonstrated and analyzed. It is found that VPIC and WVPIC/MAE BTC have the following attributes:

- a. low visual quality of the reconstructed image;
- b. poor compression ratio;
- c. low architectural complexity (lower in the wavelet domain);
- d. low computational complexity (as per [4-3]);
- e. small codebook requirement; and,
- f. universal codebook.

Although moderate compression results were achieved, the visual quality reflected by the PSNR results were not as expected. In both the spatial and wavelet domains, the maximum PSNR performances obtained were between 28 and 30.22 dB, respectively. These values are just not high enough to satisfy the visual quality of reconstructed images required by the human visual system.

The low visual quality of the reconstructed image, as confirmed by the low PSNR outputs obtained during the testing, shows that the VPIC and WVPIC/MAE BTC architectures do not produce satisfactory results at this time. The compression ratios obtained were medium performances. The main advantage of the VPIC or WVPIC approaches is the relatively small and universal codebook requirements and its simple

architecture. The codebook is universal because the pre-defined code vectors are pattern driven and visually significant to the human eye. The code vectors do not rely on image content, and so, it does not matter what kind of image is being compressed/decompressed. The internal codebook code vectors, representing visually important patterns, are such that they can be used to represent any type of image patterns. Additionally, there is no training process required by VPIC or WVPIC in the development of the codebooks.

The poor visual quality of the typical reconstructed image is mainly due to the type of local statistical information transmitted such as the gradient magnitudes, the wavelet transform used, and the fact that the VPIC and WVPIC/MAE BTC coding techniques are basically 2-level image coding architectures. As discussed by Wu and Coll [51], a 2-level moment-preserving quantizer coding scheme is simply not able to code a highly busy image with satisfactory fidelity.

In the following section, a multilevel coding scheme is presented in the hope of achieving higher performances than that obtained by standard VPIC and WVPIC/MAE BTC.

4.3 Ternary Coding

To achieve higher visual fidelity of digital images than standard VPIC and WVPIC/MAE BTC, multilevel coding architectures must be sought. Conventional 2-level moment-quantizers such as Block Truncation Coding (BTC), Vector Quantization (VQ), standard VPIC, and WVPIC/MAE BTC are simply not powerful enough to compress images while maintaining a high degree of reconstruction fidelity.

In this section, a new multilevel image coding scheme based on WVPIC/MAE BTC developed in section 4.2 is presented. This image coding design uses the concept of ternary coding which mean that its achitecture is based on 3-level quantization. Ternary coding seems to blend naturally with subband data because of the nature of data contained in wavelet information as shown in Figure 4-20.

Figure 4-20: Ternary Coding of a Given 4 x 4 Image Block Wavelet Transform Coefficients.

A given 4×4 wavelet transform coefficients image block is shown in Figure 4-20(a). In Figure 4-20(b), the new values re-adjusted by undergoing binary coding are indicated by empty circles $\bigcirc$. Figure 4-20(b) illustrates the resulting wavelet coefficients values after using a 2-level quantizer to coded an image block. All values close to zero are reproduced as larger positive ($+t$) or negative ($-t$) values as the case maybe. In Figure 4-20(c), the wavelet coefficients are ternary coded. Any coefficients equal to or larger than $+t$ or $-t$ are reproduced as $+t$ and $-t$ respectively. The remaining coefficients, the ones between $+t$ and $-t$ are set to zero. The result is that the wavelet coefficients are reproduced more naturally to the original coefficients.

The new image coding scheme developed using ternary coding notions is called *Three-Level Wavelet Minimum Maximum Error Block Truncation Coding* (Three-Level WMME BTC). There are two versions of Three-Level WMME BTC coding schemes presented in this section. The first version developed operates on 4×4 image blocks, and the second operates on 2×2 image blocks.

4.3.1 The Three-Level WMME BTC Coding Scheme

The Three-Level WMME BTC image coding scheme utilizes several concepts mainly from two image coding techniques, namely the WVPIC/MAE BTC developed in this thesis and the multilevel BTC developed by Wu and Coll [51]. The coding of edge patterns is quite different from standard VPIC. The coding of edge blocks is inspired by the work done by Wu and Coll, where the coding algorithm minimizes the maximum

quantization error. It allows for a close replica of the original transform coefficients, and a robust coding of the statistical distribution within each coefficient block.

As the term *3-level coding* indicates, 3-level coding means that each image block is coded using three levels of quantization rather than using a 2-level quantization scheme as in standard VPIC. Thus, instead of utilizing one bit in the coding process, a minimum of two bits is required to code any image block representing three independent quantization values.

The Three-Level WMME BTC coding of edges is done as follows. Assuming that the transform coefficients of an edge block are sorted from the minimum to the maximum values as shown in Figure 4-21,

Figure 4-21: Transform Coefficients of an Edge Block Sorted from *min* to *max*.

the coefficients are binary coded as indicated in Table 4-12. Any coefficient between T_- and T_+ is coded as "00" and is reconstructed as value zero. Any coefficient between T_+ and *max* is coded as "01" and is reconstructed as the MME BTC coefficient a . Any

coefficient between min and T^- is coded as “10” and is reconstructed as the MME BTC coefficient b .

Encoding Condition	Binary Code	Reconstruction Value
$T^- \leq b_{i,j} \leq T^+$	00	0
$T^+ < b_{i,j} \leq max$	01	a
$min \leq b_{i,j} < T^-$	10	b

Table 4-12: **Three-Level WMME BTC Coding/Decoding Rules for Each Transform Coefficient.**

The parameters T^- and T^+ control the portion of the transform coefficients to be coded as a zero value, a , or b . The parameters min and max represent the minimum and maximum coefficient values in the image block, respectively. The WMME BTC coefficients a and b , not to be confused with the BTC coefficients introduced in the WVPIC/MAE BTC algorithms in section 4.2, represent the average of the transform coefficients that are either greater than or equal to T^+ or less than or equal to T^- . The WMME BTC coefficient a is computed as:

$$a = \text{avg} (+coef_{i,j}) = \frac{\sum_i +coef_{i,j}}{\text{total number of } +coef_{i,j}}, +coef_{i,j} = [T^+, max] \quad 4-15$$

where $+coef_{i,j}$ denotes any transform coefficient that is between T^+ and max . And the WMME BTC coefficient b is computed as:

$$b = \text{avg}(-coef_{i,j}) = \frac{\sum_i -coef_{i,j}}{\text{total number of } -coef_{i,j}}, -coef_{i,j} = [\min, T-] \quad 4-16$$

where $-coef_{i,j}$ denotes any transform coefficient that is strictly between $\min$ and T . Therefore once the WMME BTC coefficients a and b , and the binary codes are received by the decoder, the reconstruction of each edge transform coefficient is done as per Table 4-12.

As with the WVPIC/MAE BTC achitecture, the WMME BTC achitecture is different from standard BTC coding schemes as described in [14], [15], [16], [17], and [20]. The main difference lies in the fact that only the edge blocks are coded in WMME BTC. Indeed, in images with large numbers of uniform blocks, high compression ratios can be achieved because fewer edge image blocks need to be coded.

4.3.1.1 The Three-Level WMME BTC Coder

The block diagram of the Three-Level WMME BTC architecture is illustred in Figure 4-22. Note the simplicity of the 3-Level WMME BTC architecture as oppose to the VPIC and WVPIC/MAE BTC architectures. The front end of the Three-Level WMME BTC architecture is similar to the front end of the WVPIC/MAE BTC achitecture introduced in section 4.2. More specifically, it includes the following three modules:

- a. *block extraction (4 x 4);*
- b. *compute gradient; and*
- c. *uniform/block detection (threshold).*

The operations of these three modules are exactly as in the WVPIC/MAE BTC architectures. The wavelet coefficients are first partitioned into nonoverlapping $n \times n$ coefficient blocks b . Next the gradient magnitude $\nabla b_{i,j}$ for each transform block is computed. As in the VPIC and WVPIC/MAE BTC, the uniform and edge blocks are differentiated by gradient magnitude. Similarly to WVPIC/MAE BTC, uniform blocks are not treated at all. The *blocktype* of each image block is encoded in μ^* and transmitted to the decoder.

Referring to Figure 4-22, the modules located between the *uniform/edge block detection (threshold)* and the *communication channel*, namely modules ❶ to ❹, are responsible for the 3-level coding of every edge block. Let's assume that a given $n \times n$ image block b where $n = 4$, say block 136 from subband 1, has the following wavelet coefficients values:

$$b_{136} = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 & b_7 \\ b_8 & b_9 & b_{10} & b_{11} \\ b_{12} & b_{13} & b_{14} & b_{15} \end{bmatrix}$$

$$= \begin{bmatrix} 20.757492 & 40.956947 & 8.899836 & -35.432484 \\ -10.045372 & -29.880358 & -30.651733 & 18.266260 \\ -3.288338 & -16.224844 & 26.281256 & -6.133520 \\ 12.470741 & 9.701391 & 2.520442 & 3.264223 \end{bmatrix}$$

The 1-D distribution of b_{136} over the interval $[b_0, b_n^2, 1]$ is shown in Fig. 4-23.

Figure 4-23: Wavelet Transform Coefficients Values of a Given $n \times n$ Image Block Where $n = 4$.

The first step consists in computing the *min* and *max* values of each block. This is performed by module ①. The coefficient values b_3 and b_7 are respectively the *min* and the *max* within the above 16 transform coefficients. The second step (module ②) involve the computing of the positive threshold T_+ and the negative threshold T_- . Both thresholds are computed based on a pre-defined threshold parameter t , and are initiated as such:

$$T_- = \min * t \quad 4-17$$

$$T_+ = \max * t \quad 4-18$$

where the pre-defined threshold parameter t dictates certain transform coefficients in the image block that are closest to a zero energy value to be set to zero. Say t is equal to 0.2 (20%), then T_- and T_+ become:

$$T_- = -35.432484 * 0.2 = -7.087$$

and

$$T_+ = 40.956947 * 0.2 = 8.191$$

In module ③, any transform coefficient whose value is between T_- and T_+ is set to zero, and encoded as the binary code “00” in module ⑥. Thus, transform coefficients b_8, b_{11}, b_{14} , and b_{15} in b_{136} are set to zero, and b_{136}^* becomes:

$$b_{136}^* = \begin{bmatrix} 20.757492 & 40.956947 & 8.899836 & -35.432484 \\ -10.045372 & -29.880358 & -30.651733 & 18.266260 \\ 0 & -16.224844 & 26.281256 & 0 \\ 12.470741 & 9.701391 & 0 & 0 \end{bmatrix}$$

The next step involves the computing of the WMME BTC coefficients a and b . This step is handled by module ④, and does it so by using the formula defined by equations 4-15 and 4-15. Thus, the WMME BTC coefficients a and b become:

$$a = 19.619$$

$$b = -24.447$$

At that point, the WMME BTC coefficients a and b are quantized (module ⑤) as btc_a and btc_b respectively. These two quantized quantity are then transmitted to the decoder.

The last step in the coder involves the coding of b_{136}^* according to Table 4-12. The coding process is taken care by module ⑥ and the equivalent codes transmitted for b_{136}^* are shown below:

$$b_{136}^* = \begin{bmatrix} 01 & 01 & 01 & 10 \\ 10 & 10 & 10 & 01 \\ 00 & 10 & 01 & 00 \\ 01 & 01 & 00 & 00 \end{bmatrix}$$

The binary codes shown above are encoded in a bit stream called *bmp*, and transmitted to the decoder. Entropy coding of the binary codes in *bmp*, which should in all probability increase the compression ratios of the images being compressed is not done in this thesis.

4.3.1.2 The Three-Level WMME BTC Decoder

At the decoder stage (Figure 4-22), once the binary codes *bmp* are decoded and the quantized WMME BTC coefficients *btc_a* and *btc_b* are decoded as *a* and *b*, the edge block is reconstructed as per Table 4-12. The reconstructing process is accomplished by module ⑦. The transform coefficient of a binary code “00” becomes zero, “01” becomes *a*, or “10” becomes *b*. For example, the WMME BTC coefficients *a* and *b* are decoded, using a $2^{n_{btc}}$ -level uniform dequantizer (where n_{btc} is some input parameter), as the quantities:

$$a_{decoder} = 20$$

$$b_{decoder} = -24$$

Based on the binary codes transmitted for b_{136} , the image block transform coefficients are reconstructed as:

$$b_{136}^* = \begin{bmatrix} 20 & 20 & 20 & -24 \\ -24 & -24 & -24 & 20 \\ 0 & -24 & 20 & 0 \\ 20 & 20 & 0 & 0 \end{bmatrix}$$

Each reconstructed image block must be positioned appropriately in the reconstructed image. This is accomplished by module ⑧ called the *block type* module. The type of a block being decoded is indicated by the information decoded from the bit stream u^* . A bit “1” means that an edge block is about to be decoded and positioned in the image buffer. A bit “0” means that a uniform block is to be initiated and positioned in the image buffer. A uniform block is initiated by simply setting its transform coefficients to zero, thus a mean of zero. Once the decoding process is finished, the transform coefficients must undergo an inverse wavelet transform process so to map the image from the wavelet domain to the spatial domain for visual examination.

4.3.2 Results of the Three-Level WMME BTC Coding Scheme With 4 x 4 Block Segmentation

Several experimental settings were carried out using the Three-Level WMME BTC coding scheme. The results are summarized at Table 4-13.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	-	26.37	25 : 1	.32	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 10\%$
2	1 - 9	-	26.47	25 : 1	.32	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 30\%$
3	1 - 9	-	26.47	25 : 1	.32	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
4	1 - 9	-	26.38	25 : 1	.32	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 50\%$
5	1 - 9	-	28.98	10 : 1	.8	$\nabla_{\min} = 2$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 20\%$
6	1 - 9	-	29.33	10 : 1	.8	$\nabla_{\min} = 2$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
7	4 - 9	-	29.47	14.7 : 1	.544	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 30\%$
8	4 - 6	7 - 9	30.40	12.36 : 1	.647	$\nabla_{\min} = 2$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
9	4 - 6	7 - 9	31.10	10.29 : 1	.778	$\nabla_{\min} = 1$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
10	4 - 6	7 - 9	31.40	8.48 : 1	.943	$\nabla_{\min} = .5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
11	4 - 6	7 - 9	31.48	7.14 : 1	1.12	$\nabla_{\min} = .25$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
12	4 - 6	7 - 9	28.02	15.33 : 1	.522	$\nabla_{\min} = 10$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$

Table 4-13: Experimental Results of the Three-Level WMME BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena*.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
13	1 - 6	7 - 9	28.37	24.21 : 1	.33	$\nabla_{min} = 2$ $n_{b1c} = 8$ $\nabla_{max} = 20$ $t = 40\%$
14	1 - 6	7 - 9	29.39	16.5 : 1	.49	$\nabla_{min} = 1$ $n_{b1c} = 8$ $\nabla_{max} = 20$ $t = 40\%$
15	1 - 6	7 - 9	29.79	11.8 : 1	.68	$\nabla_{min} = .5$ $n_{b1c} = 8$ $\nabla_{max} = 20$ $t = 40\%$

Table 4-13: Experimental Results of the Three-Level WMME BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena* (con't).

By carefully examining the performances shown in Table 4-13, one can easily see that the Three-Level WMME BTC achitecture outperforms the WVPIC/MAE BTC coding scheme in terms of compression and PSNR. Trials 1 to 7 were carried out with the aim of finding the optimum compression/PSNR settings. With $\nabla_{min} = 5$ in trials 1 to 4, one can conclude that the best PSNR performance for the image *Lena* is when the threshold t parameter is set at 40%. Thus, any local coefficient that has a magnitude between 40% of the *min* and the *max* is coded as zero. The threshold $t = 40\%$ is confirmed further in trails 5 and 6, where better PSNR result is achieved in the latter.

A good PSNR performance of 29.47 dB is achieved in trial 7 for a compression of 14.7. In trials 8 to 11, very good PSNR accomplishments are realized with medium compression ratios. The results produced in trials 8 to 11 surpass the best performances obtained with standard VPIC and the different WVPIC/MAE BTC achitectures. Nowhere in standard VPIC and WVPIC/MAE BTC algorithms has this level of achievement ever been obtained. Also, good PSNR/compression ratios performance are achieved in trials 13 to 15.

Shown at Figure 4-24 is the reconstructed test image *Lena* for trial 14. Although the image at Figure 4-24 does not reflect a high visual fidelity (confirmed by the low PSNR), there are no artifacts present. You may recall that artifact-free images when using WVPIC/MAE BTC were only possible with 2 x 2 block coding. In ternary, artifact-free image coding is now possible with 4 x 4 block segmentation.

Figure 4-24: Reconstructed Test Image *Lena* for Trial 15 Input Parameters (PSNR = 29.79 dB, C = 11.8).

4.3.3 Results of the Three-Level WMME BTC Coding Scheme With 2 x 2 Block Segmentation

In an effort to increase the PSNR performance of the Three-Level WMME BTC achitecture further, the Three-Level WMME BTC coding scheme was modified to operate on 2 x 2 blocks of image coefficients. The achitecture is exactly the same as shown in Figure 4-22 except for the fact that it is now adapted for 2 x 2 image coding processing. The results of the tests carried out are presented in Table 4-14.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	-	31.77	6.1 : 1	1.3	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 50\%$
2	1 - 9	-	31.34	6.1 : 1	1.3	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 20\%$
3	1 - 9	-	34.92	2.2 : 1	3.4	$\nabla_{\min} = 2$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
4	1 - 3	-	35.3	1.04 : 1	7.7	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 30\%$
5	4 - 9	-	34	5.1 : 1	1.6	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 30\%$
6	7 - 9	-	37	2.9 : 1	2.8	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 30\%$
7	4 - 6	7 - 9	31.99	7.4 : 1	1.1	$\nabla_{\min} = 2$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
8	4 - 6	7 - 9	32.17	5.7 : 1	1.4	$\nabla_{\min} = 1$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
9	4 - 6	7 - 9	32.22	4.8 : 1	1.7	$\nabla_{\min} = .5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$

Table 4-14: Experimental Results of the Three-Level WMME BTC Coding Scheme With 2 x 2 Block Segmentation Using Test Image *Lena*.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
10	4 - 6	7 - 9	30.45	12.12 : 1	.66	$\nabla_{\min} = 10$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
11	4 - 6	7 - 9	31.36	10.17 : 1	.787	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 50\%$
12	4 - 6	7 - 9	31.39	10.17 : 1	.787	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
13	1 - 6	7 - 9	31.11	9.1 : 1	.884	$\nabla_{\min} = 2$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$
14	1 - 6	7 - 9	30.05	15.5 : 1	.518	$\nabla_{\min} = 5$ $n_{btc} = 8$ $\nabla_{\max} = 20$ $t = 40\%$

Table 4-14: Experimental Results of the Three-Level WMME BTC Coding Scheme With 2 x 2 Block Segmentation Using Test Image *Lena* (con't).

The performance achieved by the Three-Level WMME BTC coding scheme working with 2 x 2 image blocks is better than when operating on 4 x 4 image blocks in terms of PSNR, but has lower compression. In the first six tests, a PSNR as high as 37 dB was produced with a very low compression of 2.9 to 1. This set of tests was performed to get a feel for the overall performance of the coding algorithm.

In trials 7 to 12, the same kind of tests carried out in the previous sections were performed. The strategy involves subbands one to three to be left untouched, subbands four to six to be processed by WMME BTC, and subbands seven to nine to be discarded. As discussed earlier, this strategy permits the best performance in terms of PSNR/compression. Tests 11 and 12 confirm again that the best threshold t setting for the processing of 2 x 2 image blocks is at 40%. As expected, this setting is the same as in the 4 x 4 image segmentation approach. Figure 4-25 shows the reconstructed test

image *Lena* for trial 12, which has a PSNR of 31.39 dB and a compression of 10.17 to one. No artifacts are apparent, and the visual fidelity is pretty good.

Figure 4-25: Reconstructed Test Image *Lena* for Trial 12 Input Parameters (PSNR = 31.39 dB, C = 10.17).

In trials 7 to 12, as the minimum gradient threshold $\nabla_{\min}$ decreases more edges are detected, hence more coding takes place resulting in higher PSNR performance and lower

compression ratios. As opposed to binary coding schemes presented in this work, the coding of more edge blocks positively contributes to better PSNR results.

In trials 13 and 14, subbands one to six are compressed by the WMME BTC algorithm and subbands seven to nine are discarded in the hope of increasing the compression ratio. A medium/high compression of 15.5 to one was achieved in test 14 resulting in a good PSNR of 30.05 dB.

In this version of Three-level WMME BTC coding, $V_{\min} = 5$ and $t = 40\%$ seem to be the optimum input settings. For identical input settings, higher PSNR and lower compression ratio results are obtained with the 2×2 image segmentation architecture than that of the 4×4 image segmentation architecture. The performances of the Three-Level WMME BTC coding schemes working with 4×4 and 2×2 image blocks, are presented in Figure 4-26.

Figure 4-26: Overall Three-Level WMME BTC Coding Scheme Performance.

4.4 Quaternary Coding

Inherent in standard VPIC and WVPIC/MAE BTC coding schemes is the 1-bit or 2-level moment-preserving quantizer used. With the Three-Level WMME BTC coding technique, a 3-level moment-preserving quantizer is used utilizing a 2-bit code as specified in Table 4-12. The one question that comes to mind is, "would it not be advantageous to use the fourth code in the 2-bit code (code "11" in Table 4-15) for coding additional statistical information?" Presently, the 3-Level WMME BTC architecture does not use code "11" as indicated in Table 4-15.

Encoding Condition	Binary Code	Reconstruction Value
$T- \leq b_{i,j} \leq T+$	00	0
$T+ < b_{i,j} \leq \max$	01	a
$\min \leq b_{i,j} < T-$	10	b
-	11	<i>not used</i>

Table 4-15: Three-Level WMME BTC Coding/Decoding Format.

Hence, using a similar achitecture of a Three-Level WMME BTC coding scheme, a four-level moment-preserving quantizer can be implemented. As opposed to a 3-level quantizer, a 4-level quantizer using no more coding bits than two, should produce better PSNR performances with no change to the compression ratios. A 4-level quantizer should permit the coding of highly busy images with better fidelity.

4.4.1 The Four-Level WMME BTC Coding Scheme

The Four-Level WMME BTC achitecture is identical to the 3-Level structure except in the coding of edge blocks. Once again, let's assume that the transform coefficients of an

edge block are sorted from the minimum to the maximum coefficient values as shown in Figure 4-27.

Figure 4-27: Transform Coefficients of an Edge Block Sorted from *min* to *max*.

Then, the local edge coefficients are coded as per Table 4-16.

Encoding Condition	Binary Code	Reconstruction Value
$0 \leq b_{i,j} \leq T+$	00	$a \cdot t_{decoder}$
$T+ < b_{i,j} \leq max$	01	a
$min \leq b_{i,j} < T-$	10	b
$T- \leq b_{i,j} < 0$	11	$b \cdot t_{decoder}$

Table 4-16: Four-Level WMME BTC Coding/Decoding Format.

The parameters $T+$ and $T-$ are computed similarly to eqn. 4-17 and 4-18 with one main difference. The main difference is that the threshold t used in the 3-Level architecture was only used in the coder stage, whereas in the 4-Level architecture, two independent types of thresholds are used. The first one, called the coder threshold t_{coder} , is strictly used at the coder stage. The threshold t_{coder} is an input parameter set by the operator at the coder

stage and represents the threshold that dictates which portion of the local transform coefficients is to be coded between T_- and T_+ .

The second threshold is called the decoder threshold $t_{decoder}$, and is strictly used at the decoder stage. The reason for this threshold is for the decoder to be more adaptable to the reconstruction process. Indeed, by varying $t_{decoder}$, the PSNR performance of the reconstructed edge block also varies. This is confirmed during the testings of the algorithms in a later section. Thus, the T_+ and T_- parameters for the Four-Level WMME BTC achitecture are computed as followed:

$$T_- = \min * t_{coder} \quad 4- 19$$

$$T_+ = \max * t_{coder} \quad 4- 20$$

Additionally, the WMME BTC coefficients a and b are computed as per equations 4-15 and 4-16.

In all, the coding strategy used here is simply an extention of the strategy used in the 3-level version. It is simply taking advantage of extra coding (code “11”), at no additional coding cost.

4.4.1.1 The Four-Level WMME BTC Coder

The achitecture of the Four-Level WMME BTC algorithm is almost identical to the 3-Level achitecture. The only difference is in the coding structure. Figure 4-28 illustrates the block diagram for the Four-Level WMME BTC achitecture.

In Figure 4-28, the modules that are different from the Three-Level architecture are identified as modules ❶ to ❸. Once the parameters min , max , $T-$, and $T+$ are computed the local transform coefficients whose values are between $T-$ and $T+$ must be identified properly so that the BTC coefficients a and b can be adequately computed. And so, the local transform coefficients whose values are between $T-$ and $T+$ are identified temporary by module ❶. In this research, it was decided that the identified transform coefficients were temporary coded as values 1000 and -1000. That is, any transform coefficients whose value is between 0 and $T+$ gets assigned the quantity +1000, and any transform coefficients whose value is between $T-$ and 0 gets assigned the quantity -1000. This temporary assignment of values 1000 or -1000 simply allows the isolation of the coefficients whose values are between $T-$ and $T+$ so that they are not considered when computing the BTC coefficients a and b .

The next step consists in computing the quantized values of the BTC coefficients a and b . The quantization that takes place is uniform quantization using a $2^{n_{btc}}$ -level quantizer as discussed in [9], [46]. As stated earlier in this thesis, the n_{btc} parameter dictates the number of bits that the BTC coefficients are quantized to.

The next step is the encoding process. In module ❷, the transform coefficients are coded as per Table 4-16. That is, any local coefficient $b_{i,j}$ whose value is:

- a. $0 \leq b_{i,j} \leq T+$ gets coded as code "00";
- b. $T+ < b_{i,j} \leq max$ gets coded as code "01";
- c. $min \leq b_{i,j} < T-$ gets coded as code "10"; and
- d. $T- \leq b_{i,j} < 0$ gets coded as code "11".

These codes are encoded in the bit stream called *bmp* and transmitted to the decoder.

Using b_{136} and threshold $t_{coder} = 50\%$ as an example, then

$$b_{136} = \begin{bmatrix} 20.757492 & 40.956947 & 8.899836 & -35.432484 \\ -10.045372 & -29.880358 & -30.651733 & 18.266260 \\ -3.288338 & -16.224844 & 26.281256 & -6.133520 \\ 12.470741 & 9.701391 & 2.520442 & 3.264223 \end{bmatrix}$$

Based on b_{136} and t_{coder} the *min*, *max*, *T-*, and *T+* quantities are initiated as,

$$min = -35.432484$$

$$max = 40.956947$$

$$T- = min \cdot t_{coder} = -35.432484 \cdot 0.5 = -17.72$$

$$T+ = max \cdot t_{coder} = 40.956947 \cdot 0.5 = 20.49$$

The next step consists in identifying the local coefficients whose values are between *T-* and *T+*. Hence, b_{136} which becomes b_{136}^* is initiated with the following values:

$$b_{136}^* = \begin{bmatrix} 20.757492 & 40.956947 & 1000 & -35.432484 \\ -1000 & -29.880358 & -30.651733 & 1000 \\ -1000 & -1000 & 26.281256 & -1000 \\ 1000 & 1000 & 1000 & 1000 \end{bmatrix}$$

The WMME BTC coefficients a and b become:

$$a = \frac{20.757492 + 40.956947 + 26.281256}{3} = 29.33$$

$$b = \frac{-35.432484 - 29.880358 - 30.651733}{3} = -31.99$$

At that point, the WMME BTC coefficients a and b are quantized uniformly and transmitted to the decoder. The last step in the coder involves the coding of b_{136}^* according to Table 4-16. The equivalent codes transmitted for b_{136}^* are shown below:

$$b_{136}^* = \begin{bmatrix} 01 & 01 & 00 & 10 \\ 11 & 10 & 10 & 00 \\ 11 & 11 & 01 & 11 \\ 00 & 00 & 00 & 00 \end{bmatrix}$$

4.4.1.2 The Four-Level WMME BTC Decoder

The decoding process of each edge block is done exactly as the decoder of the Three-Level WMME BTC architecture. That is, once the binary codes of the bit stream bmp are decoded and quantized WMME BTC coefficients btc_a and btc_b are decoded as a and b , the edge block can be reconstructed as per Table 4-16. In module ③, the local transform coefficients are reconstructed such that a binary code of:

- a. "00" is reconstructed as $a \cdot t_{decoder}$;
- b. "01" is reconstructed as a ;
- c. "10" is reconstructed as b ; and
- d. "11" is reconstructed as $b \cdot t_{decoder}$.

In the example above, the dequantized values of a and b are 29 and -32 respectively. Based on the binary codes transmitted by the coder for $b_{l,36}$ and the threshold $t_{decoder}$, the edge block transform coefficients are reconstructed as:

$$\mathbf{b}_{136}^* = \begin{bmatrix} 29 & 29 & 14.5 & -32 \\ -16 & -32 & -32 & 14.5 \\ -16 & -16 & 29 & -16 \\ 14.5 & 14.5 & 14.5 & 14.5 \end{bmatrix}$$

By varying the value of the threshold $t_{decoder}$, the coefficients corresponding to codes "00" and "11" vary and result in different visual reconstruction quality.

4.4.2 Results of the Four-Level WMME BTC Coding Scheme With 4 x 4 Block Segmentation

Table 4-17 contains the experiment results of the tests carried out using the Four-Level WMME BTC architecture with 4 x 4 block segmentation on test image *Lena*.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	.	24.21	25 : 1	.32	$\nabla_{\min} = 5$ $t_{coder} = 80\%$ $n_{bit} = 8$ $\nabla_{\max} = 20$ $t_{decoder} = 80\%$
2	1 - 9	.	26	25 : 1	.32	$\nabla_{\min} = 5$ $t_{coder} = 50\%$ $n_{bit} = 8$ $\nabla_{\max} = 20$ $t_{decoder} = 50\%$
3	1 - 9	.	26.09	25 : 1	.32	$\nabla_{\min} = 5$ $t_{coder} = 10\%$ $n_{bit} = 8$ $\nabla_{\max} = 20$ $t_{decoder} = 10\%$
4	1 - 9	.	26.38	25 : 1	.32	$\nabla_{\min} = 5$ $t_{coder} = 80\%$ $n_{bit} = 8$ $\nabla_{\max} = 20$ $t_{decoder} = 30\%$

Table 4-17: Experimental Results of the Four-Level WMME BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena*.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
5	1 - 9	-	26.58	25 : 1	.32	$\nabla_{\min} = 5$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 10\%$
6	1 - 9	-	26.64	25 : 1	.32	$\nabla_{\min} = 5$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
7	1 - 9	-	26.54	25 : 1	.32	$\nabla_{\min} = 5$ $t_{\text{coder}} = 30\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 10\%$
8	1 - 9	-	26.59	25 : 1	.32	$\nabla_{\min} = 5$ $t_{\text{coder}} = 60\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
9	1 - 9	-	29.91	9.94 : 1	.805	$\nabla_{\min} = 2$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
10	4 - 6	7 - 9	30.63	12.36 : 1	.647	$\nabla_{\min} = 2$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
11	4 - 6	7 - 9	31.43	10.29 : 1	.778	$\nabla_{\min} = 1$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
12	4 - 6	7 - 9	31.88	7.74 : 1	1.03	$\nabla_{\min} = .25$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
13	1 - 6	7 - 9	29.99	16.5 : 1	.49	$\nabla_{\min} = 1$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
14	1 - 9	-	30.50	13.9 : 1	.58	$\nabla_{\min} = 1$ for subbands 1 to 6 $\nabla_{\min} = 6$ for subbands 7 to 9 $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $n_{\text{btc}} = 8$ $t_{\text{decoder}} = 20\%$

Table 4-17. Experimental Results of the Four-Level WMME BTC Coding Scheme With 4 x 4 Block Segmentation Using Test Image *Lena* (con't).

The first step is to find the best settings for the thresholds t_{coder} and t_{decoder} . To find this information, a total of eight trials were carried out (trials 1 to 8). The optimum setting resulting in the highest PSNR and compression ratio is found to be the settings in trial 5.

The next step consists in varying the minimum threshold $\nabla_{\min}$ and the processing and discarding of various subbands. In trial 9, only the $\nabla_{\min}$ is varied resulting in a PSNR

of 29.91 dB and compression of 9.94 to one. To produce higher PSNR and compression results, the usual subband approach is used once again. That is, the subbands one to three are left untouched, subbands four to six are processed through WMME BTC, and subbands seven to nine are discarded. As seen in trials 10 to 12, the highest PSNR achieved was 31.83 dB with a compression of 7.74 to one. These results are a very good from a visual reconstruction perspective, as shown in Figure 4-29, however the compression is somewhat low. Again here there are no artifacts.

In trial 14 (see Figure 4-30), higher compression ratios is achieved with slightly lower PSNRs by processing different subbands with different $\nabla_{\min}$ settings. The artifacts are very obvious in this image, and are caused by the high quantization of the subbands, especially subbands 1 to 3.

Figure 4-29: **Reconstructed Test Image *Lena* for Trial 12 Input Parameters (PSNR = 31.88 dB, C = 7.74).**

Figure 4-30: Reconstructed Test Image *Lena* for Trial 14 Input Parameters (PSNR = 30.50 dB, C = 13.9).

Comparing these results with the ones obtained using the Three-Level WMME BTC algorithm (see Table 4-13), the results achieved here are slightly better. The PSNR and compression ratio differences achieved using the Four-Level WMME BTC coding scheme over the Three-Level WMME BTC coding scheme are 0.40 dB and 0.6,

respectively. The Four-Level WMME BTC coding scheme tends to preserve the fidelity of the image being coded to a high level as compared to the WVPIC/MAE BTC scheme.

Overall, the level of PSNR and compression ratio performances attained by the Four-Level WMME BTC architecture was not excessively impressive compared to the Three-Level WMME BTC coding scheme, but very impressive compared to the WVPIC/MAE BTC architecture. In Figure 4-31, the performances obtained for the Three and Four-Level WMME BTC coding techniques are presented.

Figure 4-31: Three-Level and Four-Level WMME BTC Performance for 4 x 4 Block Segmentation.

4.4.3 Results of the Four-Level WMME BTC Coding Scheme With 2 x 2 Block Segmentation

Table 4-18 contains the experimental results of the tests carried out using the Four-Level WMME BTC architecture with 2 x 2 block segmentation on test image *Lena*.

Trial	Compressed Subband (only)	Discarded Subband (only)	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 9	-	31.48	6.1 : 1	1.3	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 80\%$ $t_{\text{decoder}} = 50\%$ $n_{\text{bit}} = 8$
2	1 - 9	-	31.76	6.1 : 1	1.3	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 80\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
3	1 - 9	-	31.22	6.1 : 1	1.3	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 80\%$ $t_{\text{decoder}} = 10\%$ $n_{\text{bit}} = 8$
4	1 - 9	-	32.33	6.1 : 1	1.3	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 60\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
5	1 - 9	-	32.39	6.1 : 1	1.3	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
6	1 - 9	-	32.22	6.1 : 1	1.3	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 40\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
7	4 - 6	7 - 9	32.17	7.4 : 1	1.1	$\nabla_{\min} = 2$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
8	4 - 6	7 - 9	32.36	5.7 : 1	1.4	$\nabla_{\min} = 1$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
9	4 - 6	7 - 9	32.42	4.9 : 1	1.7	$\nabla_{\min} = .25$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
10	1 - 6	7 - 9	31.52	9.1 : 1	.884	$\nabla_{\min} = 2$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$
11	1 - 6	7 - 9	30.35	15.5 : 1	.518	$\nabla_{\min} = 5$ $\nabla_{\max} = 20$ $t_{\text{coder}} = 50\%$ $t_{\text{decoder}} = 20\%$ $n_{\text{bit}} = 8$

Table 4-18: Experimental Results of the Four-Level WMME BTC Coding Scheme With 2 x 2 Block Segmentation Using Test Image *Lena*.

The results obtained in trials one to three allows to conclude that the best setting for the threshold t_{decoder} is at 20%. From trials four to six, it is also concluded that the best setting for the threshold t_{coder} is at 50%.

In trials seven to nine, the minimum threshold $\nabla_{\min}$ is varied while processing subbands four to six only with the Four-Level WMME BTC algorithm and discarding

subbands seven to nine. Of course, as $\nabla_{\min}$ is decreased better PSNR is found while the compression ratios decrease. Comparing the results obtained in trials seven and eight of Table 4-14 and Table 4-18, the performances of the PSNR for the Four-Level WMME BTC are at least 0.18 dB better for the same compression ratios. As well, comparing the results obtained in trials 13 and 14 of table 4-14 against trials 10 and 11 of table 4-18, the performances of the PSNR achieved for the Four-Level WMME BTC scheme are at least 0.3 dB better than the performances of the PSNR achieved for the Three-Level WMME BTC scheme. The reconstructed test image of *Lena* for trial 11 is shown at Figure 4-32.

Figure 4-32: Reconstructed Test Image *Lena* for Trial 11 Input Parameters (PSNR = 30.35 dB, C = 15.5).

Overall, although the compression ratios for both the Three-Level and Four-Level WMME BTC coding schemes are the same, the PSNR values obtained by the Four-Level WMME BTC architecture are slightly better. In Figure 4-33, the performances obtained for the Three-Level and Four-Level WMME BTC coding techniques with 2 x 2 block segmentation are presented.

Figure 4-33: **Three-Level and Four-Level WMME BTC Performance for 2 x 2 Block Segmentation.**

4.4.4 Results of the Four-Level WMME BTC Coding Scheme With Variable Block Segmentation

In this section, different coding techniques are combined aiming at increasing the compression ratio. It is known right from the start that the highest PSNR that will be obtained will not surpass the PSNR achieved by the Four-Level WMME BTC coding scheme with 2 x 2 block segmentation. However, the compression ratio should be greater. The results of the different combinations of coding techniques and variable block segmentations are presented in Table 4-19.

Trial	Compressed Subband (only)	Coding Technique Used	PSNR (dB)	C	bpp	WVPIC Parameters
1	1 - 3	WMME BTC (2 x 2)	31.09	11.6 : 1	.688	$\nabla_{\min} = .25$ $t_{\text{coder}} = 50\%$ $n_{\text{btl}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
	4 - 6	WMME BTC (4 x 4)				$\nabla_{\min} = 1$ $t_{\text{coder}} = 50\%$ $n_{\text{btl}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
	7 - 9	discarded				
2	1 - 3	WMME BTC (2 x 2)	29.65	11.8 : 1	.676	$\nabla_{\min} = .25$ $t_{\text{coder}} = 50\%$ $n_{\text{btl}} = 8$ $\nabla_{\max} = 20$ $t_{\text{decoder}} = 20\%$
	4 - 6	WVPIC/MAE BTC (2 x 2)				$\nabla_{\min} = 2$ $n_{\text{btl}} = 8$ $n_p = 4$ $\nabla_{\max} = 80$ $n_x = 8$
	7 - 9	discarded				

Table 4-19: Experimental Results Using Variable Block Segmentation and Coding Techniques Using Test Image *Lena*.

The compression achieved in trial 1 is 2.6 dB better than the one achieved in the previous section. However, the PSNR is a bit lower. The results obtained in trial 2 is not as good as the results produced in trial 11 of Table 4-18. It is just to say that the Four-Level WMME BTC performs better than the WVPIC/MAE BTC scheme.

Given that subband 1 to 3 contain more energy than the other subbands, they must be coded with higher fidelity, such as the 2 x 2 Four-Level WMME BTC coding scheme. Thus in trial 1, subband 1 to 3 are coded with the 2 x 2 Four-Level WMME BTC coding scheme, subband 4 to 6 with the 4 x 4 Four-Level WMME BTC coding scheme and subbands 7 to 9 are discarded. A PSNR of 31.09 dB and compression of 11.6 to 1 is achieved. The reconstructed test image *Lena* for trial 1 is shown at Figure 4-34. Overall, the image has a high visual fidelity with no artifacts.

In trial 2, subbands 4 to 6 are coded with the 2 x 2 WVPIC/MAE BTC coding scheme, and subbands 1 to 3 and subbands 7 to 9 are coded similarly to trial 1. As expected, the PSNR is lower and compression is higher.

Figure 4-34: Reconstructed Test Image *Lena* for Trial 1 Input Parameters (PSNR = 31.09 dB C = 11.6).

In summary, the Four-Level WMME BTC coding scheme outperforms any other coding techniques presented in this thesis. The advantages of the 4-Level WMME BTC are:

- a. high visual image quality;
- b. good compression ratios;
- c. fewer block effects (compared to WVPIC/MAE BTC);
- d. low complexity;
- e. no codebook requirements;
- f. no coding of uniform image blocks; and
- g. artifact-free.

CHAPTER 5

Conclusion

5.1 Summary of Contributions

In this study, a thorough description of binary and multi-level image pattern matching-based data compression architectures applied in a wavelet domain were presented as well as the performance achieved by all. The basic pattern matching algorithm used at the beginning of this thesis was standard VPIC introduced by Chen and Bovik. New image pattern matching coding schemes integrating some basic functionality of the VPIC and BTC architectures in the wavelet domain were proposed and documented. Rate distortion curves were shown to illustrate the performance of the new coding algorithms developed and tested in this work.

5.2 Conclusions

The following are the conclusions obtained through utilizing image pattern matching coding schemes applied in the wavelet domain.

- a. It was proven that the wavelet transform coefficients for a 6-tap Daubechies QMF filter banks tend to have zero means in subbands 1 to 9 for a $M = 3$ level wavelet decomposition. This behavior allows for the capability to discard the mean of any subimage rather than transmitting it;

- b. Like most image coding in the transform domain, the coding of images in the wavelet domain is advantageous because it provides the capability to discard low energy transform subimages easily, thus a higher degree of compression can be achieved. An important question that must be researched further is the selection of the wavelet transform itself which can have a certain level of impact on the visual quality of the image being processed;
- c. It was also found that indeed the wavelet transform do not produce blockiness effects and mosquito noise. Only the coding algorithm using VPIC produces such blockiness effects;
- d. It was found that smaller block sizes have higher image fidelity. In the wavelet domain, variable block image coding can be used to code any subband independently depending on their energy content. Subbands containing a high degree of energy can be finely coded by using smaller block sizes. Subbands containing a low degree of energy can be coarsely coded by using larger block sizes. However, the use of small block partitions is inversely proportional to the compression ratio achievable by the codec;
- e. By the same token, the use of different codecs operating on different subbands is a relatively simple task in the wavelet domain. For example, image coding techniques providing a high degree of compression and medium PSNRs can be well suited for processing high order subbands (i.e. subbands 7 to 9), thus achieving high compression ratios. Image coding schemes producing high PSNRs and medium compression ratios can

3 of/de 3

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

effectively be used to code the lower order subbands, such as subbands 1 to 3, thus obtaining very good visual quality:

- f. The development of the binary-based coding scheme WVPIC/MAE BTC in the wavelet domain using standard VPIC as the basic architecture foundation provides slightly better performance in terms of PSNR and compression ratio than standard VPIC in the spatial domain; However, some noticeable artifacts are present in reconstructed images with PSNRs lower than 29 dB. This is mainly due to coarse quantization of the subband images and the type of wavelet transform used. As stated by Villasenor et *al.*, artifacts in the wavelet domain may also be caused by the type of wavelet transform used which may have a tendency to produce artifacts if the transform coefficients are coarsely quantized [47];
- g. The standard VPIC and WVPIC/MAE BTC coding schemes are excellent image compressors, but overall, can reconstruct somewhat moderate visual quality images only; It is suspected that the local gradients attribute might not have the capacity to accurately preserve the high fidelity requirement to achieve the high degree of visual quality needed for the HVS. This is apparent when the local gradients are transmitted or not, the performance are very similar;
- h. With more and more edge subimages being detected, by lowering the minimum gradient threshold ∇_{min} in standard VPIC and WVPIC/MAE BTC coding schemes, there comes a point where there is no more contribution to

the visual quality of the reconstructed image. It is suspected that this is caused by the way the edges are reconstructed, that is, the product of the oriented gradient coefficients (code vectors) and the local gradient transmitted by the coder. This approach might not allow for a high degree of reconstruction fidelity. Future research needs to be done in this area to further confirm this hypothesis;

- i. The complexity of the standard VPIC and WVPIC/MAE BTC coding architecture in term of operation per pixels is higher than that of standard BTC coding. The complexity of the WVPIC/MAE BTC coding scheme is slightly lower than standard VPIC;
- j. The Three-Level and Four-Level WMME BTC coding schemes do not require any codebook;
- k. The complexity of the WMME BTC is lower than standard BTC because all uniform subimages are not coded;
- l. The coding of the lower resolution subbands (i.e. subband 1 to 3) tends to generate artifacts especially with binary coding;
- m. Multi-level image coding tends to preserve a higher degree of image fidelity;
- n. Image coding using pattern matching in the wavelet domain works better with multi-level coding than with binary coding;
- o. The coding of more edge blocks using the WMME BTC coding schemes tend to increase the PSNR performance;

- p. By not transmitting uniform image blocks, and coding the edge blocks only, allows for images to be decoded with very good visual quality and fidelity, especially when using the WMME BTC coding scheme;

5.3 Future Research

There remains a significant body of work to be done in the development of an optimized pattern matching image coding scheme which could outperform JPEG for example. In terms of complexity, the WMME BTC architecture is quite simple. However, more research must be done to increase its overall performance. The following are some of the topics that need to be carefully examined in the future:

- a. At the early stages of the thesis, we came accross standard VPIC and we thought that it was an outstanding coding technique with very high potential due to its low complexity, small universal codebook and pattern matching orientation. To our surprise, we found out that while VPIC has high compression performance, it has very poor fidelity. It is suspected that its low image fidelity attribute is due to the incapacity for the local gradient magnitudes to preserve a high degree of image fidelity. Using the 64K-Pattern WVPIC/MAE BTC architecture, where all possible patterns are used, supposedly guarantees that all edge blocks detected can be reproduced faithfully. However, results show that the PSNR is still low. More research would be required in examining other moment-preserving quantizing attributes or designing a modified version of the reconstruction process that

takes place when using gradient magnitudes. At this point in time, one must not forget that binary coding can easily outperform multi-level coding in term of compression only;

- b. The effort put in this thesis was more concentrated in the coding of images using pattern matching in the wavelet domain than the selection of a perfect wavelet transform for data compression purposes. However, one should realize that the choice of a wavelet transform is a critical issue that affects image quality as well. According to Villasenor et al., there is a direct correlation between the choice of a wavelet transform and its ability to compress data [47]. Ultimately, the filter banks should be chosen so that the maximum amount of information about the original image is packed into the reference signal (subband 0), and the result of the decomposition/reconstruction leads to artifact-free reconstruction of the image reconstructed from the *reference* signal alone. In addition, it is recommended that more research be put in the effects of coding images with higher levels of wavelet decomposition;
- c. In this work, block partition was focused around 2 x 2 and 4 x 4 block segmentations. Knowledge about the performance of the multi-level WMME BTC algorithm working with larger block sizes may be very helpful. Undoubtedly, coding large subband image regions with uniform distributions would contribute to higher compression ratios.

- d. Another area of research that comes to mind is the effective coding of the binary information transmitted by the WVPIC/MAE BTC and WMME BTC codecs. Like JPEG, the binary information such as the *blocktype*, quantized gradient values, BTC coefficients *btc_a* and *btc_b*, and so on, can be entropy coded to achieve higher compression ratios. The bit maps being transmitted, as *j* in the WVPIC/MAE BTC algorithm or as *bmp* in the WMME BTC algorithm, can be effectively entropy coded and would in all probability contribute to higher compression ratios. This effort would require a second pass in the coding process;
- e. As we have seen in this thesis, multi-level image coding performs better than binary coding in the wavelet domain. Future effort can be well spent in designing more powerful multi-level moment-preserving quantizers that would outperform the performance of the Four-Level WMME BTC coding scheme presented herein. Even five and six-level coding schemes with outstanding quantizers may be the way to go;
- f. In this thesis, floating point numbers were used because no additional cost was required. However, if such coding schemes introduced herein were to be implemented in real-time applications where memory and speed are the most important factors, then one would require that the algorithms be implemented in integer number environment. Therefore, the coding schemes introduced in this work might merit additional efforts to be implemented in an integer

environment while preserving the accuracy offered by the floating point environment;

- g. The edge subimages in the WVPIC/MAE BTC and WMME BTC coding schemes are detected by gradient thresholding. Sometimes the identification of edge subimages can be a difficult task especially in the presence of noise. An important area of research would be to examine the effects of performing some kind of image enhancement before the coding process stage to help in identifying edge subimages in an easier fashion. For example, in the WMME BTC coding architecture, an increase in detected edge subimages has a positive contribution in term of image fidelity.
- h. Future efforts may be put in the design of an algorithm that can automatically compute the total energy in each subband so that low energy subbands can be automatically discarded. This may become handy when employed in higher multiresolution wavelet subbands. As we have seen in this work, the test image *Lena* happens to have less energy content in subband 6 than in subband 7 (see section 4.1.1). Such an algorithm would automatically discard subband 6 instead of 7.

References

- [1] B. V. Dasarathy, *Image Data Compression Block Truncation Coding*, IEEE Computer Society Press, 1995.
- [2] Anil K. Jain, *Fundamentals of Digital Image Processing*, New Jersey: Prentice-Hall, 1989.
- [3] D.H. Hubel and T. N. Wiesel, *Brain Mechanisms of Vision*, Sci. Amer., vol. 241, pp. 150-162, Sept. 1979.
- [4] D. Chen and A. Bovik, *Visual Pattern Image Coding*, IEEE Transactions on Communications, Vol. 38, No. 12, pp. 2137-2146, December 1990.
- [5] M. Rabbani and P.W. Jones, *Digital Image Compression Techniques*, SPIE Press, Washington: 1991.
- [6] I. Kadar and V. Libby, *Signal Processing, Sensor Fusion and Target Recognition IV*, Proceedings SPIE, Vol. 2484, Orlando, Florida, April 1995.
- [7] M. Kunt, A. Ikonomopoulos and M. Kocher, *Second-Generation Image-Coding Techniques*, Proceedings of the IEEE, Vol. 73, No. 4, pp. 549-574, April 1985.
- [8] A. N. Netravali and J. O. Limb, *Picture Coding: A Review*, Proceedings of the IEEE, Vol. 63, pp. 366-406, March 1980.
- [9] A. K. Jain, Proceedings of the IEEE, *Image Data Compression: A Review*, Vol. 69, pp. 349-389, March 1981.
- [10] A. Ikonomopoulos, M. Kocher and M. Kunt, *Image Coding Based on Human Visual Properties for Optimal Reduction of Redundancy*, Proceedings 3rd Scandinavian Conference on Image Analysis, Copenhagen, Denmark, pp. 216-222, July 1983.
- [11] H. Abut, *Vector Quantization*, New York: IEEE Press, 1990.
- [12] N. M. Nasrabadi and R. A. King, *Image Coding Using Vector Quantization: A Review*, IEEE Transactions on Communications, Vol. 36, pp. 957-971, August 1988.
- [13] A. Buzo et al., *Speech Coding Based Upon Vector Quantization*, IEEE Transaction on Acoustics, Speech and Signal Processing, pp. 562-574, October 1980.
- [14] E. J. Delp and O. R. Mitchell, *Image Compression Using BTC*, IEEE Transactions on Communications, Vol. COM-29, pp. 1335-1342, December 1979.

- [15] V. R. Udpikar and J. P. Raina, *BTC Image Coding Using Vector Quantization*, IEEE Transactions on Communications, Vol. COM-35, pp. 352-356, March 1987.
- [16] D. R. Halverson et al., *A Generalized BTC Algorithm for Image Compression*, IEEE Transactions on Acoustics, Speech, Signal Processing, Vol. ASSP-32, pp. 664-668, June 1984.
- [17] M. D. Lema and O. R. Mitchell, *Absolute Moment Block Truncation Coding and Applications to Colour Image*, IEEE Transactions on Communications, Vol. COM-32, pp. 1148-1157, October 1984.
- [18] P. M. Embree and B. Kimble, *C Language Algorithms for Digital Signal Processing*, New Jersey: Prentice Hall, 1991.
- [19] D. C. Coll and Y. Wu, *BTC-VQ-DCT Hybrid Coding of Digital Images*, IEEE Transactions on Communications, Vol. 39, No. 9, September 1991.
- [20] D. C. Coll and Y. Wu, *Multi-Level Block Truncation Coding for High Fidelity Compression of Composite Imagery*, Proceedings of Canadian Conference on Electrical Comput. Engineering, Vancouver, B.C., Canada, 1988.
- [21] John Shoesmith, *Compression Software has Benefits for Users of Wide-Area-Network*, Technology Government, Vol. 2, No. 8, p. 12, August 1995.
- [22] A. N. Akansu and R.A. Haddad, *Multiresolution Signal Decomposition - Transform, Subbands and Wavelets*, Academic Press, 1992.
- [23] M. Ohta, M. Yano and T. Nishitani, *Wavelet Picture Coding with Transform Coding Approach*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E75-A, No. 7, pp. 776-785, July 1992.
- [24] A. Low, *Introductory Computer Vision and Image Processing*, McGraw-Hill, 1991.
- [25] K. R. Rao and P. Yip, *Discrete Cosine Transform - Algorithms, Advantages, Applications*, New York: Academic Press Inc., 1990.
- [26] H.H. Szu, *Wavelet Theory and Applications (Part I and II)*, SPIE, Aerosense 95, 17-21 April 95, Orlando, Florida, 1995.
- [27] A. Rosenfeld, *Multiresolution Image Processing*, New York: Springer-Verlag, 1984.

- [28] S. G. Mallat, *Multifrequency Channel Decompositions of Images and Wavelet Models*, IEEE Transactions on Acoustic, Speech, and Signal Processing, Vol. 37, No. 12, December 1989.
- [29] N. Jayant, J. Johnston, and R. Safranek, *Signal Compression Based on Models of Human Perception*, Proceedings of the IEEE, Vol. 81, No. 10, October 1993.
- [30] C. K. Chui, *An Introduction to Wavelets*, Academic Press, 1992.
- [31] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, *Image Coding Using Wavelet Transform*, IEEE Transactions on Image Processing, Vol. 1, No. 2, pp. 205-220, April 1992.
- [32] I. Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, 1992.
- [33] P. J. Burt and E. H. Adelson, *The Laplacian Pyramid as a Compact Image Code*, IEEE Transactions on Communications, Vol. COM-31, pp. 532-540, April 1983.
- [34] O. Rioul and M. Vetterli, *Wavelets and Signal Processing*, IEEE Signal Processing Magazine 8, pp. 14-38, 1991.
- [35] R. Wilson, A. Calway, and E Pearson, *A Generalized Wavelet Transform for Fourier Analysis: The Multiresolution Fourier Transform and Its Application to Image and Audio Signal Analysis*, IEEE Transactions on Information Theory, Vol. 38, No. 2, pp. 674-690, March 1992.
- [36] A Gorea, *Representations of Vision - Trends and Tacit Assumptions in Vision Research*, New York: Cambridge University Press, 1990.
- [37] W. R. Hendee and P.N.T. Wells, *The Perception of Visual Information*, New York: Springer-Verlag, 1992.
- [38] K. W. Chan and , K. L. Chan, *Variable Block and Multi-Level Extended Visual Pattern Image Coding*, Proceedings of First International Conference on Image Processing (ICIP-94), Vol. 1, pp. 375-379, Nov. 13-16, Texas, 1994.
- [39] C. A. Lindley, *Practical Image Processing in C - Acquisition, Manipulation, Storage*, New York: Wiley, 1991.
- [40] E. O. Brigham, *The Fast Fourier Transform*, New Jersey: Prentice Hall, 1974.
- [41] D.F. Elliott, *Handbook of Digital Signal Processing: Engineering Applications*, an Diego, California: Academic Press, 1987.

- [42] A. V. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*, New Jersey: Prentice Hall, 1989.
- [43] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*, New Jersey: Prentice Hall, 1975.
- [44] M. Ejiri, *Machine Vision - A Practical Technology for Advanced Image Processing*, New York: Gordon and Breach Science Publisher, 1989.
- [45] B. K. P. Horn, *Robot Vision*, New York: McGraw-Hill, 1986.
- [46] W. K. Pratt, *Digital Image Processing*, New York: Wiley, 1978.
- [47] John D. Villasenor, B. Belzer, and J. Liao. *Wavelet Filter Evaluation for Image Compression*, IEEE Transactions on Image Processing, Vol. 4, No. 8, pp. 1053 - 1060, August 1995.
- [48] A. Leon-Gracia, *Probability and Random Processes for Electrical Engineering*, New York: Addison-Wesley Publishing Company, 1994.
- [49] N.S. Jayant and P. Noll, *Digital Coding of Waveforms - Principles and Applications to Speech and Video*, New Jersey: Prentice-Hall, 1984.
- [50] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, New York: McGraw-Hill, Inc., 1991.
- [51] Y. Wu and D. C. Coll, *Multilevel Block Truncation Coding Using a Minimax Error Criterion for High-Fidelity Compression of Digital Images*, IEEE Transactions on Communications, Vol. 41, No. 8, pp. 1179-1191, August 1993.
- [52] M. D. Levine, *Vision in Man and Machine*, New York: McGraw-Hill, 1985.
- [53] N.R. Pal and S.K. Pal, *A Review on Image Segmentation Techniques*, Pattern Recognition, Vol. 26, pp. 1277-1294, August 1993.

Appendix A

Source Code

As mentioned at the beginning of this thesis, approximately 99,200 lines of code were generated in C and C++ languages to implement and test all of the various coding architectures introduced in this work. It would require too much documentation to list all of the source code developed during this work. Instead, a description of all source codes is included herein. First, all UNIX platform targeted algorithms are tabulated at Table A-1.

Program	lines of code	Description
subband	3382	This C language program converts images from the spatial domain to the wavelet domain, and was not developed as part of this thesis. Available through public domain, this program was developed and tested by Stanford University targeted for UNIX platforms. command: subband <infile.unx > -g 512 512 -f <filterfile> -i >! <outfile.raw>
displ_byte	400	This program was developed to allow the display of <i>raw</i> images on UNIX machines. Images must be squared in term of dimension and can be of any lenght. Third party primitives are also used in as part of this program. Primitives are called using the C subroutine "system". command: displ_byte -i <infile.raw> -o <outfile.unx >
displ_int	404	This program is similar to the displ_byte program. It is used to display images that are composed of integer instead of bytes. command: displ_int -i <infile.unx > -o <outfile.unx >

Table A-1: Thesis Source Codes Description for UNIX-Based Programs.

Program	lines of code	Description
displ_float	421	This program is similar to the displ_byte program. It is used to display images that are composed of floating points instead of bytes. The main purpose of this program was to display the transform coefficients subband images, which are in floating point format. command: displ_float -i <infile.unx > -o <outfile.unx >
diff_ftt	391	This program was developed to compute the difference between two floating point images and save the results into float format. Again, both images must be squared images. command: diff_ftt -i <infile1.unx > -o <infile2.unx > -d 512

Table A-1: Thesis Source Codes Description for UNIX-Based Programs (con't).

Second, all DOS platform targeted algorithms are described in the following

Tables:

- a. Table A-2: Digital Image Processing Utilities;
- b. Table A-3: Digital Image Statistical Algorithms;
- c. Table A-4: Digital Image Codebook/Pattern Generation Algorithms; and
- d. Table A-5: Digital Image Coding Algorithms.

The algorithms presented below are some of the main programs that are described in this work. Having said that, there are lots of other algorithms that are not presented here because some of them served as tests programs, and others, are simply intermediate versions of the final or main program presented below. The references used in developing all the source code were based on [18], [39], [52] to [60] inclusively.

UTILITIES		
Program	lines of code	Description
unx2dos.exe	372	<p>This program was written and developed to have the capability of converting integer/floating point UNIX-based image files into DOS format. Since the wavelet transform coefficients for a given image are provided by the subband program residing on a UNIX platform, and that the compression algorithms are developed on IBM/PC platform, then this capability is indeed a requirement.</p> <p>command: unx2dos -i <infile.unx> -o <outfile.dos> -d 512</p>
dos2unx.exe	249	<p>This program was written and developed to have the capability of converting integer/floating point DOS-based image files into UNIX format. This capability is required so that the transform coefficients for a given image can be transformed from the wavelet domain back to the spatial domain by using the inverse wavelet transform using the subband program residing on UNIX platform.</p> <p>command: dos2unx -i <infile.dos> -o <outfile.unx> -d 512</p>
int2byte.exe	505	<p>This program was developed to have the capability of converting integer-based images into <i>raw</i> (byte) format. This capability was required for displaying images to the monitor using photoshop or other third party imaging programs. Most third party image imaging program will only display standardized image format such as <i>tiff</i>, <i>pcx</i>, <i>bmp</i>, etc... Photoshop is one of the rare imaging program that can display <i>raw</i>-based images.</p> <p>command: int2byte -i <infile.dos> -o <outfile.raw> -d 512</p>
flt2raw.exe	453	<p>This program was developed to have the capability of converting floating point-based images into <i>raw</i> format for displaying purposes as discussed above. The images involved here are the wavelet subband images that are composed of floating points transform coefficients.</p> <p>command: flt2raw -i <infile.dos> -o <outfile.raw> -d 512</p>
band2raw.exe	430	<p>This program was developed to have the capability of converting individual wavelet subbands, which are inherently composed of floating point transform coefficients, into <i>raw</i> format for displaying purposes.</p> <p>command: band2raw -i <subbandfile.flt> -o <subbandfile.raw> -d 512</p>
pyr2byte.exe	496	<p>This program was developed to have the capability of converting floating point pyramidal transform coefficients image files into <i>raw</i>-based image file for display purposes.</p> <p>command: pyr2byte -i <pyramidfile.flt> -o <pyramidfile.raw> -d 512</p>

Table A-2: Digital Image Processing Utilities for DOS-Based Programs.

UTILITIES		
Program	lines of code	Description
band2fil.exe	735	<p>This program was developed to have the capability of decomposing the 10 subbands from a $M = 3$ wavelet pyramidal image file, composed of transform coefficients, and saves each individual subbands into 10 separate subband files. The internal information contained in each subband is 100% left untouched. This capability was required to allow the compression algorithms developed in this thesis to compress each subband completely separately. And so, subband 0 from the pyramidal file (containing all of the subbands) is saved in a file called <i>band0.flr</i>. Subband 1 is saved as <i>band1.flr</i>, and so on up to <i>band9.flr</i>. In a later optimizing stage, this program can be incorporated in the front end of any compression algorithms so that the pyramidal wavelet file can be inputted directly into the coder.</p> <p>command: band2fil -i <pyramidfile.flr> -d 512</p>
fil2band.exe	646	<p>This program was developed to have the capability to convert each floating point transform coefficient subband, once decoded, into a pyramidal-based integer format so to be compatible with subband just prior to perform the inverse wavelet transform.</p> <p>command: fil2band -o <pyramidfile.int> -d 512</p>
recomb.exe	642	<p>This <i>recombine</i> program has the same purpose as the fil2band program except that the pyramidal file is saved in floating point format rather than integer format. This capability was provided so that pyramidal floating point transform coefficient image files can be used by subband with the <i>float</i> option turned on. We realized later that the subband program does not perform well with this option.</p> <p>command: recomb -o <pyramidfile.flr> -d 512</p>
recomb1.exe	681	<p>This second <i>recombine</i> program was developed to have a similar capability to the recomb program. Although it recombines all floating point subband files into pyramidal format, it has the additional capability to scale each transform coefficient, then saves all of the information in integer format. This is the algorithm to be used for the subband program once the decoded process is performed.</p> <p>command: recomb1 -o <pyramidfile.int> -d 512</p>

Table A-2 Digital Image Processing Utilities for DOS Environment (con't).

UTILITIES		
Program	lines of code	Description
recomb2.exe	709	<p>This third <i>recombine</i> program was developed to have a similar capability to the recomb2 program. This program is actually the same as recomb1 except that subband 0 only is not scaled. Results showed that this program does not work properly with subband compatibility.</p> <p>command: recomb2 -o <pyramidfile.int> -d 512</p> <p>Note: One must realize that these algorithms (<i>fil2band</i>, <i>recomb</i>, <i>recomb1</i>, <i>recomb2</i>) were developed for testing purposes because we did not have any information about the specifications and requirements of the subband program. In fact there was no documentation available anywhere that could have enlightened us about subband. Although the subband source code is somewhat documented, there is no clue anywhere about its specifications, type of file format inputted, output spec, etc... This is why the above algorithms were produce because we had to learn about the requirements of subband by trial and error</p>
zero_out.exe	216	<p>This program was created to have the capability to initiate subband images with zeros. This is used when one wants to discard a particular subband.</p> <p>command: zero_out -o <subband.flit> -d 256</p>

Table A-2: Digital Image Processing Utilities for DOS Environment (con't).

STATISTICAL ALGORITHMS		
Program	lines of code	Description
stat.exe	941	<p>This algorithm was developed to perform statistical subband analysis on $M = 3$ wavelet decomposition images. The statistical information computed are statistical data about the original image in the spatial domain and about each wavelet transformed subband. The statistical information are:</p> <ul style="list-style-type: none"> a. dynamic range of each pixel and transform coefficient; b. mean; c. variance; and d. second moment. <p>command: stat</p>
stat2.exe	901	<p>This algorithm is similar to the stat algorithm except that it performs a statistical analysis on the reconstructed subbands only. This means that a statistical analysis of each subband is performed only after the image has gone through the coding and decoding process.</p> <p>command: stat2</p>
img_anal.exe	1613	<p>This program was developed to have the capability to perform pattern analysis on images in the spatial domain. The information computed are:</p> <ul style="list-style-type: none"> a. number of 4 x 4 blocks; b. dimension of the image; c. dynamic ranges of the local gradient values; d. minimum gradient thresholds ∇_{min} used; e. number of uniform blocks detected based on ∇_{min}; f. number of edge blocks detected based on ∇_{min}; and g. the first xth most popular subimages detected. <p>command: img_anal -i <imagefile.raw> -d 512 -m < minimum threshold ∇_{min}> -p <first xth most popular pattern detected></p>
var.exe	852	<p>This program has been created to compute the variance of the means for each subband images. All this in the hope of determine if the mean of any uniform and edge block is zero. This information was needed in determining the possibility of not transmitting the mean of any block to the decoder in the WVPL, MAE BTC and WMME BTC coding schemes.</p> <p>command: var -a <initial subband to be computed> -b < last subband to be computed></p>

Table A-3: Digital Image Statistical Algorithms for DOS Environment.

STATISTICAL ALGORITHMS		
Program	lines of code	Description
analysis.exe	1625	<p>This program was developed to have the capability to perform some detailed statistical analysis on $M = 3$ wavelet decomposition pyramidal image files with 4×4 block segmentation. The purpose of this analysis was to have a better idea about the statistical behavior of wavelet subbands with 4×4 partitions. The information computed are:</p> <ul style="list-style-type: none"> a. number of 4×4 blocks per subband; b. dimension of each subband; c. dynamic ranges of the local gradients in each subband; d. dynamic ranges of the local means in each subband; e. minimum gradient thresholds ∇_{min} used; f. number of uniform blocks detected based on ∇_{min}; and g. number of edge blocks detected based on ∇_{min}. h. the first xth most popular subimages detected in each subband. <p>command: analysis -i <subbandfile.flr> -d 512 -m < minimum threshold ∇_{min}> -p <first xth most popular pattern detected></p> <p>This program must be executed for each subband. All of the subbands must be in separate files by using the band2fil algorithm.</p>
2x2anal.exe	1030	<p>This program is the same as analysis except for the fact that it produces a detailed statistical analysis on $M = 3$ wavelet decomposition pyramidal image files with 2×2 block segmentation.</p> <p>command: 2x2anal -i <subbandfile.flr> -d 512 -m < minimum threshold ∇_{min}> -p <first xth most popular pattern detected></p>
mse.exe	387	<p>This program has been created to compute various statistical information between the original image and reconstructed image. These statistical info are:</p> <ul style="list-style-type: none"> a. MSE; b. RMS; c. PSNR; d. mean of the original image; e. mean of the reconstructed image; f. variance of the original image; g. variance of the reconstructed image; h. second order moment of the original image; i. second order moment of the reconstructed image; <p>command: mse -i <imagefile1.raw> -b < imagefile1 raw > -d 512</p>

Table A-3: Digital Image Statistical Algorithms for DOS Environment (con't).

STATISTICAL ALGORITHMS		
Program	lines of code	Description
energy.exe	440	<p>This algorithm performs an energy analysis on the reconstructed subbands. This means that it computes the total energy in each subband in order for the user to have a better idea of what level of energy is contained in each subband.</p> <p>command: energy</p>
hist.exe	577	<p>This program has been created to compute an histogram of images in the spatial domain. The reason for was to figure out the type of patterns most popular. This kind of information is welcome when designing a codebook. In addition to building an histogram, the program would display the first <i>m</i>th most popular patterns, indicated by the user, in the image and the number of time detected.</p> <p>command: hist -i <infile.raw> -b < histfile.raw > -d <first <i>m</i>th most popular patterns></p>
diff.exe	386	<p>This program was developed to compute the difference between two <i>raw</i>-based images and save the result into <i>raw</i> format. Both images must be squared images.</p> <p>command: diff -i <infile1.raw > -o < infile2.raw > -d 512</p>
diffflt.exe	418	<p>This program was developed to compute the difference between two <i>floating point</i>-based images and save the result into <i>floating point</i> format. Both images must be squared images.</p> <p>command: diffflt -i <infile1.flt > -o < infile2.flt > -d 512</p>

Table A-3: Digital Image Statistical Algorithms for DOS Environment (con't).

CODEBOOK/PATTERN GENERATION ALGORITHMS		
Program	lines of code	Description
codebook.exe	445	This program was developed to generate codebook oriented gradient coefficients for each code vector for all possible 65536 patterns in a 64K-Pattern codebook. The codebook generated file is saved in <i>codebook.vpc</i> . This 64K-Pattern codebook is used with the WVPIC/MAE BTC coding architecture. command: codebook
genpat.exe	436	This program was created to compute the oriented gradient coefficients for a 4 x 4 inputted pattern described in hexadecimal format. command: genpat -p <pattern ex.: genpat -p 8421>
rcbk4x4.exe	472	This program has a similar purpose to that of codebook . This program was designed to generate a 256-Pattern codebook for the 256-Pattern WVPIC/MAE BTC algorithm. command: rcbk4x4
rcbk28.exe	472	This program has a similar purpose to that of rcbk4x4 . This program was designed to generate a 28-Pattern codebook for the 28-Pattern WVPIC/MAE BTC algorithm. command: rcbk28
cbook2x2.exe	351	This program has a similar purpose to that of rcbk4x4 except that it is dedicated to generates the oriented gradient coefficients for all possible 16 patterns that can exist in 2 x 2 patterns (4-bit patterns). This program was designed to generate a 16-Pattern codebook for the 16-Pattern WVPIC/MAE BTC algorithm introduced in this work. command: cbook2x2
cbk2x2.exe	338	This program was created to compute the oriented gradient coefficients for a 2 x 2 inputted pattern described in hexadecimal format. command: cbk2x2 -p <pattern ex.: cbk2x2 -p e>

Table A-4: Digital Image Codebook/Pattern Generation Algorithms for DOS Environment.

IMAGE CODING ALGORITHMS			
Main Codec	Program	lines of code	Description
VPIC 4 x 4 Partition	vpic.exe	2982	<p>This program is the coder part of the VPIC algorithm developed to code images in the spatial domain. A detailed description of this program is presented in Chapter 3 of this thesis. The images to be processed must be squared.</p> <p>command: vpic -i <infile.raw> -o <histfile.raw> -d 512 -m <minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -u <no. of bits per uniform local mean parameter> -e <no. of bits per edge local mean parameter> -p <no. of bits per pattern address parameter> -g <no. of bits per edge local gradient parameter></p>
	dvpic.exe	1698	<p>This algorithm is the decoder part of the VPIC architecture described in Chapter 3.</p> <p>command: dvpic -o <outfile.raw> -d 512 -m <minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -u <no. of bits per uniform local mean parameter> -e <no. of bits per edge local mean parameter> -p <no. of bits per pattern address parameter> -g <no. of bits per edge local gradient parameter> -f <postfiltering parameter></p>
64K- Pattern WVPIC/ MAE BTC 4 x 4 Partition	wvpicb4.exe	3058	<p>This program is the coder part of the 64K-Pattern WVPIC/MAE BTC architecture. The design of its architecture is detailed in Chapter 4.</p> <p>command: wvpicb4 -m <minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -u <no. of bits per uniform local mean parameter> -e <no. of bits per edge local mean parameter> -c <no. of bits per BTC coefficient parameter> -p <no. of bits per pattern address parameter> -g <no. of bits per edge local gradient parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
	dwvpicb4.exe	1986	<p>This program is the decoder portion of the 64K-Pattern WVPIC/MAE BTC architecture</p> <p>command: dwvpicb4 -m <minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -u <no. of bits per uniform local mean parameter> -e <no. of bits per edge local mean parameter> -c <no. of bits per BTC coefficient parameter> -p <no. of bits per pattern address parameter> -g <no. of bits per edge local gradient parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>

Table A-5: Digital Image Coding Algorithms for DOS Environment.

IMAGE CODING ALGORITHMS			
Main Codec	Program	lines of code	Description
Modified 64K-Pattern WVPIC/MAE BTC 4 x 4 Partition	wvpicb4.exe	3012	<p>This program is the coder part of the Modified 64K-Pattern WVPIC/MAE BTC architecture detailed in Chapter 4.</p> <p>command: wvpicb4 -m < minimum threshold V_{min}> -n < maximum threshold V_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>
	dwvpicb4.exe	1895	<p>This program is the decoder portion of the Modified 64K-Pattern WVPIC/MAE BTC architecture.</p> <p>command: dwvpicb4 -m < minimum threshold V_{min}> -n < maximum threshold V_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>
256-Pattern WVPIC/MAE BTC 4 x 4 Partition	wvpicb4r.exe	2910	<p>This program is the coder part of the 256-Pattern WVPIC/MAE BTC architecture detailed in Chapter 4. Will accept $M = 3$ wavelet decomposed subband images only</p> <p>command: wvpicb4r -m < minimum threshold V_{min}> -n < maximum threshold V_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>
	dwvpicb4r.exe	1767	<p>This program is the decoder portion of the 256-Pattern WVPIC/MAE BTC architecture</p> <p>command: dwvpicb4r -m < minimum threshold V_{min}> -n < maximum threshold V_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>

Table A-5: Digital Image Coding Algorithms for DOS Environment (con't).

IMAGE CODING ALGORITHMS			
Main Codec	Program	lines of code	Description
28- Pattern WVPIC/ MAE BTC 4 x 4 Partition	wvpic28.exe	3011	<p>This program is the coder part of the 28-Pattern WVPIC/MAE BTC architecture detailed in Chapter 4. Will accept $M = 3$ wavelet decomposed subband images only.</p> <p>command: wvpic28 -m < minimum threshold ∇_{min}> -n < maximum threshold ∇_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>
	dwvpic28.exe	1895	<p>This program is the decoder portion of the 28-Pattern WVPIC/MAE BTC architecture.</p> <p>command: dwvpic28 -m < minimum threshold ∇_{min}> -n < maximum threshold ∇_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>
16- Pattern WVPIC/ MAE BTC 2 x 2 Partition	wvpicb2.exe	2949	<p>This program is the coder part of the 16-Pattern WVPIC/MAE BTC architecture detailed in Chapter 4. Will accept $M = 3$ wavelet decomposed subband images only.</p> <p>command: wvpicb2 -m < minimum threshold ∇_{min}> -n < maximum threshold ∇_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>
	dwvpicb2.exe	1928	<p>This program is the decoder portion of the 16-Pattern WVPIC/MAE BTC architecture.</p> <p>command: dwvpicb2 -m < minimum threshold ∇_{min}> -n < maximum threshold ∇_{max}> -u < no. of bits per uniform local mean parameter> -e < no. of bits per edge local mean parameter> -c < no. of bits per BTC coefficient parameter> -p < no. of bits per pattern address parameter> -g < no. of bits per edge local gradient parameter> -a < initial subband to be processed> -b < last subband to be processed ></p>

Table A-5: Digital Image Coding Algorithms for DOS Environment (con't).

IMAGE CODING ALGORITHMS			
Main Codec	Program	lines of code	Description
Three-Level WMME BTC 4 x 4 Partition	wvpict4.exe	3161	<p>This program is the coder part of the Three-Level WMME BTC architecture detailed in Chapter 4 with 4 x 4 block segmentation. Will accept $M = 3$ wavelet decomposed subband images only.</p> <p>command: wvpict4 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -t <threshold parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
	dwpict4.exe	1964	<p>This program is the decoder portion of the Three-Level WMME BTC architecture with 4 x 4 block segmentation.</p> <p>command: dwpict4 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
Three-Level WMME BTC 2 x 2 Partition	wvpict2.exe	3000	<p>This program is the coder part of the Three-Level WMME BTC architecture detailed in Chapter 4 with 2 x 2 block segmentation. Will accept $M = 3$ wavelet decomposed subband images only.</p> <p>command: wvpict2 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -t <threshold parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
	dwpict2.exe	1879	<p>This program is the decoder portion of the Three-Level WMME BTC architecture with 2 x 2 block segmentation.</p> <p>command: dwpict2 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>

Table A-5: Digital Image Coding Algorithms for DOS Environment (con't).

IMAGE CODING ALGORITHMS			
Main Codec	Program	lines of code	Description
Four-Level WMME BTC 4 x 4 Partition	wvpicq4.exe	3042	<p>This program is the coder part of the Four-Level WMME BTC architecture detailed in Chapter 4 with 4 x 4 block segmentation. Will accept $M = 3$ wavelet decomposed subband images only.</p> <p>command: wvpicq4 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -t <threshold coder parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
	dwpicq4.exe	1926	<p>This program is the decoder portion of the Four-Level WMME BTC architecture with 4 x 4 block segmentation.</p> <p>command: dwpicq4 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -t <threshold decoder parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
Four-Level WMME BTC 2 x 2 Partition	wvpicq2.exe	3057	<p>This program is the coder part of the Four-Level WMME BTC architecture detailed in Chapter 4 with 2 x 2 block segmentation. Will accept $M = 3$ wavelet decomposed subband images only.</p> <p>command: wvpicq2 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -t <threshold coder parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>
	dwpicq2.exe	1906	<p>This program is the decoder portion of the Four-Level WMME BTC architecture with 2 x 2 block segmentation.</p> <p>command: dwpicq2 -m < minimum threshold ∇_{min}> -n <maximum threshold ∇_{max}> -c <no. of bits per BTC coefficient parameter> -t <threshold decoder parameter> -a <initial subband to be processed> -b <last subband to be processed ></p>

Table A-5: Digital Image Coding Algorithms for DOS Environment (con't).

Appendix B

Compression/Decompression Process

In this thesis, the process of image coding is done using two computer platforms, namely an Unix station and DOS system. At Figure A-1, a block diagram of the coding and decoding processes is shown.

Figure A-1 - Schematic Diagram for Compression/Decompression Processes Used in this Thesis.

Examining Figure A-1, readers may realize that the wavelet transform algorithm is implemented on the Unix station and the coding/decoding algorithms are implemented on IBM/PC station.

The compression/decompression file convention shown below helps identifying the type of files that is produced after each module shown in Figure A-1.

pyrF.unx

file extension - .*dos* = DOS file
 - .*unx* = UNIX file

 F = floating point-based file
 I = integer-based file

name of the file

Appendix C

Source Code Example

In this thesis, all of the source code generated has been documented. The following example of the source code is typical of the documentation (shown below is a portion of the WMME BTC with 2 x 2 block partitions).

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C

Updated 13:49 19 Aug 95;

page 1

Printed 13:57

```

1 //*****
2 //
3 //          Four-Level Wavelet Minimum Maximum Error
4 //          Block Truncation Coding
5 //
6 //
7 //          (4-Level WMME BTC)
8 //
9 //
10 //          WVPICQ2 - CODER
11 //
12 // This image coding technique uses two types of threshold, namely t_coder and
13 // t_decoder. The threshold t_coder is used in the coder stage, and the threshold
14 // t_decoder is used in the decoder stage. It employs a four-level moment-preserving
15 // quantizer allowing a relatively high PSNR and compression ratio results. In this
16 // algorithm, no residual and mean is used or transmitted. This version operates on
17 // 2 x 2 block segmentation.
18 //
19 // This program was written by Mr.Luc Boileau from the Carleton University, Ottawa,
20 // Canada.
21 //
22 //
23 // Copyright 1995, Carleton University. All rights reserved.
24 //*****
25
26 // include files
27
28 #include <stdio.h>
29 #include <stdlib.h>
30 #include <conio.h>
31 #include <dos.h>
32 #include <math.h>
33 #include "wvpicq2.h"
34
35 // Procedures and Fonctions Prototype
36
37 void init_some_parameters ();
38 void CommandLine ();
39 void ShowCommandLine();

```


C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
Updated 13:49 19 Aug 95;

page 2
Printed 13:57

```

40 void open_image_file ();
41 void read_float_image ();
42 void write_byte_image ();
43 void write_long_image ();
44 void write_int_image ();
45 void alloc_byte ();
46 void alloc_long ();
47 void alloc_float ();
48 void alloc_int ();
49 void byte_2_float ();
50 void float_2_byte ();
51 void bloc_extraction ();
52 void blocimage_2_image ();
53 void mean_intensity_extr ();
54 void residual_bloc_extr ();
55 void reconstruct_subimage ();
56 void create_image_file ();
57 void compute_gradient_data ();
58 float compute_orientation ();
59 void uniform_edge_block_detection ();
60 void compute_min_and_max ();
61 void generate_bmp ();
62 void compute_ab ();
63 void encode_bmp ();
64 int edge_uniform_quantization_level ();
65 int btc_quantization_level ();
66 int gm_quantization_level ();
67 void update_bit_flag ();
68 void encode_image_with_nuu_nue_ng_bits ();
69 void encode_bit_stream ();
70 void pattern_matching ();
71 WORD get_block_and_convert_2_bmp ();
72 WORD find_pattern ();
73 int number_of_ones ();
74 void encode_j_with_np_bits ();
75 void compute_compression_ratio ();
76 void show_results ();
77 void error();
78 void error2 ();
79
80
81 //*****
82 //
83 //          Main Program
84 //
85 //*****
86
87 main(int argc, char *argv[] )
88 {
89

```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
 Updated 13:49 19 Aug 95;

page 3
 Printed 13:57

90 // Variables Declaration

```

91
92 FILE *fp_image;
93 FILE *fp_reco;
94 FILE *fp_b;
95 FILE *fp_ustar;
96 FILE *fp_bmp;
97 FILE *fp_btc_a;
98 FILE *fp_btc_b;
99 FILE *fp_hist;
100 byte *u_star;
101 byte *bmp;
102 byte *btc_a;
103 byte *btc_b;
104 float *float_image;
105 float *float_reco;
106 float *b;
107 float *u;
108 float *r;
109 float *gm;
110 float *o;
111 int *hist;
112 int nrow;
113 int ncol;
114 int img_size;
115 int image_dim = 0;
116 float min_grad = 1.0;
117 float max_grad = 20.0;
118 int Nu = 0;
119 int Ne = 0;
120 int nuu = 5;
121 int nbtc = 8;
122 int np = 3;
123 float t = 0.2;
124 int bloc_rowsize;
125 int bloc_size;
126 int total_no_bloc;
127 int u_star_no_bytes;
128 int g_no_bytes;
129 int bmp_no_bytes;
130 int btc_no_bytes;
131 int u_star_no_encoded_bits;
132 int g_no_encoded_bits;
133 int bmp_no_encoded_bits;
134 int btc_no_encoded_bits;
135 int i;
136 float compression_ratio;
137 int subband;
138 int initial_subband;
139 int subband_no;
140 char subband_filename[10] = "band1.flr";

```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C

Updated 13:49 19 Aug 95;

page 4

Printed 13:57

```

141 char u_filename[12]      = "u_star1.raw";
142 char bmp_filename[9]     = "bmp1.raw";
143 char btc_a_filename[11]  = "btc_a1.raw";
144 char btc_b_filename[11]  = "btc_b1.raw";
145 int  bmp_max_no_bytes    = 65536;
146 int  start_band          = 1;
147 int  end_band            = 9;
148 int  no_of_pattern       = 0;
149 BOOL histogram_flag      = FALSE;
150
151 int  m, n, k, block;
152
153
154 // Start of main program
155 create_image_file (&fp_reco, "dummy.raw");           // due to compiler bug, this must be
done
156
157 CommandLine (&min_grad, &max_grad, &nbtc, &t, &start_band, &end_band, &histogram_flag, argc,
argv; // Get command line arguments
158
159 // initialization of initial parameters
160 nrow = 256;
161 ncol = 256;
162 img_size = nrow * ncol; // compute image size
163 bloc_rowsize = bloc_dim; // initialize size of bloc row
164 bloc_size = bloc_rowsize * bloc_rowsize; // compute size of image bloc
165 total_no_bloc = img_size / bloc_size; // compute total no. of bloc in image
166
167 alloc_float (&float_image, img_size, "band"); // Allocate memory to store image in float format
168 alloc_float (&b, img_size, "band_b"); // Allocate memory to store image in bloc format
169 alloc_float (&gm, total_no_bloc, "band_gm"); // Allocate memory to store gradient magnitudes
170 alloc_float (&o, total_no_bloc, "band_o"); // Allocate memory to store gradient orientations
171 alloc_byte (&u_star, img_size, "band_u*"); // Allocate memory to store quantized u* for TX
172 alloc_byte (&bmp, bmp_max_no_bytes, "band_bmp"); // Allocate memory to store quantized
bitmap for TX
173 alloc_byte (&btc_a, total_no_bloc, "band_btc_a"); // Allocate memory to store quantized btc
"a's" for TX
174 alloc_byte (&btc_b, total_no_bloc, "band_btc_b"); // Allocate memory to store quantized btc
"b's" for TX
175 alloc_int (&hist, 256, "hist.int");
176
177 bloc_rowsize = bloc_dim; // initialize size of bloc row
178 bloc_size = bloc_rowsize * bloc_rowsize; // compute size of image bloc
179 initial_subband = start_band;
180 subband_no = end_band;
181

```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
Updated 13:49 19 Aug 95;

page 5
Printed 13:57

```

182 for (i=0; i < 256; i++)                // initiates hist to 0
183   hist[i] = 0;
184
185 for (subband = initial_subband; subband <= subband_no; subband++)
186 {
187   init_some_parameters ( subband, &nrow, &ncol, subband_filename, u_filename, bmp_filename,
188                           btc_a_filename,
189                           btc_b_filename );
188
189   img_size = nrow * ncol;                // compute subband image size
190   total_no_bloc = img_size / bloc_size;  // compute total no. of bloc in subband
191
192   open_image_file (&fp_image, subband_filename);
193   read_float_image ( fp_image, float_image, img_size );    // read subband file into memory
194   fclose (fp_image);
195
196   bloc_extraction (b, float_image, total_no_bloc, nrow, bloc_rowsize);    // block extraction
197   compute_gradient_data (b, gm, o, bloc_size, bloc_rowsize, total_no_bloc);
198
199
200   Nu = 0;                                // init number of uniform blocks to "0"
201   Ne = 0;                                // init number of edge blocks to "0"
202
203   for (i=0; i<28; i++)
204     pattern_count[i] = 0;
205
206   uniform_edge_block_detection (b, gm, u_star, bmp, btc_a, btc_b, min_grad, max_grad, nbtc, t,
207                                 img_size, bloc_size, bloc_rowsize, total_no_bloc, &Nu, &Ne, &pattern_count[0], bmp_max_no_bytes,
208                                 hist);
209
210   compute_compression_ratio (    &u_star_no_bytes,    &bmp_no_bytes,    &btc_no_bytes,
211                                 &u_star_no_encoded_bits, &btc_no_encoded_bits, &compression_ratio, Nu, Ne, nbtc, t, img_size );
212
213
214   create_image_file (&fp_ustar, u_filename);
215   write_byte_image ( fp_ustar, u_star, u_star_no_bytes, u_filename );
216   fclose (fp_ustar);                    // close output file
217
218   create_image_file (&fp_bmp, bmp_filename);
219   write_byte_image ( fp_bmp, bmp, bmp_no_bytes, bmp_filename );
220   fclose (fp_bmp);                    // close output file
221
222   create_image_file (&fp_btc_a, btc_a_filename);
223   write_byte_image ( fp_btc_a, btc_a, btc_no_bytes, btc_a_filename );
224   fclose (fp_btc_a);                    // close output file
225
226   create_image_file (&fp_btc_b, btc_b_filename);
227   write_byte_image ( fp_btc_b, btc_b, btc_no_bytes, btc_b_filename );

```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
 Updated 13:49 19 Aug 95;

page 6
 Printed 13:57

```

224 fclose (fp_btc_b); // close output file
225
226
227 show_results (u_star_no_bytes, bmp_no_bytes, btc_no_bytes, u_star_no_encoded_bits,
btc_no_encoded_bits, compression_ratio, Nu, Ne, nbtc, t, min_grad, max_grad, &pattern_count[0] );
228
229
230 } // end for loop
231
232 create_image_file (&fp_hist, "hist.int");
233 write_int_image ( fp_hist, hist, 256, "hist.int" );
234 fclose (fp_btc_b); // close output file
235
236 if (histogram_flag)
237 {
238   for (i=0; i < 256; i++)
239   {
240     no_of_pattern = no_of_pattern + hist[i];
241     printf("\nhist[%x] = %d", i, hist[i]);
242   }
243   printf("\nno_of_pattern = %d", no_of_pattern);
244 }
245
246 free (float_image);
247 free (b);
248 free (gm);
249 free (o);
250 free (u_star);
251 free (bmp);
252 free (btc_a);
253 free (btc_b);
254 free (hist);
255
256 printf("\nEnd of program!\n");
257
258 } // end main
259
260
261 //*****
262 //
263 //
264 //*****
265
266
267 //-----
268 //
269 // CommandLine - Get command line arguments, and parse the arglist.
270 //

```

Functions and Procedures

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
Updated 13:49 19 Aug 95;

page 7
Printed 13:57

```

271 //
272 // Parameters                Description                i/o
273 // -----                -----                -----
274 // subband                variable identifying the subband being processed                in
275 // nrow                size of a row in a block                in
276 // ncol                size of a column in a block                in
277 // subband_filename        filename of the subband to be processed                out
278 // u_filename                filename of the "u*" buffer related to the subband to be processed                out
279 // bmp_filename                filename of the "bmp" buffer related to the subband to be processed                out
280 // btc_a_filename            filename of the "btc_a" buffer related to the subband to be processed                out
281 // btc_b_filename            filename of the "btc_b" buffer related to the subband to be processed                out
282 //
283 //-----
284
285 void init_some_parameters ( subband, nrow, ncol, subband_filename, u_filename, bmp_filename,
btc_a_filename, btc_b_filename )
286 int subband;
287 int *nrow;
288 int *ncol;
289 char subband_filename[10];
290 char u_filename[12];
291 char bmp_filename[9];
292 char btc_a_filename[11];
293 char btc_b_filename[11];
294 {
295
296 switch (subband)
297 {
298 case 1:
299     *nrow = 64;
300     *ncol = 64;
301     subband_filename[4] = '1';
302     u_filename[6] = '1';
303     bmp_filename[3] = '1';
304     btc_a_filename[5] = '1';
305     btc_b_filename[5] = '1';
306     break;
307
308 case 2:
309     *nrow = 64;
310     *ncol = 64;
311     subband_filename[4] = '2';
312     u_filename[6] = '2';
313     bmp_filename[3] = '2';
314     btc_a_filename[5] = '2';
315     btc_b_filename[5] = '2';
316     break;
317
318 case 3:
319     *nrow = 64;

```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
Updated 13:49 19 Aug 95;

page 8
Printed 13:57

```
320     *ncol = 64;
321     subband_filename[4] = '3';
322     u_filename[6]      = '3';
323     bmp_filename[3]    = '3';
324     btc_a_filename[5]  = '3';
325     btc_b_filename[5]  = '3';
326     break;
327
328     case 4:
329         *nrow = 128;
330         *ncol = 128;
331         subband_filename[4] = '4';
332         u_filename[6]      = '4';
333         bmp_filename[3]    = '4';
334         btc_a_filename[5]  = '4';
335         btc_b_filename[5]  = '4';
336         break;
337
338     case 5:
339         *nrow = 128;
340         *ncol = 128;
341         subband_filename[4] = '5';
342         u_filename[6]      = '5';
343         bmp_filename[3]    = '5';
344         btc_a_filename[5]  = '5';
345         btc_b_filename[5]  = '5';
346         break;
347
348     case 6:
349         *nrow = 128;
350         *ncol = 128;
351         subband_filename[4] = '6';
352         u_filename[6]      = '6';
353         bmp_filename[3]    = '6';
354         btc_a_filename[5]  = '6';
355         btc_b_filename[5]  = '6';
356         break;
357
358     case 7:
359         *nrow = 256;
360         *ncol = 256;
361         subband_filename[4] = '7';
362         u_filename[6]      = '7';
363         bmp_filename[3]    = '7';
364         btc_a_filename[5]  = '7';
365         btc_b_filename[5]  = '7';
366         break;
367
368     case 8:
369         *nrow = 256;
370         *ncol = 256;
```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
Updated 13:49 19 Aug 95;

page 9
Printed 13 57

```

371  subband_filename[4] = '8';
372  u_filename[6]      = '8';
373  bmp_filename[3]    = '8';
374  btc_a_filename[5]  = '8';
375  btc_b_filename[5]  = '8';
376  break;
377
378  case 9:
379      *nrow = 256;
380      *ncol = 256;
381      subband_filename[4] = '9';
382      u_filename[6]      = '9';
383      bmp_filename[3]    = '9';
384      btc_a_filename[5]  = '9';
385      btc_b_filename[5]  = '9';
386      break;
387
388  } // end switch
389
390  subband_filename[9] = NULL;
391  u_filename[11]      = NULL;
392  bmp_filename[8]     = NULL;
393  btc_a_filename[10]  = NULL;
394  btc_b_filename[10]  = NULL;
395
396
397  } // end init_nrow_and_ncol
398
399  //-----
400  //
401  // CommandLine - Get command line arguments, and parse the arglist.
402  //
403  //
404  // Parameters          Description          i/o
405  // -----
406  // min_grad            minimum perceivable gradient magnitude (low threshold)      out
407  // max_grad            maximum gradient magnitude (high threshold)                  out
408  // nbtc                BTC coefficients block quantization parameter (no. of bits/coefficient) out
409  // t                   quaternary threshold parameter                             out
410  // start_band          initial subband parameter                                  out
411  // end_band            last subband parameter                                      out
412  // histogram_flag      flag enabling to build an histogram                        out
413  // argc                command-line argument                                     in
414  // argv                command-line argument                                     in
415  //
416  //-----
417
418  void CommandLine (min_grad, max_grad, nbtc, t, start_band, end_band, histogram_flag, argc, argv )
419  float *min_grad;
420  float *max_grad;
421  int *nbtc;

```


C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
 Updated 13:49 19 Aug 95;

page 10
 Printed 13:57

```

422 float *t;
423 int *start_band;
424 int *end_band;
425 BOOL *histogram_flag;
426 int argc;
427 char *argv[];
428 {
429
430 int CurrArg;
431
432 CurrArg = 0;
433 if (argc == 1) // if "wvpicq2" only is typed at the command line
434 ShowCommandLine();
435
436 // assuming wvpicq2 -m min_gradient -n max_grad -e nue -t t -a start_band -b end_band [h] as a
  minimum
437
438 if (argc < 12)
439 ShowCommandLine();
440
441 while (++CurrArg < argc)
442 {
443 if (argv[CurrArg][0] != '-')
444 ShowCommandLine();
445
446 switch(argv[CurrArg][1])
447 {
448 case 'm': // minimum image gradient magnitude
  parameter
449 if ((*min_grad = atof(argv[++CurrArg])) < 0)
450 error("error in min image gradient parameter...");
451 break;
452
453 case 'n': // maximum image gradient magnitude
  parameter
454 if ((*max_grad = atof(argv[++CurrArg])) < 1)
455 error("error in max image gradient parameter...");
456 break;
457
458 case 'c': // BTC coefficients edge parameter
459 if (((*nbtc = atoi(argv[++CurrArg])) < 1) || (*nbtc > 8))
460 error("error in mean intensity edge parameter...");
461 break;
462
463 case 't': // quaternary threshold parameter
464 if ((*t = atof(argv[++CurrArg])) < 0.0)
465 error("error in high quaternary threshold parameter...");
466 break;
467
468 case 'a': // initial subband parameter
469 if ((*start_band = atoi(argv[++CurrArg])) < 0)

```

C:\THESIS\TXM_VPIC\QERNARY2\WVPICQ2.C
Updated 13:49 19 Aug 95;

page 11
Printed 13:57

```
470         error("error in initial subband parameter...");
471         break;
472
473     case 'b':                                     // end subband parameter
474         if ((*end_band = atoi(argv[++CurrArg])) < 0)
475             error("error in end subband parameter...");
476         break;
477
478     case 'h':                                     // histogram flag
479         CurrArg++;
480         *histogram_flag = TRUE;
481         break;
482
483     } // end switch
484 } // end while loop
485 } // end CommandLine
```

END

0 5-0 6-9|6

FIN