

LOW COMPLEXITY H.264 TO VC-1 TRANSCODER

by

VIDHYA VIJAYAKUMAR

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

AUGUST 2010

Copyright © by Vidhya Vijayakumar 2010

All Rights Reserved

ACKNOWLEDGEMENTS

As true as it would be with any research effort, this endeavor would not have been possible without the guidance and support of a number of people whom I stand to thank at this juncture.

First and foremost, I express my sincere gratitude to my advisor and mentor, Dr. K.R. Rao, who has been the backbone of this whole exercise. I am greatly indebted for all the things that I have learnt from him, academically and otherwise.

I thank Dr. Ishfaq Ahmad for being my co-advisor and mentor and for his invaluable guidance and support. I was fortunate to work with Dr. Ahmad as his research assistant on the latest trends in video compression and it has been an invaluable experience.

I thank my mentor, Mr. Vishy Swaminathan, and my team members at Adobe Systems for giving me an opportunity to work in the industry and guide me during my internship.

I would like to thank the other members of my advisory committee Dr. W. Alan Davis and Dr. William E Dillon for reviewing the thesis document and offering insightful comments. I express my gratitude Dr. Jonathan Bredow and the Electrical Engineering department for purchasing the software required for this thesis and giving me the chance to work on cutting edge technologies. I also extend my thanks to SMPTE for providing the VC-1 test materials (sample encoder, reference decoder and bitstream).

I would like to thank all my good friends and fellow graduate students for their friendship and support during my stay at University of Texas at Arlington.

Finally, I thank my family for their constant motivation, support and wishes. I thank my husband, Balaji Nagarajan, especially, without whom I would not be what I am right now. His dreams and support towards my graduate studies has been special.

July 19, 2010

ABSTRACT

LOW COMPLEXITY H.264 TO VC-1 TRANSCODER

Vidhya Vijayakumar, MS

The University of Texas at Arlington, 2010

Supervising Professor: K.R.Rao

The high definition video adoption has been growing rapidly for the last five years. The H.264 and VC-1 coding standards are two of the recent and dominant coding standards. Compared to H.264, VC-1 is less complex but capable of achieving the same visual quality as H.264. Hence there is increasing importance of techniques which can convert video from H.264 to VC-1 and thereby enable mobile devices to work with less complex codecs. While there has been recent work on VC-1 to H.264 transcoding, the published work on H.264 to VC-1 transcoding is nearly non-existent. This has created the motivation to develop a transcoder that can efficiently transcode a H.264 bitstream into a VC-1 bitstream.

A low complex H.264 to VC-1 transcoder is proposed and developed in this research. The proposed architecture is similar to the cascaded transcoder architecture but with important changes. It consists of a complexity reduction module which is the key in this research. The aim of the complexity reduction module is to completely by-pass the motion estimation process, the mode decision process and also the transform size selection process. This is achieved by firstly extracting re-usable data from the incoming H.264 bitstream. Secondly, efficient re-use of mode decisions, macroblock partition size and motion vector are employed. The proposed transcoder was implemented in C programming language and was tested with video sequences at various bit rates and motion activity. The performance of the proposed transcoder is comparable to the reference cascaded

transcoder in subjective quality and is about 80% faster. This translates to memory and power savings in mobile devices which is the ultimate aim of this research.

TABLE OF CONTENTS

ABSTRACT	iv
LIST OF ILLUSTRATIONS.....	xii
LIST OF TABLES	xviii
LIST OF ACRONYMS	xx
Chapter	Page
1 INTRODUCTION.....	1
1.1 Significance of video	1
1.2 Significance of video compression and its standardization	2
1.3 Why is video transcoding important?	4
1.4 Requirement and usefulness of H.264 to VC-1 transcoder	4
1.4.1 Comparison of VC-1 with other codecs	6
1.4.1.1 Quality Comparison	6
1.4.1.2 Complexity Comparison.....	9
1.4.1.3 VC-1 Adoption.....	9
1.5 Summary.....	11
2 H.264 VIDEO CODING STANDARD	12
2.1 Introduction	12
2.2 Profiles and levels of H.264	14
2.2.1 Profiles in H.264	14
2.2.1.1 Baseline Profile	15
2.2.1.2 Main Profile	16
2.2.1.3 Extended Profile.....	16
2.2.1.4 High Profiles defined in the FRExts amendment.....	16
2.2.1.5 Overview of Scalable Video Coding	18
2.2.1.5.1 Spatial scalability	19
2.2.1.5.1.1 Inter-layer intra prediction.....	20

2.2.1.5.2SNR Scalability.....	21
2.2.1.5.2.1 Coarse-Grain Scalability.....	21
2.2.1.5.2.2 Fine-Grain Scalability	21
2.2.1.5.2.3 Medium-Grain Scalability.....	22
2.2.1.5.3Temporal scalability.....	22
2.2.2 Levels in H.264	23
2.3 H.264 Encoder.....	23
2.3.1 Intra-prediction.....	25
2.3.2 Inter-prediction.....	27
2.3.3 Transform coding.....	30
2.3.4 Deblocking filter	31
2.3.5 Entropy Coding	32
2.3.6 B-slices and adaptive weighted prediction	33
2.4 H.264 Decoder.....	35
2.5 Summary.....	35
3 VC-1 VIDEO CODING STANDARD.....	36
3.1 Introduction	36
3.2 Profiles and levels in VC-1.....	37
3.2.1 Profiles in VC-1	37
3.2.2 Levels in VC-1	38
3.3 VC-1 Codec structure	39
3.4 Coding concepts of VC-1	40
3.4.1 Color Space	40
3.4.2 Macroblocks, Blocks, and Sub-blocks	40
3.4.3 Transform Coding	41
3.4.4 Zigzag	41
3.4.5 Quantization.....	43
3.4.6 Bit plane Coding	43
3.4.7 VC-1 Intra Prediction	43

3.4.8 Motion Compensation.....	44
3.4.9 Huffman Coding.....	46
3.4.10 Intensity Compensation	47
3.4.11 Range Reduction	48
3.4.12 Overlap transform smoothing	48
3.4.13 In loop filtering.....	51
3.4.14 Intra and Inter macroblock encoding process.....	53
3.5 Innovations in VC-1.....	54
3.5.1 Adaptive block size transform.....	54
3.5.2 16-Bit Transforms	55
3.5.3 Motion Compensation.....	57
3.5.3.1 Sub-pixel Filters	58
3.5.3.2 Chrominance Channel	59
3.5.4 Quantization.....	60
3.5.5 Loop Filtering	61
3.5.6 Overlap transform	61
3.5.7 Interlaced Coding.....	63
3.5.8 Advanced B Frame Coding	63
3.5.9 Fading Compensation.....	63
3.6 Summary.....	64
4 COMPARISON OF H.264 AND VC-1	65
4.1 Introduction	65
4.2 Comparison of features and coding tool.....	65
4.3 Performance comparison.....	67
4.4 Results from the DVD Forum video codec tests [42].....	69
4.5 Computational complexity.....	71
4.6 Summary.....	73
5 TRANSCODING.....	74
5.1 Introduction	74

5.2 Video transcoding architectures	80
5.2.1 Cascaded decoder and encoder model.....	80
5.2.2 Open loop transcoding architecture.....	82
5.2.2.1 Picture drift error	82
5.2.2.2 Truncation of high-frequency coefficients	84
5.2.2.3 Re-quantization to reduce bit rate.....	84
5.2.3 Spatial domain transcoding architecture	85
5.2.4 Frequency domain transcoding architecture	86
5.2.5 Spatial resolution reduction	87
5.2.5.1 Filtering and sub-sampling.....	87
5.2.5.2 Pixel averaging	88
5.2.5.3 Discarding high frequency DCT coefficients.....	88
5.2.5.4 Motion vector composition and refinement.....	89
5.2.5.5 MB coding mode decision.....	91
5.2.6 Temporal resolution reduction	92
5.2.6.1 Bilinear Interpolation	92
5.2.6.2 Forward Dominant Vector Selection (FDVS).....	93
5.2.6.3 Telescopic Vector Composition (TVC).....	93
5.2.6.4 Activity-Dominant Vector Selection (ADVS)	93
5.2.7 Heterogeneous Transcoding	94
5.2.7.1 Main Issues in Heterogeneous Transcoding	94
5.2.7.2 Generic Heterogeneous Transcoder	94
5.3 Summary.....	96
6 PROPOSED TRANSCODER.....	97
6.1 Introduction	97
6.2 Cascaded reference transcoder	97
6.3 Choice of transcoding architecture	97
6.4 Proposed transcoder	98
6.4.1 Intra MB Mode Mapping	99

6.4.2 Inter MB Mode Mapping	99
6.4.3 Motion vector mapping	100
6.4.4 Reference Pictures:	101
6.4.5 Skipped Macroblock	102
6.4.6 Flowchart of proposed transcoder	102
6.4.7 Extraction of re-usable information.....	103
6.4.7.1 Extracted details	104
6.4.8 Simplified VC-1 Encoder	106
6.4.9 Implementation and Results	106
6.4.9.1 Implementation	106
6.4.9.2 Results	106
6.4.9.2.1 Comparison of proposed transcoder with cascaded transcoder with respect to H.264 video	107
6.4.9.2.1.1 Test sequence: Akiyo, QCIF Resolution, 10 frames.....	108
6.4.9.2.1.2 Test sequence: Miss America, QCIF Resolution, 10 frames.....	112
6.4.9.2.1.3 Test Sequence: Foreman, CIF Sequence, 10 frames.....	116
6.4.9.2.1.4 Test Sequence: Football, CIF Sequence, 10 frames.....	120
6.4.9.2.1.5 Test Sequence: Mobile, CIF Sequence, 10 frames.....	124
6.4.9.2.2 Comparison of proposed transcoder, cascaded transcoder and H.264 decoder with respect to original video	128
6.4.9.2.2.1 Test sequence: Akiyo, QCIF Resolution, 10 frames	129
6.4.9.2.2.2 Test sequence: Miss America, QCIF Resolution, 10 frames.....	132
6.4.9.2.2.3 Test sequence: Foreman, CIF Resolution, 10 frames.....	135
6.4.9.2.2.4 Test sequence: Football, CIF Resolution, 10 frames.....	138
6.4.9.2.2.5 Test sequence: Mobile, CIF Resolution, 10 frames.....	141

6.4.9.3 Observations	143
6.5 Conclusions and future work	144
6.5.1 Conclusions	144
6.5.2 Future work.....	144
REFERENCES.....	145
BIOGRAPHICAL INFORMATION	152

LIST OF ILLUSTRATIONS

Figure	Page
1.1 Home media ecosystem [4].....	2
1.2 An example of current VC-1 technology applications [4]	6
1.3 DVD Forum Quality Test [28]	8
1.4 An application scenario for H.264 to VC-1 transcoding [29]	11
2.1 Different profiles in H.264 with distribution of various coding tools among the profiles [32]	15
2.2 Tools introduced in FRExts and their classification under the new high profiles [35]	17
2.3 Scalable video coding [39]	19
2.4 The basic types of scalability in video coding [39]	19
2.5 H.264 Encoder block diagram [1].....	24
2.6 4x4 Luma prediction (intra-prediction) modes in H.264 [1]	25
2.7 4x4 Luma prediction (intra-prediction) modes in H.264 [1]	26
2.8 Chroma sub sampling [1]	27
2.9 Macroblock portioning in H.264 for inter prediction [1] (a) 16x16, 8x16, 16x8, 8x8 blocks (b) 8x8, 4x8, 8x4, 4x4 blocks	28
2.10 Interpolation of luma half-pel positions [1]	29
2.11 Interpolation of luma quarter-pel positions [1].....	29
2.12 Motion compensated prediction with multiple reference frames [1].....	30
2.13 H.264 Transformation (a) DC coefficients of 16 4x4 luma blocks, 4 4x4 Cb and Cr blocks [1] (b) Matrices H1, H2 and H3 of the three transforms used in H.264 [32]	31
2.14 Boundaries in a macroblock to be filtered (luma boundaries shown with solid lines and chroma boundaries shown with dotted lines) [1].....	32
2.15 Schematic block diagram of CABAC [1].....	33
2.16 Partition prediction examples in a B macroblock type (a) past/future, (b) past, (c) future [1].....	34

2.17 H.264 Decoder block diagram [1].....	35
3.1 Block diagram of VC-1 encoder [42]	40
3.2 8x8 Transform in VC-1 [79]	41
3.3 4x4 Transform in VC-1 [79]	41
3.4 Intra Zig Zag scan in VC-1 (a) Normal (b) Horizontal (c) Vertical [79].....	42
3.5 Inter Zig Zag scan in VC-1 (a) 8x8 (b) 4x8 (c)8x4 (d) 4x4 [79]	42
3.6 Intra AC/DC Prediction of 2 MBs in VC-1 [79]	44
3.7 Partition for Motion Compensation in VC-1 [79].....	45
3.8 Integer, half and quarter pel positions [42].....	46
3.9 Intensity compensation in VC-1 [79]	47
3.10 Intensity mapping and look up table [79]	48
3.11 Blocky effect due to quality discontinuity [79] (a) High-Low (b) Low-Low (c) High - High	49
3.12 Concept of over lapped transform smoothing [79]	49
3.13 Over lapped transform [79]	50
3.14 Definition of edge pixels in OLT in VC-1 [79]	50
3.15 Filtering exception of in loop filtering	51
3.16 Typical block boundaries in P frames for In loop filtering in VC-1 [79]	51
3.17 Definition of edge pixels in in loop filtering [79].....	52
3.18 Coding of Intra blocks [10]	53
3.19 Coding of inter blocks [10].....	53
3.20 Macroblock (4:2:0) [1].....	54
3.21 Transform sizes allowed in VC-1 [42]	55
3.22 VC-1 quantization and dequantization rules showing (a) dead-zone and (b) regular uniform quantization [42]	60
4.1 Glasgow - PSNR vs. Bitrate for WMV-9 Main (without B frames) and H.264/AVC Baseline, in favor of the former [42]	68
4.2 Stefan - PSNR vs. Bitrate for WMV-9 Main (without B frames) and H.264/AVC Baseline, in favor of the former [42]	68

4.3 Average perceptual scores of VC-1 in comparison to D5 and D-VHS references, as performed by experts in the DVD Forum [42].....	71
5.1 Communication between various multimedia devices [55]	74
5.2 Video transcoding operations [12].....	75
5.3 Video coding standards (a) Timeline [58] (b) Increase in complexity [57]	77
5.4 Video transcoding operations and classification [12]	78
5.5 Various ways of spatial transcoding [12].....	79
5.6 Transcoding with normal down sampling and user-interest-based down sampling.....	79
5.7 Cascaded decoder and encoder model (a) block level diagram (b) detailed diagram	81
5.8 Open loop transcoder architecture [55].....	82
5.9 Bit rate reduction by truncation of high frequency coefficients [12]	84
5.10 Transcoding with re-quantization scheme [12]	85
5.11 Spatial domain transcoding architecture with MV reuse and STR [12].....	86
5.12 Simplified SDTA without STR [12]	86
5.13 Frequency domain transcoding architecture (FDTA) [12].....	87
5.14 Decimation on 16 x 16 MB [61].....	88
5.15 Pixel averaging [61].....	88
5.16 DCT domain decimation for SRR [61].....	89
5.17 Four motion vectors being down sampled to one [12]	89
5.18 Motion vector refinement using search window (a) best case – small search (b) worst case – long search [64]	91
5.19 Four macroblock types down sampled to one [12]	92
5.20 FDVS motion vector composition scheme for TRR [12]	93
5.21 Heterogeneous video transcoder [12].....	95
6.1 Cascade decoder and encoder	97
6.2 Proposed transcoder architecture	98
6.3 Selection of motion vectors in different cases.....	101
6.4 Proposed transcoder flowchart	103

6.5 Sample extracted information from H.264 bitstream	
(a) I frame (b) P frame	105
6.6 Transcoder Results - Akiyo, QCIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder	109
6.7 Transcoder Results - Akiyo, QCIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder	109
6.8 Transcoder Results - Akiyo, QCIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder	110
6.9 Transcoder Results - Akiyo, QCIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder	110
6.10 Transcoder Results - Akiyo, QCIF, Percentage change in bitstream file size, cascade transcoder Vs proposed transcoder	111
6.11 Akiyo sequence	
(a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10	111
6.12 Transcoder Results – Miss America, QCIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder	113
6.13 Transcoder Results – Miss America, QCIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder	113
6.14 Transcoder Results – Miss America, QCIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder	114
6.15 Transcoder Results – Miss America, QCIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder.....	114
6.16 Transcoder Results – Miss America, QCIF, Percentage change in Y bitstream file size, cascade transcoder Vs proposed transcoder	115
6.17 Miss America sequence	
(a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10	115
6.18 Transcoder Results – Foreman, CIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder	117
6.19 Transcoder Results – Foreman, CIF, Comparison of Y peak signal to noise ration, cascade transcoder Vs proposed transcoder.....	117
6.20 Transcoder Results – Foreman, CIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder	118
6.21 Transcoder Results – Foreman, CIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder	118
6.22 Transcoder Results – Foreman, CIF, Percentage change in Y mean bitstream file size, cascade transcoder Vs proposed transcoder	119

6.23 Foreman sequence	
(a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10	119
6.24 Transcoder Results – Football, CIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder	121
6.25 Transcoder Results – Football, CIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder	121
6.26 Transcoder Results – Football, CIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder	122
6.27 Transcoder Results – Football, CIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder	122
6.28 Transcoder Results – Football, CIF, Percentage change in bitstream file size, cascade transcoder Vs proposed transcoder	123
6.29 Football sequence	
(a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10	123
6.30 Transcoder Results – Mobile, CIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder	125
6.31 Transcoder Results – Mobile, CIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder	125
6.32 Transcoder Results – Mobile, CIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder	126
6.33 Transcoder Results – Mobile, CIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder	126
6.34 Transcoder Results – Mobile, CIF, Percentage change in bitstream file size, cascade transcoder Vs proposed transcoder	127
6.35 Football sequence	
(a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10	127
6.36 Transcoder Results - Akiyo, QCIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder	130
6.37 Transcoder Results - Akiyo, QCIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder	130
6.38 Transcoder Results - Akiyo, QCIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder	131
6.39 Transcoder Results – Miss America, QCIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder	133
6.40 Transcoder Results - Miss America, QCIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder	133

6.41 Transcoder Results - Miss America, QCIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder	134
6.42 Transcoder Results – Foreman, CIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder	136
6.43 Transcoder Results - Foreman, CIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder	136
6.44 Transcoder Results - Foreman, CIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder	137
6.45 Transcoder Results – Football, CIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder	139
6.46 Transcoder Results - Football, CIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder	139
6.47 Transcoder Results - Football, CIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder	140
6.48 Transcoder Results – Mobile, CIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder	142
6.49 Transcoder Results - Mobile, CIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder	142
6.50 Transcoder Results - Mobile, CIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder	143

LIST OF TABLES

Table	Page
1.1 Raw bit rate of uncompressed video	3
1.2 Ranking of codec by clip on a scale of 1(best) – 5(worst) [28]	8
1.3 Complexity comparison of VC-1 and H.264 for different video clips [28].....	9
2.1 Comparison of the high profiles and corresponding coding tools introduced in the FRExts [35]	18
2.2 Levels defined in H.264.....	23
3.1 List of profiles and levels in VC-1 [10].....	37
3.2 Features in VC-1 profiles [10]	38
3.3 Levels in VC-1 [41]	39
4.1 Comparison of H.264 baseline and VC-1 simple profile tools	66
4.2 Codec ranking for the three codecs that performed best in the DVD Forum - final set of tests. [42]	71
4.3 Comparison of WMV-9 main and H.264/AVC baseline decoder complexity in an ARM chip. [42]	72
5.1 Different multimedia applications and corresponding video standards [14].....	76
6.1 H.264 and VC-1 Intra MB mapping	99
6.2 H.264 and VC-1 Inter MB mapping and VC-1 transform type	100
6.3 H.264 and VC-1 Inter MB motion vector mapping	100
6.4 Transcoder results for Akiyo QCIF sequence, cascade transcoder Vs proposed transcoder	108
6.5 Transcoder results for Akiyo QCIF sequence, Time saving, cascade transcoder Vs proposed transcoder	109
6.6 Transcoder results for Miss America QCIF sequence, cascade transcoder Vs proposed transcoder.....	112
6.7 Transcoder results for Miss America QCIF sequence, Time saving, cascade transcoder Vs proposed transcoder	113
6.8 Transcoder results for Foreman CIF sequence, cascade transcoder Vs proposed transcoder.....	116

6.9 Transcoder results for Foreman CIF sequence, Time saving, cascade transcoder Vs proposed transcoder	117
6.10 Transcoder results for Football CIF sequence, cascade transcoder Vs proposed transcoder	120
6.11 Transcoder results for Football CIF sequence, Time saving, cascade transcoder Vs proposed transcoder	121
6.12 Transcoder results for Mobile CIF sequence, cascade transcoder Vs proposed transcoder	124
6.13 Transcoder results for Mobile CIF sequence, Time saving, cascade transcoder Vs proposed transcoder	125
6.14 Transcoder results for Akiyo QCIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder	129
6.15 Transcoder results for Miss America QCIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder	132
6.16 Transcoder results for Foreman CIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder	135
6.17 Transcoder results for Football CIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder	138
6.18 Transcoder results for Mobile CIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder	141

LIST OF ACRONYMS

3G/4G	Third or Fourth Generation
3GPP	3rd Generation Partnership Project
ASO	Arbitrary slice ordering
AVC	Advanced Video Coding
AVI	Audio Video Interleave
B MB	Bi-predicted MB
CIF	Common Intermediate Format
DCT	Discrete Cosine Transform
DP	Data Partition
DSP	Digital Signal Processing
DVD	Digital Versatile Disc
FMO	Flexible macroblock ordering
FRExt	Fidelity Range Extensions
GOP	Group Of Pictures
IDR	Instantaneous Decoder Refresh
I MB	Intra Predicted MB
IEC	International Electro Technical Commission
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union – Transmission sector
JM	Joint Model
JVT	Joint Video Team
P MB	Inter Predicted MB
IDCT	Inverse Discrete Cosine Transform
IQ	Inverse Quantizer
MB	Macroblock

MBAFF	Macroblock level Adaptive Frame/Field
PicAFF	Picture level Adaptive Frame/Field
ME	Motion Estimation
MC	Motion Compensation
MV	Motion Vector
MPEG	Moving Picture Experts Group
MSE	Mean Square Error
NAL	Network Abstraction Layer
PSNR	Peak –to – peak Signal to Noise Ratio
Q	Quantizer
QCIF	Quarter Common Intermediate Format
R-D	Rate – Distortion
SP/SI	Switched P / Switched I
SMPTE	Society of Motion Picture and Television Engineers
SSIM	Structural Similarity Index Measure
SVC	Scalable Video Coding
VCEG	Video Coding Experts Group
VLC	Variable Length Coding
VLD	Variable Length Decoder
YUV	Y- Luminance and UV- Chrominance

CHAPTER 1

INTRODUCTION

1.1 Significance of video

Pervasive, seamless, high-quality digital video has been the goal of companies, researchers and standards bodies over the last two decades [1] [2] [3]. In some areas (for example broadcast television and consumer video storage), digital video has clearly captured the market, whilst in others (videoconferencing, video email, mobile video), market share is growing by the day. However, there is no doubt that digital video is a globally important industry which will continue to pervade businesses, networks and homes. The continuous evolution of the digital video industry is being driven by commercial and technical forces. The commercial drive comes from the huge revenue potential of persuading consumers and businesses to replace analog technology and older digital technology with new, efficient, high-quality digital video products and to adopt new communication and entertainment products. The technical drive comes from continuing improvements in processing performance, the availability of higher-capacity storage and transmission mechanisms and research and development of video and image processing technology [1] [2] [3]. Figure 1.1 gives an example of a home media ecosystem and the importance of video.



Figure 1.1 Home media ecosystem [4]

1.2 Significance of video compression and its standardization

Getting digital video from its source (a camera or a stored clip) to its destination (a display) involves a chain of components or processes. Key to this chain are the processes of compression (encoding) and decompression (decoding), in which bandwidth-intensive 'raw' digital video is reduced to a manageable size for transmission or storage and then reconstructed for display. Getting the compression and decompression processes 'right' can give a significant technical and commercial edge to a product, by providing better image quality, greater reliability and/or more flexibility than competing solutions. There is therefore a keen interest in the continuing development and improvement of video compression and decompression methods and systems [1] [2] [3] .

Data compression involves taking advantage of the redundancy in data to represent the information in compact form [5]. There are lossless as well as lossy compression techniques. If the data has been losslessly compressed, the original data can be recovered exactly with no loss. It is applied in areas where data loss can be detrimental. Text compression is an important area of lossless compression [5]. Lossy compression involves some loss of information; so the data cannot be recovered exactly. In exchange for such tolerable distortion much higher compression can be achieved. Speech compression can be an application where loss of information such as high frequency sounds above human hearing capacity can be lost without loss of fidelity. Video compression also involves lossy

compression [5]. Video has both spatial and temporal redundancies. Getting rid of these redundancies and other lossy techniques facilitate very high compression. The volume of video data is usually very high. For example, in order to digitally represent 1 second of video without compression (using CCIR 601 format [6]), more than 20 Mega bytes or 160 Mega bits is required [5]. This amount of data indicates the importance of compression for video signals since bandwidth usage plays a crucial role. Table 1.1 gives an idea of the bit rate needed for raw video without any compression.

Table 1.1 Raw bit rate of uncompressed video

Code	Width	Height	Description	Bit rate
QCIF	176	144	Quarter CIF	9 Mbps
QVGA	320	240	Quarter Video Graphics Array	27 Mbps
CIF	352	288	Common Intermediate Format	36 Mbps
HVGA	640	240	Half Video Graphics Array	55 Mbps
VGA	640	480	Video Graphics Array	110 Mbps
SD	720	486	Standard Definition	125 Mbps
SVGA	800	600	Super Video Graphics Array	172 Mbps
XGA	1024	768	Extended Graphics Array	283 Mbps
XGA+	1152	768	Extended Graphics Array plus	318 Mbps
	1152	864		358 Mbps
SXGA	1280	1024	Super Extended Graphics Array	471 Mbps
SXGA+	1400	1050	Super Extended Graphics Array plus	529 Mbps
UXGA	1600	1200	Ultra Extended Graphics Array	691 Mbps
HD	1920	1080	High Definition	746 Mbps
QXGA	2048	1536	Quad Extended Graphics Array	1.1 Gbps

Multimedia applications are targeted for a wide range of applications such as video conferencing, video on demand, mobile video broadcast and even medical applications. Given such wide range of applications, standardization of video compression techniques is essential. Standardization ensures interoperability of implementation from different vendors thereby enabling end-users to access video from different services both software and hardware [7]. There are numerous video compression standards both open-source and proprietary depending on the application and end-usage [8] [9] [10].

Experts from academia and industry have formed the Moving Pictures Experts Group (MPEG) and Video Coding Experts Group (VCEG) with the aim to bring standardization in coding moving pictures or video. The MPEG and VCEG joined together to form the Joint Video Team (JVT) in 2001 which developed the ITU-T Rec. H.264 | ISO/IEC 14496-10, commonly referred to as H.264/MPEG-4-AVC, H.264/AVC, or MPEG-4 Part 10 AVC [9].

VC-1 [10] is the informal name of the Society of Motion Picture and Television Engineers (SMPTE) 421 M video codec standards, which was initially developed as a proprietary video format by Microsoft before it was released as a formal SMPTE standard video format on April 3, 2006. It is today a widely supported standard found in HD DVDs, Blu-ray Discs, Windows Media Video 9, and Microsoft's Silver light framework [10].

1.3 Why is video transcoding important?

Different video compression standards assume significance due to the difference in the access to network connectivity, bandwidth, computational capacity, display rate, etc. available to the end-user. To be able to deliver and reproduce video and other multimedia data flexibly according to the end-user's requirements and capability, content should be dynamically adapted to the user's environment. This can include altering characteristics such as bit-rate, frame-rate, spatial resolution, coding syntax and even content. The process of transcoding plays an important role fulfilling this requirement. A video transcoder is defined as an operation of converting video from one video format to another [11] [12]. Transcoding also has other applications such as in statistical multiplexing of video to maintain bandwidth, to include new attributes such as company logos, watermarks, etc., adding error resilience capabilities and others [11] [12].

1.4 Requirement and usefulness of H.264 to VC-1 transcoder

The high definition (HD) video adoption has been growing rapidly for the last five years. The high definition DVD format blue ray has mandated MPEG-2 [13] [16], H.264 [9] and VC-1 [10] as video compression formats. The coexistence of these different video coding standards creates a need for transcoding. As more and more end products use the

above standards, transcoding from one format to another adds value to the product's capability. While there has been recent work on MPEG-2 to H.264 transcoding [13] [16] [29] [30], VC-1 to H.264 transcoding [17], the published work on H.264 to VC-1 transcoding is nearly non-existent.

The H.264 video format has a very broad application range that covers all forms of digital compressed video from low bit-rate Internet streaming applications to HDTV broadcast and Digital Cinema applications with nearly lossless coding. Digital Satellite TV quality, for example, was reported to be achievable at 1.5 Mbit/s, compared to the current operation point of MPEG 2 video at around 3.5 Mbit/s [14]. To ensure compatibility and problem-free adoption of H.264/AVC, many standards bodies have amended or added to their video-related standards so that users of these standards can employ H.264/AVC. The Digital Video Broadcast project (DVB) approved the use of H.264/AVC for broadcast television in late 2004. The Advanced Television Systems Committee [15] (ATSC) standards body in the United States approved the use of H.264/AVC for broadcast television in July 2008. It has also been approved for use with the more recent ATSC-M/H (Mobile/Handheld) standard, using the AVC and SVC portions of H.264. With the application of the H.264 compression technology to the video surveillance industry, the quality of the video recordings became substantially improved.

VC-1 [10] [17] [18] is widely characterized as an alternative to the ITU-T/ MPEG video codec standard H.264/MPEG-4 AVC. VC-1 contains coding tools for interlaced video sequences as well as progressive encoding. The main goal of VC-1 development and standardization is to support the compression of interlaced content without first converting it to progressive, making it more attractive to broadcast and video industry professionals. The emphasis during development on reducing the computational power required by the VC-1 decoder provides advantages for a broad range of media consumers. Personal computer users can decode full 1080i/p (interlaced/progressive) resolution video with off-the-shelf hardware, making HD video delivery a reality for the home computer. Perhaps more important than the benefits of VC-1 to the personal computer market is its value in the consumer electronics space. Hardware supporting VC-1 includes next generation DVD players, set-top boxes, portable media devices, wireless phones, and more. Major industry

players are selecting VC-1 for its scalability and quality. Figure 1.2 gives an example of the current VC-1 applications.

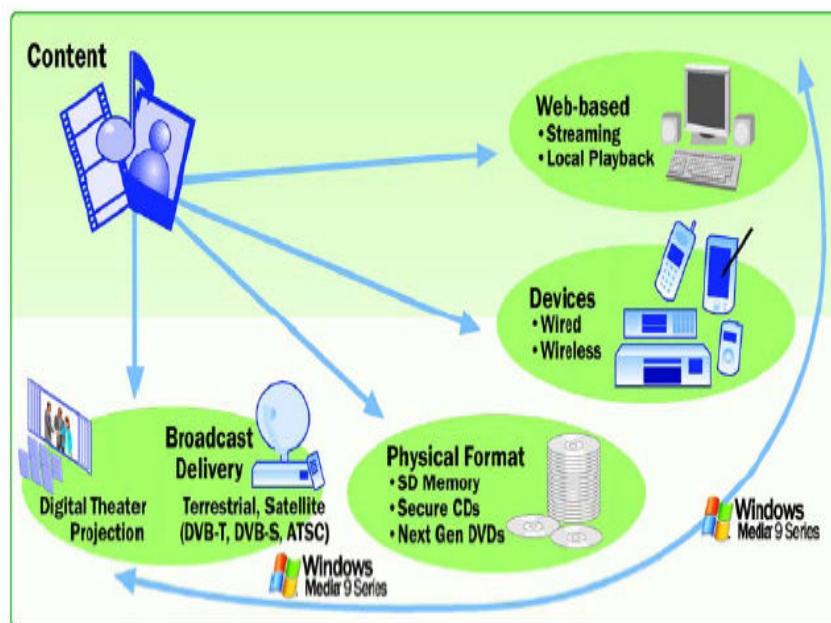


Figure 1.2 An example of current VC-1 technology applications [4]

1.4.1 Comparison of VC-1 with other codecs

VC-1 is very competitive when compared to other codecs in use today. This section compares the performance of VC-1 with MPEG-2 and H.264.

1.4.1.1 Quality Comparison

VC-1 and H.264 represent a logical technological evolution in video compression compared to MPEG-2. Both of these codecs are generally able to achieve superior quality over MPEG-2 at comparable bit rates.

Measuring the quality of a video codec is not easy, because the reconstructed image is not meant to be identical to the original. Ideally, only information that is perceptually irrelevant will be lost in the compression/decompression process, but what counts as "irrelevant" depends on the viewer's subjective response. It is important to note that an empirical codec comparison is always based on a practical implementation of a codec specification, which therefore means a compared codec is only as good as its implementation. It is very difficult to compare video compression standards in general terms

when there can be significant differences between quality and performance of various implementations within the same codec class. Codec comparison results can therefore vary greatly depending on selected encoder implementations, decoder (post-processing) implementations, video sources, encoding methods and user scenarios.

One useful objective metric is the peak signal-to-noise ratio (PSNR) plotted against bit rate. PSNR is the ratio between the maximum value of a signal (255 for 8-bit video) and the quantization noise. A higher PSNR indicates a less noisy signal. For any codec, PSNR is expected to increase at higher bit rates, because higher bit rates translate to less aggressive compression. Thus, a graph that plots PSNR against bit rate shows the performance of the codec over a range of compression settings.

The key arbiter of codec quality is the subjective appearance of the decoded video. For example, the DVD Forum [19] conducted tests to select codecs for the next-generation optical disc standard. Viewers from Hollywood film studios and major consumer electronics companies rated video clips on a scale of 1 to 5 for resolution, noise, and overall impression. Multiple codec implementations were tested, including MPEG-2 (24 Mbps represented as 24M and 7 Mbps represented as 7M), VC-1 (provided by Microsoft), H.264, and MPEG-4 Part 2 Advanced Simple Profile. The baselines against which the codecs were compared were D5 masters and D-VHS (24 Mbps).

On all the three measures (resolution, noise, and overall impression), the quality of the Microsoft VC-1 encoder was judged closest to the original D5 master. By comparison, the H.264 encoder that was tested rated as comparable only to MPEG-2 on two of the three measures (resolution and overall impression), and was rated somewhat worse than VC-1 on noise. Figure 1.3 shows the results of the DVD forum quality test. Table 1.2 shows the ranking of the codecs by clip on a scale of 1-5.

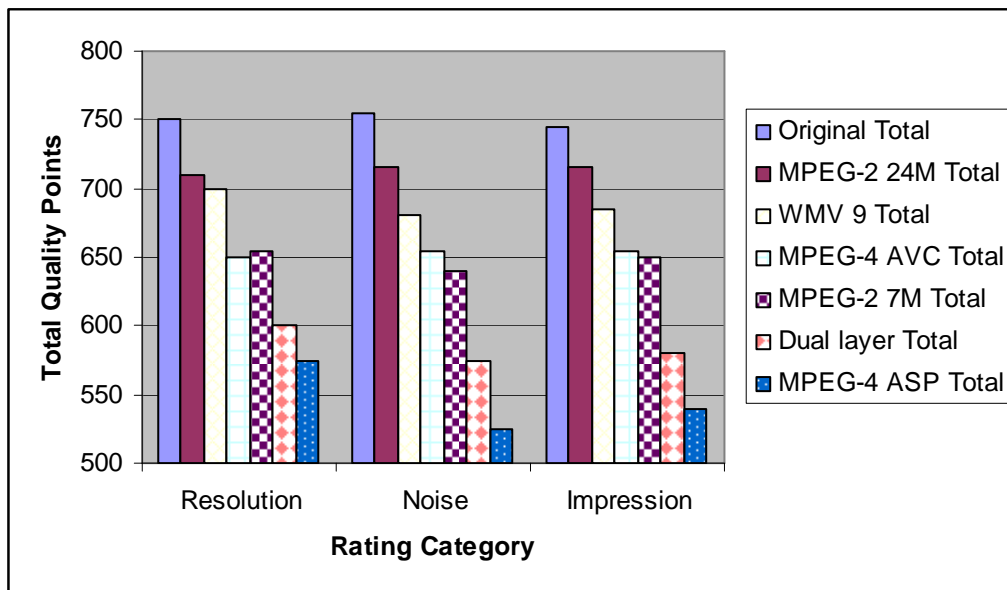


Figure 1.3 DVD Forum Quality Test [28]

Table 1.2 Ranking of codec by clip on a scale of 1(best) – 5(worst) [28]

	VC-1	H.264	MPEG-2
Dick Tracy	1	1	4
Titan AE	1	2	3
Harry Potter	1	2	2
Stuart Little 2	1	3	2
Seven	1	3	1
Monsters, Inc	1	2	3

VC-1 codecs have performed well in other independent subjective quality tests:

- DV Magazine [20] found VC-1 to be superior to both MPEG-2 and MPEG-4.
- TANDBERG Television [21] has found VC-1 produces significantly better quality than MPEG-2 and has quality comparable to H.264. These results were presented at the 2003 International Broadcasting Convention (IBC).
- C'T Magazine [22], Germany's premiere audio-video magazine, compared various codecs, including VC-1, H.264, and MPEG-2, and selected VC-1 as producing the best subjective and objective qualities for high-definition (HD) video.

- The European Broadcasting Union [23] (EBU) found VC-1 had the most consistent quality in tests that compared Microsoft VC-1 [18], RealMedia V9 [25], the Envivio MPEG-4 encoder [26], and the Apple MPEG-4 encoder [27].

1.4.1.2 Complexity Comparison

It is not enough to deliver high-quality video. A video codec must also be efficient to decode, particularly when the codec is implemented in hardware. Lower complexity means less silicon, lower cost, and fewer problems with power consumption and heat. Because they are more sophisticated, VC-1 and H.264 are both more complex to decode than MPEG-2. Yet VC-1 is more than twice as efficient to decode as H.264 [18]. A study by Third generation partnership project (3GPP), a collaboration group that sets up 3G mobile phone standards, found that VC-1 Main Profile requires 25 percent fewer cycles than the H.264 Baseline. It should be noted that H.264 Main Profile requires even more cycles than Baseline, because it includes highly complex arithmetic coding, also known as CABAC. In fact, software decoding of VC-1 at 1080p (1920 × 1080 progressive) resolution is possible on today's off-the-shelf computer hardware. In the hardware domain, companies can do more with a single DSP because VC-1 is easier to implement [18].

Table 1.3 Complexity comparison of VC-1 and H.264 for different video clips [28]

Sequence	Millions of ARM cycles/second	
	VC-1 Main	H.264 Baseline
Foreman	27	38
News	17	22
Container	19	24
Silent	18	25
Glasgow	25	30

1.4.1.3 VC-1 Adoption

VC-1 has already been adopted by the digital video industry and a number of standards bodies and industry organizations in addition to SMPTE.

- Next-Generation Optical Media - All of the leading next-generation optical media

formats have adopted VC-1 as a mandatory codec. The DVD Forum [19] has mandated VC-1, H.264, and MPEG-2 for the HD DVD format. The Blu-ray Disc Association [24] has mandated the same three codecs for their blue-laser Blu-ray Disc format. And the recent forward versatile disc (FVD) standard from Taiwan has adopted VC-1 as the only mandated video codec.

- Chips [18] - Numerous DSP and chip manufacturers have begun to support VC-1.
- Professional Video Equipment - VC-1 is being used for professional video broadcast and delivery today. Leading industry companies already have products on the market that support VC-1, ranging from encoders and decoders to professional video test equipment.
- Home Networks [18] - VC-1 is an optional format in the Digital Living Network Alliance (DLNA) standards. DLNA is developing a set of interoperability guidelines for home networks. These guidelines will enable computers, portable devices, and home consumer electronic devices such as stereos and set-top boxes to share digital media seamlessly over a home network.
- Mobile Devices [18] - VC-1 is one of the formats included in the Digital Video Broadcasting - Handheld (DVB-H) specification, and is a key component of Modeo's new DVB-H solution. VC-1 is also part of new broadband, Wi-Fi, and cellular delivery solutions such as MobiTV.

Considering the ubiquitous usage of H.264 and VC-1 and the advantages of VC-1 over H.264 in terms of both quality and complexity from an end user's point of view, a need for transcoding from H.264 to VC-1 is created. Figure 1.4 gives a typical application scenario.

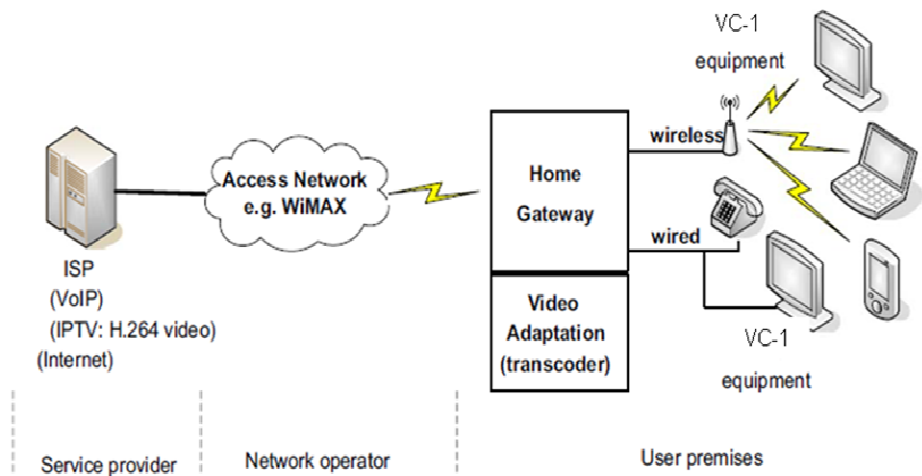


Figure 1.4 An application scenario for H.264 to VC-1 transcoding [29]

1.5 Summary

The research outline presented here proposes a reduced complexity transcoding technique to transcode an H.264 bitstream to a VC-1 bitstream. Before transcoding between these two codecs, it is important to understand the coding tools and syntax of both the codecs. The second chapter contains an overview of the H.264 codec. The third chapter describes the VC-1 standard. This chapter gives an overview of the coding technique used in VC-1 to achieve good quality at lower bit rates with minimal complexity. The fourth chapter deals with a comparison between the two standards. The fifth chapter describes the different transcoding architectures and the issues associated with heterogeneous transcoding. Finally, chapter six describes the proposed architecture, results, conclusion and suggestions on possible future work.

CHAPTER 2

H.264 VIDEO CODING STANDARD

2.1 Introduction

H.264/MPEG4-Part 10 advanced video coding (AVC) introduced in 2003 became one of the latest and most efficient video coding standards [9]. The H.264 standard was developed by the Joint Video Team (JVT), consisting of VCEG (Video Coding Experts Group) of ITU-T (International Telecommunication Union – Telecommunication standardization sector), and MPEG (Moving Picture Experts Group) of ISO/IEC [9].

H.264 can support various interactive (video telephony) and non-interactive applications (broadcast, streaming, storage, video on demand) as it facilitates a network friendly video representation. It leverages on the previous coding standards such as MPEG-1, MPEG-2, MPEG-4 part 2, H.261, H.262 and H.263 [32] and adds many other coding tools and techniques which give it superior quality and compression efficiency.

The standardization of the first version of H.264/AVC was completed in May 2003. The JVT then developed extensions to the original standard that are known as the fidelity range extensions (FRExt) [35]. These extensions enable higher quality video coding by supporting increased sample bit depth precision and higher-resolution color information, including sampling structures known as YUV 4:2:2 and YUV 4:4:4. Several other features are also included in the fidelity range extensions, such as adaptive switching between 4x4 and 8x8 integer transforms, encoder-specified perceptual-based quantization weighting matrices, efficient inter-picture lossless coding, and support of additional color spaces. The design work on the fidelity range extensions was completed in July 2004, and the drafting work on them was completed in September 2004.

Scalable video coding (SVC) [38] as specified in Annex G of H.264/AVC allows the construction of bitstreams that contain sub-bitstreams that conform to H.264/AVC. For temporal bitstream scalability, i.e., the presence of a sub-bitstream with a smaller temporal

sampling rate than the bitstream, complete access units are removed from the bitstream when deriving the sub-bitstream. In this case, high-level syntax and inter prediction reference pictures in the bitstream are constructed accordingly. For spatial and quality bitstream scalabilities, i.e. the presence of a sub-bitstream with lower spatial resolution or quality than the bitstream, network abstraction layer (NAL) units are removed from the bitstream when deriving the sub-bitstream. In this case, inter-layer prediction, i.e., the prediction of the higher spatial resolution or quality signal by data of the lower spatial resolution or quality signal, is typically used for efficient coding. The scalable video coding extension was completed in November 2007.

The next major feature added to the standard was Multiview Video Coding (MVC). Specified in Annex H of H.264/AVC, MVC enables the construction of bit streams that represent more than one view of a video scene. An important example of this functionality is stereoscopic 3D video coding. Two profiles were developed in the MVC work: one supporting an arbitrary number of views and designed specifically for two-view stereoscopic video. The Multi view Video Coding extensions were completed in November 2009 [14].

Like any other previous motion-based codecs, it uses the following basic principles of video compression [9]:

- Transform for reduction of spatial correlation
- Quantization for control of bit rate
- Motion compensated prediction for reduction of temporal correlation
- Entropy coding for reduction in statistical correlation.

The improved coding efficiency of H.264 can be attributed to the additional coding tools and the new features. Listed below are some of the new and improved techniques used in H.264 for the first time [33]:

- Adaptive intra-picture prediction
- Small block size transform with integer precision
- Multiple reference pictures and generalized B-frames
- Variable block sizes
- Quarter pixel precision for motion compensation

- Content adaptive in-loop deblocking filter and
- Improved entropy coding by introduction of CABAC (context adaptive binary arithmetic coding) and CAVLC (context adaptive variable length coding)

The increase in the coding efficiency and increase in the compression ratio result in a greater complexity of the encoder and the decoder algorithms of H.264, as compared to previous coding standards. In order to develop error resilience for transmission of information over the network, H.264 supports the following techniques [33]:

- Flexible macroblock ordering (FMO)
- Switched slice
- Arbitrary slice order (ASO)
- Redundant slice (RS)
- Data partitioning (DP)
- Parameter setting

2.2 Profiles and levels of H.264

The H.264/AVC standard is composed of a wide range of coding tools. Also, the standard addresses a large range of bit rates, resolutions, qualities, applications and services. Not all the tools and all the bit rates are required for any given application at a given point of time. All the various tools of H.264 are grouped in profiles.

2.2.1 Profiles in H.264

Profiles are defined as a subset of coding tools. They help to maximize the interoperability while limiting the complexity [34]. Also, the various levels define the various parameters like size of decoded pictures, bit rate, etc.

Figure 2.1 illustrates the coding tools for the various profiles of H.264. The profiles defined for H.264 can be listed as follows [35]:

- Constrained baseline profile
- Baseline profile
- Main profile

- Extended profile
- High profile
- High 10 profile
- High 4:2:2 profile
- High 4:4:4 predictive profile
- High stereo profile
- High 10 intra profile
- High 4:2:2 intra profile
- High 4:4:4 intra profile
- CAVLC 4:4:4 intra profile
- Scalable baseline profile
- Scalable high profile
- Scalable high intra profile

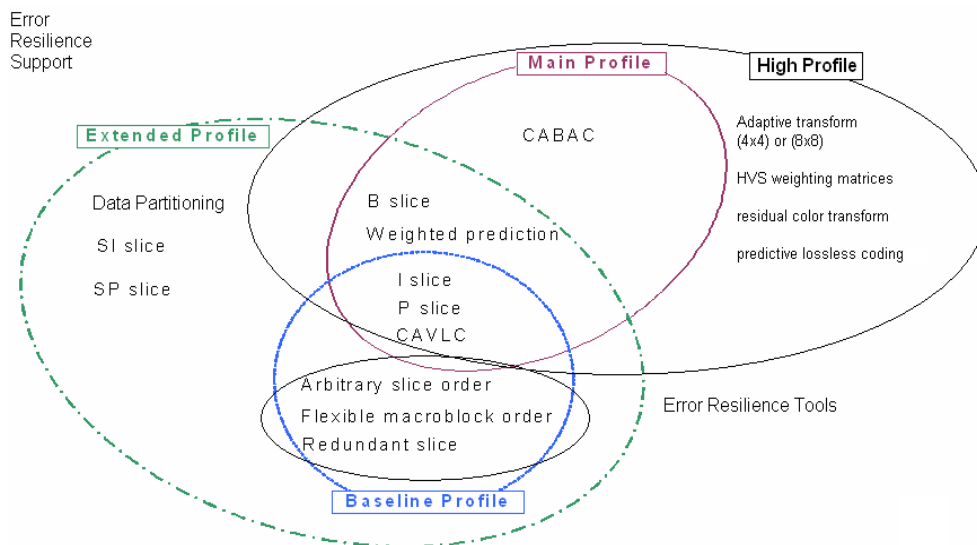


Figure 2.1 Different profiles in H.264 with distribution of various coding tools among the profiles [32]

2.2.1.1 Baseline Profile

The list of tools included in the baseline profile are I (intra coded) and P (predictive coded) slice coding, enhanced error resilience tools of flexible macroblock ordering, arbitrary slices and redundant slices. It also supports CAVLC (context-based adaptive

variable length coding). The baseline profile is intended to be used in low delay applications, applications demanding low processing power, and in high packet loss environments. This profile has the least coding efficiency among all the three profiles.

2.2.1.2 Main Profile

The coding tools included in the main profile are I, P, and B (bi directionally predictive coded) slices, interlace coding, CAVLC and CABAC (context-based adaptive binary arithmetic coding). The tools not supported by main profile are error resilience tools, data partitioning and SI (switched intra coded) and SP (switched predictive coded) slices. This profile is aimed to achieve highest possible coding efficiency.

2.2.1.3 Extended Profile

This profile has all the tools included in the baseline profile. As illustrated in the Figure 2.1 this profile also includes B, SP and SI slices, data partitioning, interlaced frame and field coding, picture adaptive frame/field coding and MB adaptive frame/field coding. This profile provides better coding efficiency than baseline profile. The additional tools result in increased complexity.

2.2.1.4 High Profiles defined in the FRExts amendment

In September 2004 the first amendment of H.264/MPEG-4 AVC video coding standard was released [35]. A new set of coding tools were introduced as a part of this amendment. These are termed as “Fidelity Range Extensions” (FRExts). The aim of releasing FRExts is to be able to achieve significant improvement in coding efficiency for higher fidelity material. It also has lossless representation capability: I PCM raw sample value macroblocks and entropy coded transform by-pass lossless macroblocks (FRExts only). The application areas for the FRExts tools are professional film production, video production and high-definition TV/DVD. The FRExts amendment defines four new profiles (refer Figure 2.2) [36]:

- High (HP)
- High 10 (Hi10P)

- High 4:2:2 (Hi422P)
- High 4:4:4 (Hi444P)

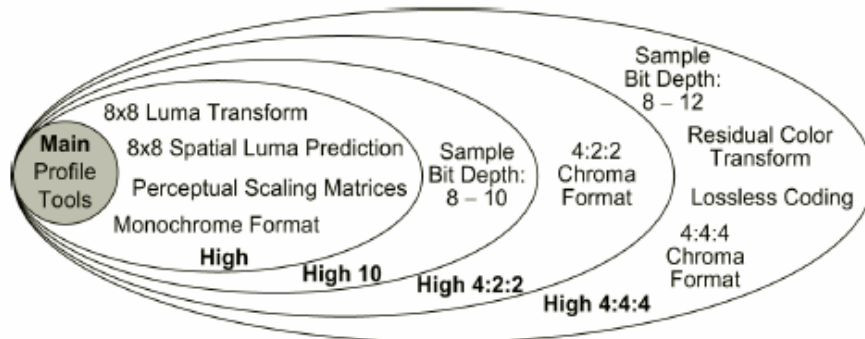


Figure 2.2 Tools introduced in FRExts and their classification under the new high profiles [35]

All four of these profiles build further upon the design of the prior Main profile. The provides a comparison of the high profiles introduced in FRExts with a list of different coding tools and which of them are applied to which profile. All of the high profiles include the following three enhancements of coding efficiency performance [37]:

- Adaptive macroblock-level switching between 8x8 and 4x4 transform block sizes

The main aim behind introducing 8x8 transform in FRExts was because high fidelity video demands preservation of fine details and textures. To achieve this, larger basis functions are required. However smaller transform like 4x4 reduces ringing artifacts and reduces computational complexity. The encoder adaptively choses between 4x4 and 8x8 transform.

The transform selection process is limited by the following conditions

- If an inter-coded Macroblock (MB) has sub-partition smaller than 8x8 (i.e. 4x8, 8x4, 4x4), then 4x4 transform is used.
- If an intra-coded MB is predicted using 8x8 luma spatial prediction, only 8x8 transform is used.
- Encoder-specified perceptual-based quantization scaling matrices

The encoder can specify a matrix for scaling factor according to the specific frequency associated with the transform coefficient for use in inverse quantization

scaling by the decoder. This allows optimization of the subjective quality according to the sensitivity of the human visual system, less sensitive to the coded error in high frequency transform coefficients [32].

- Encoder-specified separate control of the quantization parameter for each chroma component

Table 2.1 Comparison of the high profiles and corresponding coding tools introduced in the FRExts [35]

Coding tools	High	High 10	High 4:2:2	High 4:4:4
Main Profile tools			x	x
4:2:0 Chroma format			x	x
8 bit sample bit depth			x	x
8x8 vs. 4x4 transform adaptivity			x	x
Quantization scaling matrices			x	x
Separate Cb and Cr QP control			x	x
Monochrome video format			x	x
9 and 10 bit sample depth			x	x
4:2:2 Chroma format			x	x
11 and 12 sample bit depth				x
4:4:4 Chroma format				x
Residual color transform				x
Predictive lossless coding				x

2.2.1.5 Overview of Scalable Video Coding

A video bit stream is called scalable when parts of the stream can be removed in a way that the resulting sub-stream forms another valid bit stream for some target decoder, and the sub-stream represents the source content with a reconstruction quality that is less than that of the complete original bit stream. The modes of scalability are temporal, spatial, and quality [38]. Spatial scalability and temporal scalability describe cases in which subsets of the bit stream represent the source content with a reduced picture size (spatial resolution) or frame rate (temporal resolution), respectively. With quality scalability, the sub-stream provides the same spatial-temporal resolution as the complete bit stream, but with a lower signal-to-noise ratio (SNR). Quality scalability is also commonly referred to as fidelity or SNR scalability. Figure 2.3 shows the basic principle and applications of scalable coding. Figure 2.4 illustrates the different scalabilities in MPEG-4/H.264.

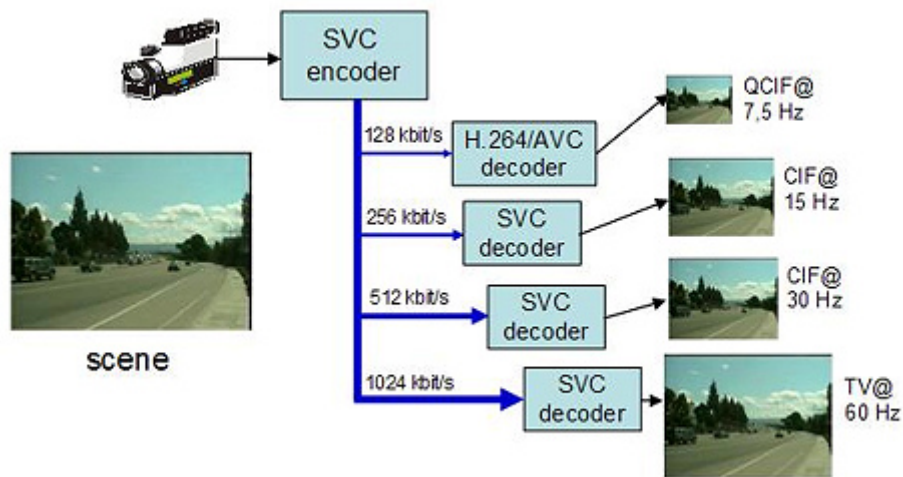


Figure 2.3 Scalable video coding [39]

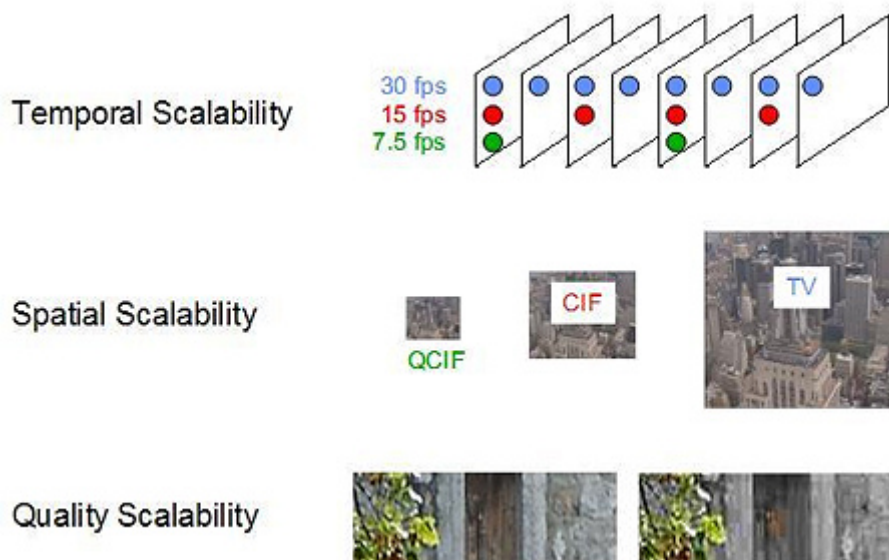


Figure 2.4 The basic types of scalability in video coding [39]

2.2.1.5.1 Spatial scalability

For supporting spatial scalable coding, SVC follows the conventional approach of multi-layer coding. In each spatial layer, motion-compensated prediction and intra prediction are employed as for single-layer coding. In addition to these basic coding tools of H.264/AVC, SVC provides inter-layer prediction methods, which allow an exploitation of the statistical dependencies between different layers for improving the coding efficiency of

enhancement layers. The supported inter-layer prediction methods employ the reconstructed samples of the lower layer signal. The prediction signal is either formed by motion-compensated prediction inside the enhancement layer, by up-sampling the reconstructed lower layer signal, or by averaging such an up-sampled signal with a temporal prediction signal. Apart from these, two additional inter-layer prediction concepts have been added in SVC: prediction of macroblock (MB) modes and associated motion parameters and prediction of the residual signal. All inter-layer prediction tools can be chosen on a macroblock or sub-macroblock basis allowing an encoder to select the coding mode that gives the highest coding efficiency [38].

2.2.1.5.1.1 Inter-layer intra prediction

For SVC enhancement layers, an additional macroblock coding mode (signaled by the syntax element `base_mode_flag` equal to 1) is provided, in which the macroblock prediction signal is completely inferred from co-located blocks in the reference layer without transmitting any additional side information. When the co-located reference layer blocks are intra-coded, the prediction signal is built by the up-sampled reconstructed intra signal of the reference layer – a prediction method also referred to as inter-layer intra prediction.

Inter-layer macroblock mode and motion prediction:

When `base_mode_flag` is equal to 1 and at least one of the co-located reference layer blocks is not intra-coded, the enhancement layer macroblock is inter-picture predicted as in single-layer H.264/AVC coding, but the macroblock partitioning – specifying the decomposition into smaller block with different motion parameters – and the associated motion parameters are completely derived from the co-located blocks in the reference layer. This concept is also referred to as inter-layer motion prediction. For the conventional inter-coded macroblock types of H.264/AVC, the scaled motion vector of the reference layer blocks can also be used as replacement for usual spatial motion vector predictor.

Inter-layer residual prediction:

A further inter-layer prediction tool referred to as inter-layer residual prediction targets a reduction of the bit rate required for transmitting the residual signal of inter-coded

macroblocks. With the usage of residual prediction (signaled by the syntax element `residual_prediction_flag` equal to 1), the up-sampled residual of the co-located reference layer blocks is subtracted from the enhancement layer residual (difference between the original and the inter-picture prediction signal) and only the resulting difference, which often has a smaller energy than the original residual signal, is encoded using transform coding as specified in H.264/AVC.

2.2.1.5.2 SNR Scalability

Different techniques exist in the Joint Scalable Video Model (JSVM) for providing SNR scalability [38].

2.2.1.5.2.1 Coarse-Grain Scalability

Coarse-Grain Scalability (CGS), which is similar to spatial scalability, uses different dependency layers with refinements. In the case of CGS, the difference is that no up sampling is required between successive enhancement layers. In every layer, quality refinements of the transform coefficients are stored by using a decreasing quantization step size. SVC supports up to eight CGS layers, corresponding to eight quality extraction points. Between successive refinement layers, inter-layer prediction is possible for both the motion information and the residual data. Also, the inter-layer intra prediction tool is used to further improve coding efficiency of intra-coded macroblocks.

2.2.1.5.2.2 Fine-Grain Scalability

Fine-Grain Scalability (FGS) uses an advanced form of bit-plane coding for encoding successive refinements of transform coefficients. The FGS slices have the property that they can be truncated at any byte-aligned position for SNR scalability. FGS SNR scalability has the advantage that it provides a larger degree of flexibility, allowing a quasi-continuous spectrum of achievable bitrates, while CGS is limited to a number of pre-determined bitrates, i.e., one extraction point per layer. Due to its high computational complexity, however, the FGS concept was not included in one of the SVC profiles. As a

consequence, it was removed from the Joint Draft. After further study and complexity reduction, FGS might be included in a future amendment to the current SVC specification.

2.2.1.5.2.3 Medium-Grain Scalability

As an alternative to FGS, Medium-Grain Scalability (MGS) was introduced. MGS tackles a number of problems that are encountered for CGS, such as the limited number of rate points, and the lack of flexibility for bitstream adaptation. MGS increases the number of achievable rate points by allowing different quality levels within one dependency layer. The flexibility is improved by allowing the removal of these quality levels at any point in the bitstream. Switching between the number of dependency layers (as is required for CGS), is only allowed at certain pre-defined points. Presently, 16 quality refinement levels are allowed for every dependency layer. In conjunction with CGS, this means that 128 quality extraction points are now achievable for SVC bit-streams.

2.2.1.5.3 Temporal scalability

A bit stream provides temporal scalability when the set of corresponding access units can be partitioned into a temporal base layer and one or more temporal enhancement layers with the following property - Let the temporal layers be identified by a temporal layer identifier T , which starts from 0 for the base layer and is increased by 1 from one temporal layer to the next. Then for each natural number k , the bit stream that is obtained by removing all access units of all temporal layers with a temporal layer identifier T greater than k forms another valid bit stream for the given decoder [38]. For hybrid video codecs, temporal scalability can generally be enabled by restricting motion-compensated prediction to reference pictures with a temporal layer identifier that is less than or equal to the temporal layer identifier of the picture to be predicted. H.264/AVC provides a significantly increased flexibility for temporal scalability because of its reference picture memory control. It allows the coding of picture sequences with arbitrary temporal dependencies, which are only restricted by the maximum usable DPB size. Hence, for supporting temporal scalability with

a reasonable number of temporal layers, no changes to the design of H.264/AVC are required. The only related change in SVC refers to the signaling of temporal layers.

2.2.2 Levels in H.264

In H.264 /AVC, 16 levels are specified. Each level defines upper bounds for the bit stream or lower bounds for the decoder capabilities. A profile and level can be combined to define the conformance points. These points signify the point of interoperability for applications with similar functional requirements. The levels defined in H.264 are listed in Table 2.2. The level '1b' was added in the FRExts amendment.

Table 2.2 Levels defined in H.264

Level number	Typical picture size	Typical frame rate	Maximum compression bit rate	Maximum number of reference frames
1	QCIF	15	64 kbps	4
1b	QCIF	15	128 kbps	4
1.1	CIF or QCIF	7.5 (CIF) / 30 (QCIF)	192 kbps	2 (CIF) / 9 (QCIF)
1.2	CIF	15	384 kbps	6
1.3	CIF	30	768 kbps	6
2	CIF	30	2 Mbps	6
2.1	HHR	30 / 25	4 Mbps	6
2.2	SD	15	4 Mbps	5
3	SD	30 / 25	10 Mbps	5
3.1	1280x720p	30	14 Mbps	5
3.2	1280x720p	60	20 Mbps	4
4	HD formats	60p / 30i	20 Mbps	4
4.1	HD formats	60p / 30i	50 Mbps	4
4.2	1920x1080p	60p	50 Mbps	4
5	2k x 1k	72	135 Mbps	5
5.1	2k x 1k or 4k x 2k	120 / 30	240 Mbps	5

2.3 H.264 Encoder

Figure 2.5 illustrates the schematic of the H.264 encoder. H.264 encoder works on macroblocks and motion-compensation like most other previous generation codecs. Video is formed by a series of picture frames. Each picture frame is an image which is split down into blocks. The block sizes can vary in H.264. The encoder may perform intra-coding or inter-coding for the macroblocks of a given picture. Intra coded frames are encoded and decoded independently. They do not need any reference frames. Hence they provide access points to the coded sequence where decoding can start. H.264 uses nine spatial prediction modes

in intra-coding to reduce spatial redundancy in the source signal of the picture. These prediction modes are explained in Section 2.3.1. Inter-coding uses inter-prediction of a given block from some previously decoded pictures. The aim is to use inter-coding to reduce the temporal redundancy by making use of motion vectors. Motion vectors give the direction of motion of a particular block from the current frame to the next frame. The prediction residuals are obtained which then undergo transformation to remove spatial correlation in the block. The transformed coefficients, thus obtained, undergo quantization. The motion vectors (obtained from inter-prediction) or intra-prediction modes are combined with the quantized transform coefficient information. They are then encoded using entropy code such as context-based adaptive variable length coding (CAVLC) or context-based adaptive binary arithmetic coding (CABAC) [32].

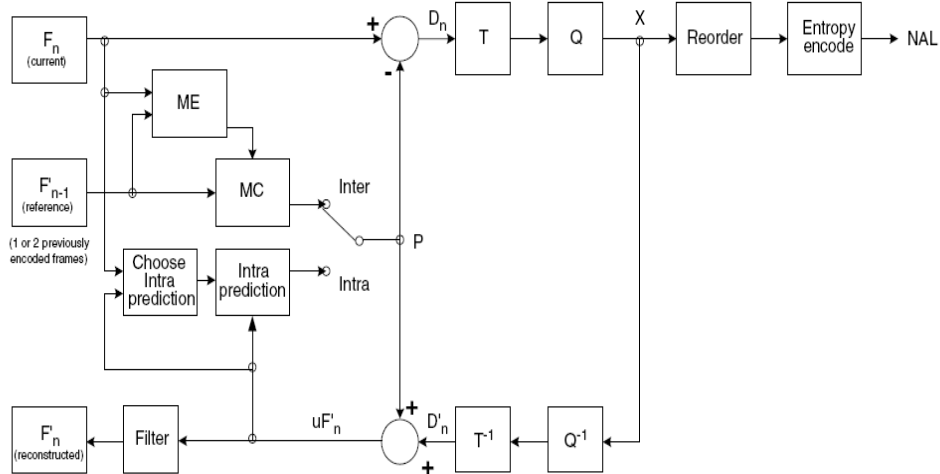


Figure 2.5 H.264 Encoder block diagram [1]

There is a local decoder within the H.264 encoder. This local decoder performs the operations of inverse quantization and inverse transform to obtain the residual signal in the spatial domain. The prediction signal is added to the residual signal to reconstruct the input frame. This input frame is fed in the deblocking filter to remove blocking artifacts at the block boundaries. The output of the deblocking filter is then fed to inter/intra prediction blocks to generate prediction signals.

The various coding tools used in the H.264 encoder are explained in the Sections 2.3.1 through 2.3.6.

2.3.1 Intra-prediction

Intra-prediction uses the macroblocks from the same image for prediction. Two types of prediction schemes are used for the luminance component. These two schemes can be referred as INTRA_4x4 and INTRA_16x16 [16]. In INTRA_4x4, a macroblock of size 16x16 pixels are divided into 16 4x4 sub blocks. Intra prediction scheme is applied individually to these 4x4 sub blocks. There are nine different prediction modes supported as shown in Figure 2.6. In FRExts profiles alone, there is also 8x8 luma spatial prediction (similar to 4x4 spatial prediction). FRExts also have low-pass filtering of the prediction to improve prediction performance.

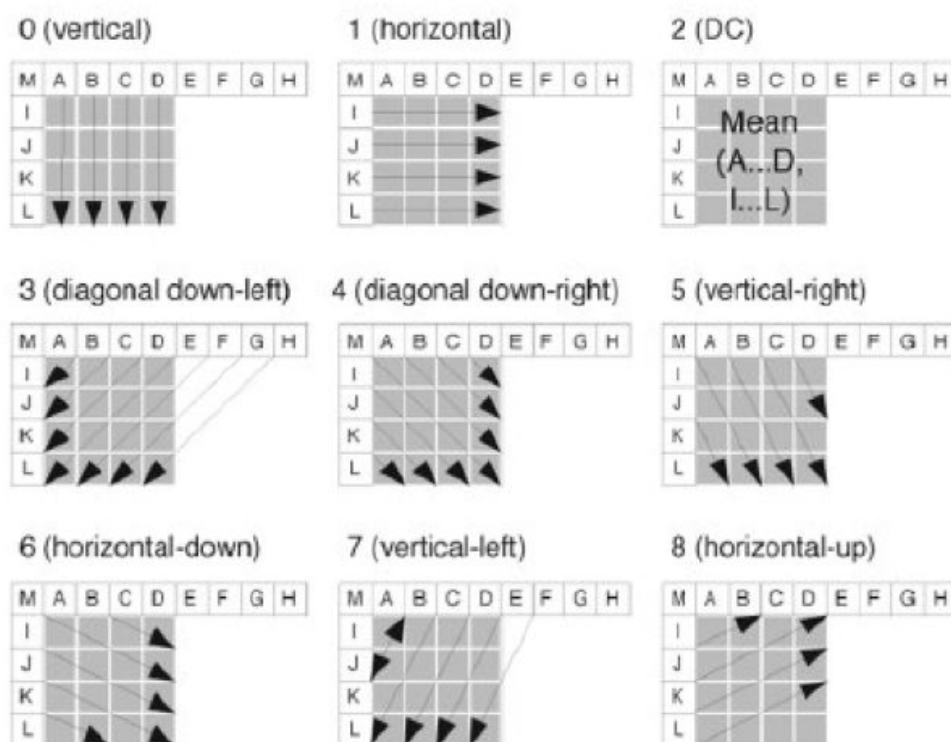


Figure 2.6 4x4 Luma prediction (intra-prediction) modes in H.264 [1]

In vertical mode, the samples of the macroblock are predicted from the neighboring samples on the top. In horizontal mode, the samples of the macroblock are predicted from the neighboring samples from the left. In DC mode, the mean of all the neighboring samples is used for prediction. Diagonal down left mode is in diagonally down-left direction. Diagonal down right mode is in diagonal down-right direction. Vertical right mode is in vertical-right

direction. Horizontal down mode is in horizontal-down direction. Vertical left mode is in vertical-left direction. Horizontal up mode is in horizontal up direction. The predicted samples are calculated from a weighted average of the previously predicted samples A to M.

For prediction of 16x16 intra block of luminance components, four modes are used. The three modes of mode 0 (vertical), mode 1 (horizontal) and mode 2 (DC) are similar to the prediction modes for 4x4 block. In the fourth mode, the linear plane function is fitted in the neighboring samples.

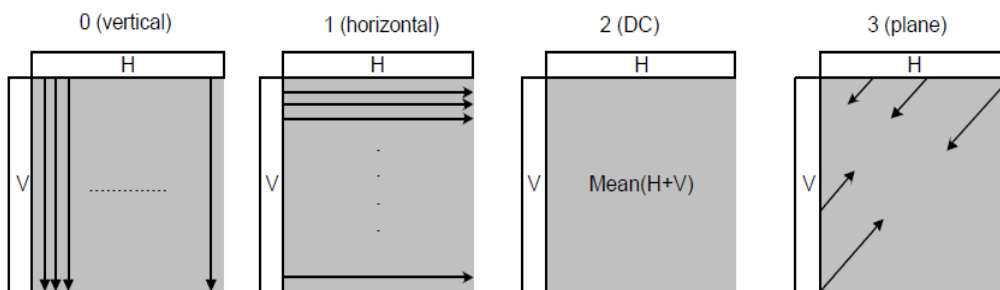


Figure 2.7 4x4 Luma prediction (intra-prediction) modes in H.264 [1]

The chroma macroblock is predicted from neighboring chroma samples. The four prediction modes used for the chroma blocks are similar to 16x16 luma prediction modes. The number in which the prediction modes are ordered is different for chroma macroblock: mode 0 is DC, mode 1 is horizontal, mode 2 is vertical and mode 3 is plane. The block sizes for the chroma prediction depend on the sampling format. For 4:2:0 format, the 8x8 size of chroma block is selected. For 4:2:2 format, the 8x16 size of chroma block is selected. For 4:4:4 format, the 16x16 size of chroma block is selected [1]. Figure 2.8 illustrates chroma sub sampling.

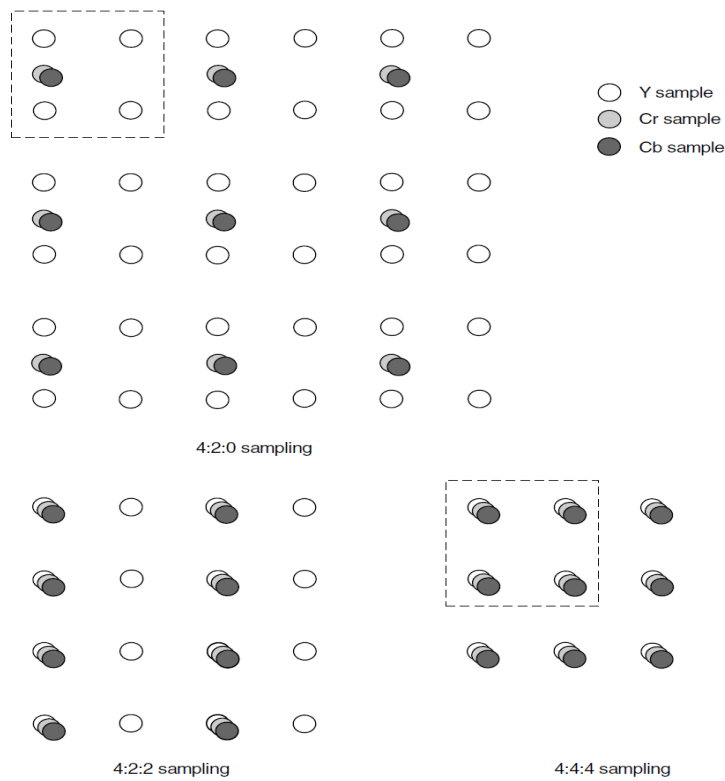


Figure 2.8 Chroma sub sampling [1]

2.3.2 Inter-prediction

Inter-prediction is used to capitalize on the temporal redundancy in a video sequence. The temporal correlation is reduced by inter prediction through the use of motion estimation and compensation algorithms [1]. An image is divided into macroblocks; each 16x16 macroblock is further partitioned into 16x16, 16x8, 8x16, 8x8 sized blocks. A 8x8 sub-macroblock can be further partitioned into 8x4, 4x8, 4x4 sized blocks. Figure 2.9 illustrates the partitioning of a macroblock and a sub-macroblock [1]. The input video characteristics govern the block size. A smaller block size ensures less residual data; however smaller block sizes also mean more motion vectors and hence a greater number of bits required to encode these motion vectors [1].

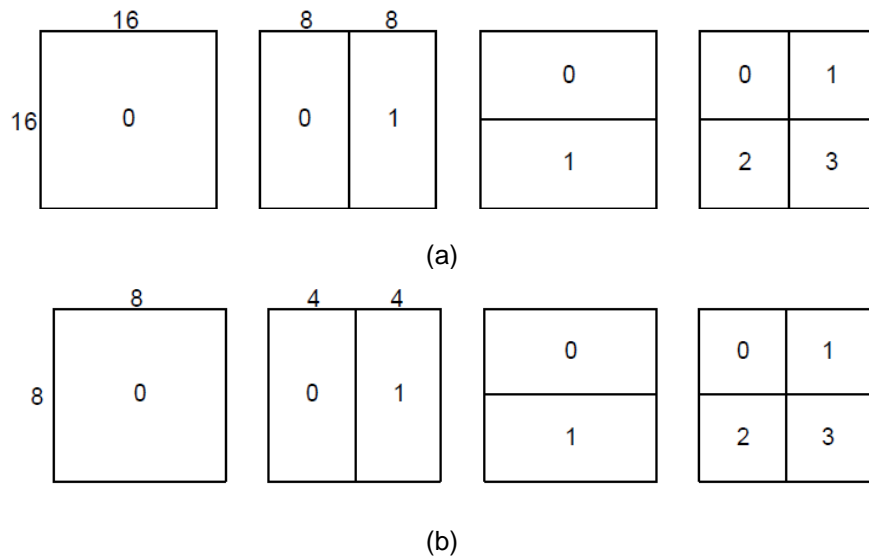


Figure 2.9 Macroblock partitioning in H.264 for inter prediction [1] (a) (L-R) 16x16, 8x16, 16x8, 8x8 blocks (b) (L-R) 8x8, 4x8, 8x4, 4x4 blocks

Each partition or sub-macroblock partition in an inter-coded macroblock is predicted from an area of the same size in a reference picture. The offset between the two areas (the motion vector) has quarter-sample resolution for the luma component and one-eighth-sample resolution for the chroma components. The luma and chroma samples at sub-sample positions do not exist in the reference picture and so it is necessary to create them using interpolation from nearby coded samples. Figure 2.10 and Figure 2.11 illustrate half and quarter pixel interpolation used in luma pixel interpolation respectively. Six-tap filtering is used for derivation of half-pel luma sample predictions, for sharper sub pixel motion-compensation. Quarter-pixel motion is derived by linear interpolation of the half pixel values, to save processing power.

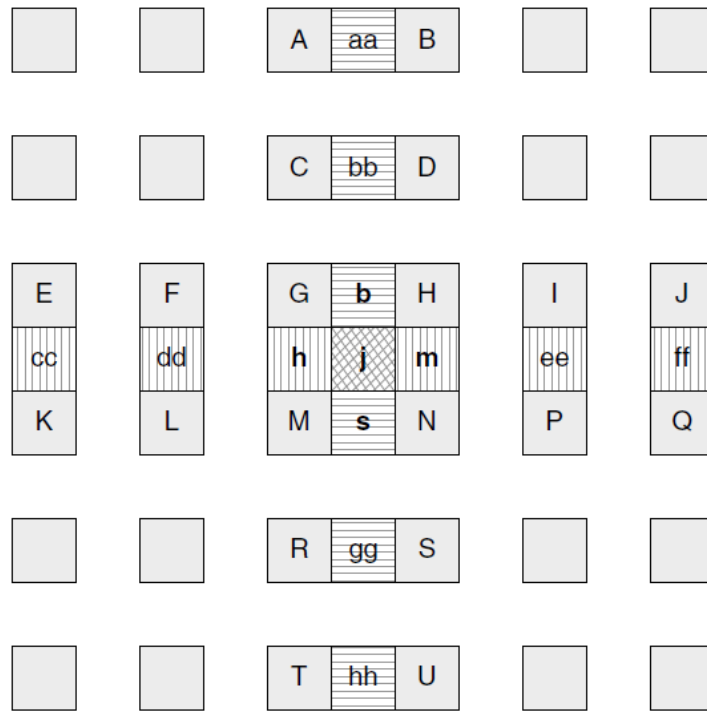


Figure 2.10 Interpolation of luma half-pel positions [1]

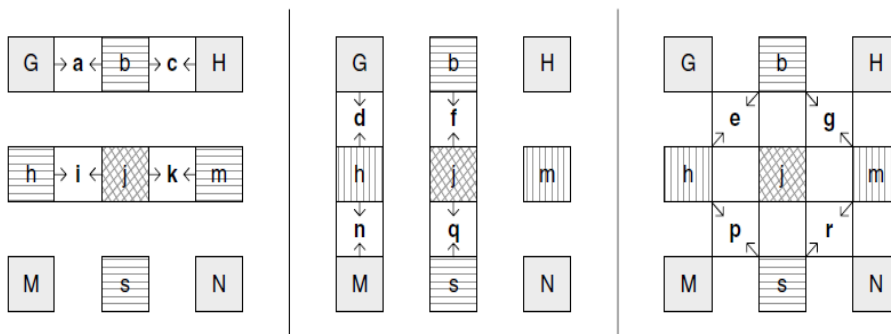


Figure 2.11 Interpolation of luma quarter-pel positions [1]

The reference pictures used for inter prediction are previously decoded frames and are stored in the picture buffer. H.264 supports the use of multiple frames as reference frames. This is implemented by the use of an additional picture reference parameter which is transmitted along with the motion vector. Figure 2.12 illustrates an example with 4 reference pictures.

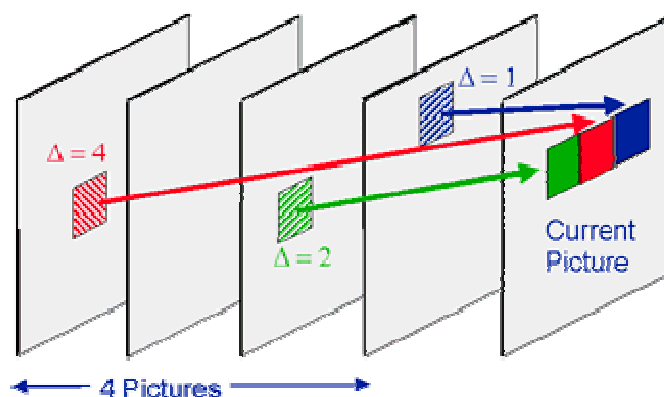


Figure 2.12 Motion compensated prediction with multiple reference frames [1]

2.3.3 Transform coding

There is high spatial redundancy among the prediction error signals. H.264 implements a block-based transform to reduce this spatial redundancy [1]. The former standards of MPEG-1 and MPEG-2 employed a two dimensional discrete cosine transform (DCT) for the purpose of transform coding of the size 8x8 [1]. H.264 uses integer transforms instead of the DCT. The size of these transforms is 4x4 [1]. The advantages of using a smaller block size in H.264 are stated as follows:

- The reduction in the transform size enables the encoder to better adapt the prediction error coding to the boundaries of the moving objects and to match the transform block size with the smallest block size of motion compensation.
- The smaller block size of the transform leads to a significant reduction in the ringing artifacts.
- The 4x4 integer transform has the benefit for removing the need for multiplications.

H.264 employs a hierarchical transform structure, in which the DC coefficients of neighboring 4x4 transforms for luma and chroma signals are grouped into 4x4 blocks (blocks -1, 16 and 17) and transformed again by the Hadamard transform as shown in Figure 2.13 (a). As shown in Figure 2.13 (b) the first transform H1 is applied to all samples of all prediction error blocks of the luminance component (Y) and for all blocks of chrominance components (Cb and Cr). For blocks with mostly flat pixel values, there is significant correlation among transform DC coefficients of neighboring blocks. Hence, the

standard specifies the 4x4 Hadamard transform (matrix H2 in Figure 2.13 (b)) for luma DC coefficients for 16x16 intra-mode only, and 2x2 Hadamard transform as shown in Figure 2.13 (b) (matrix H3) for chroma DC coefficients.

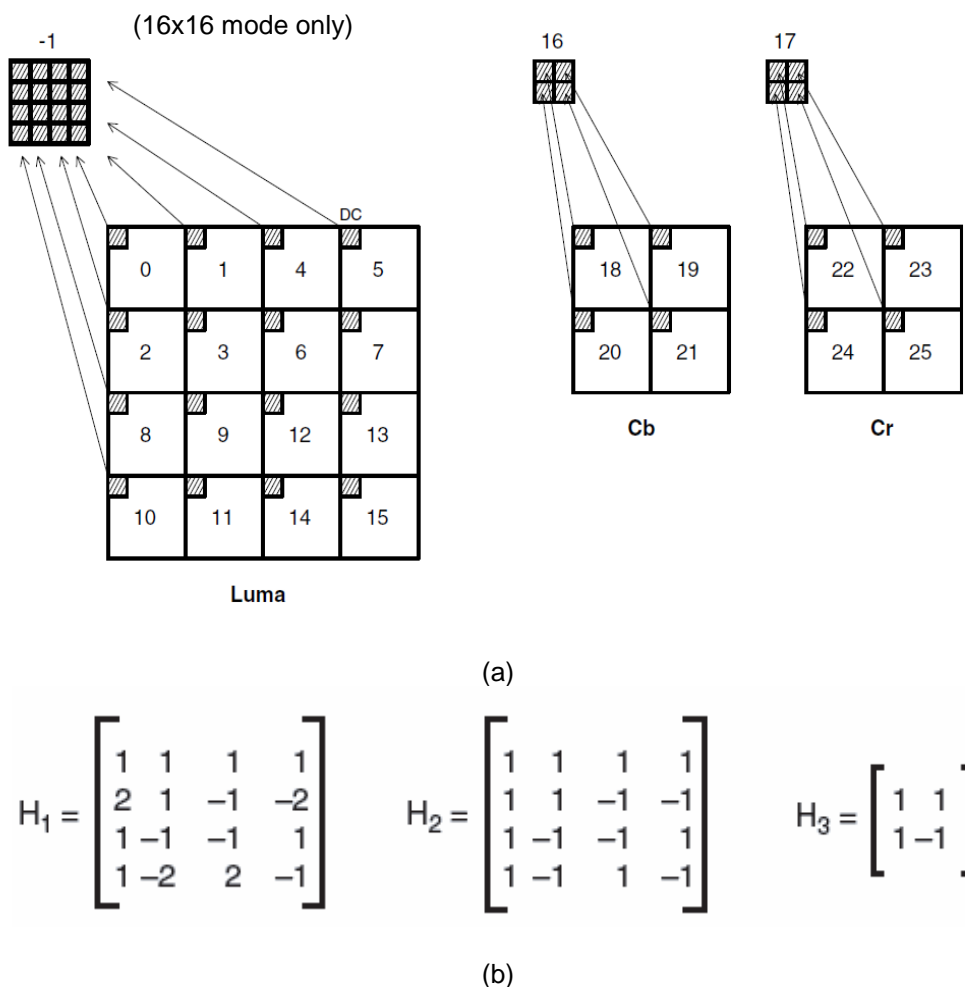


Figure 2.13 H.264 Transformation (a) DC coefficients of 16 4x4 luma blocks, 4 4x4 Cb and Cr blocks [1] (b) Matrices H1, H2 and H3 of the three transforms used in H.264 [32]

2.3.4 Deblocking filter

The deblocking filter is used to remove the blocking artifacts due to the block based encoding pattern. The transform applied after intra-prediction or inter-prediction is on blocks; the transform coefficients then undergo quantization. These block based operations are responsible for blocking artifacts which are removed by the in-loop deblocking filter. It reduces the artifacts at the block boundaries and prevents the propagation of accumulated noise. The presence of the filter however adds to the complexity of the system [1]. Figure 2.14 illustrates a macroblock with sixteen 4x4 sub blocks along with their boundaries.

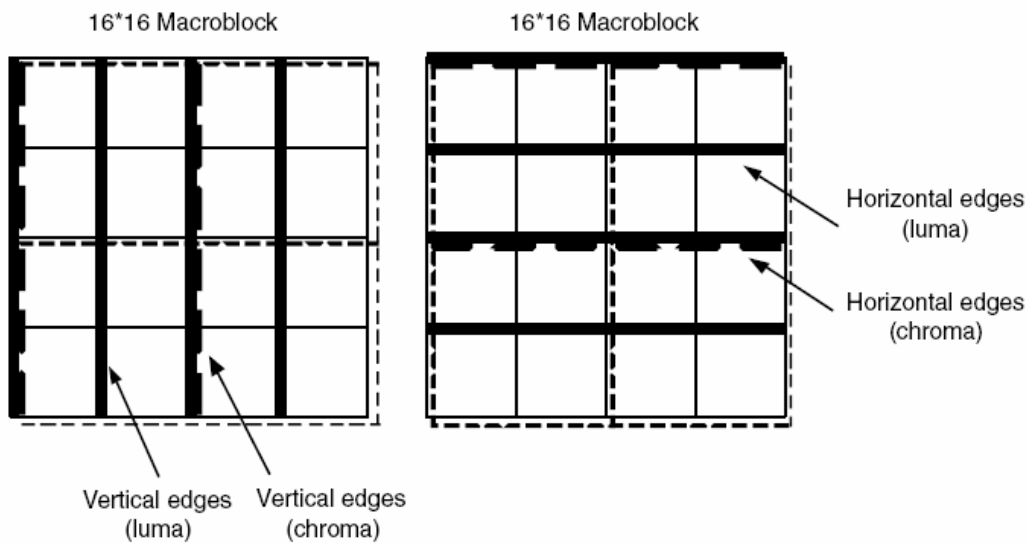


Figure 2.14 Boundaries in a macroblock to be filtered (luma boundaries shown with solid lines and chroma boundaries shown with dotted lines) [1]

As shown in the Figure 2.14, the luma deblocking filter process is performed on the 16 sample edges – shown by solid lines. The chroma deblocking filter process is performed on 8 sample edges – shown in dotted lines.

H.264 employs the deblocking process adaptively at the following three levels:

- At the slice level, the global filtering strength is adjusted to the individual characteristics of the video sequence
- At the block-edge level, a deblocking filter decision is based on inter or intra prediction of the block, motion differences, and the presence of coded residuals in the two participating blocks.
- At the sample level, it is important to distinguish between the blocking artifact and the true edges of the image. True edges should not be de-blocked. Hence a decision for deblocking at a sample level becomes important.

2.3.5 Entropy Coding

H.264 uses variable length coding to match a symbol to a code based on the context characteristics. All the syntax elements except for the residual data are encoded by

the Exp- Golomb codes [1]. The residual data is encoded using CAVLC. The main and the high profiles of H.264 use CABAC.

- Context-based adaptive variable length coding (CAVLC):

After undergoing transform and quantization the probability that the level of coefficients is 0 or +1 is very high [1]. CAVLC handles these values differently. It codes the number of 0s and +1. For other values, their values are coded.

- Context-based adaptive binary arithmetic coding (CABAC):

This technique uses the arithmetic encoding to achieve good compression. The schematic for CABAC is shown in Figure 2.15.

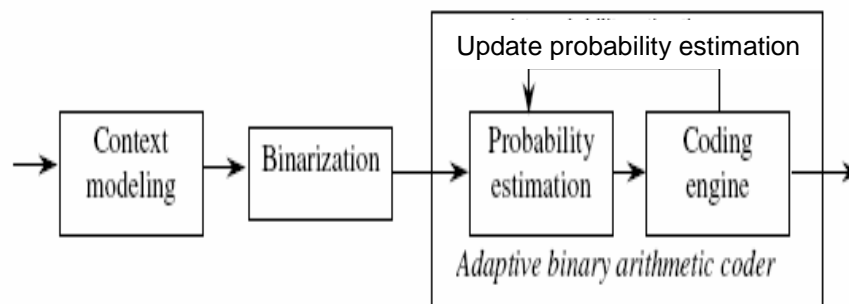


Figure 2.15 Schematic block diagram of CABAC [1]

CABAC consists of three steps:

- Step 1: Binarization: A non-binary value is uniquely mapped to a binary sequence
- Step 2: Context modeling: A context model is a probability model for one or more elements of a binarized symbol. The probability model is selected such that corresponding choice may depend on previously encoded syntax elements.
- Step 3: Binary arithmetic coding: An arithmetic encoder encodes each element according to the selected probability model.

2.3.6 B-slices and adaptive weighted prediction

Bi-directional prediction, which uses both past and future frames for reference, can be very useful in improving the temporal prediction. Bi-directional prediction in H.264 uses multiple reference frames. Figure 2.16 show bidirectional prediction from multiple reference frames. The standards, before H.264, with B pictures use the bidirectional mode, with

limitation that it allows the combination of a previous and subsequent prediction signals. In the previous standards, one prediction signal is derived from subsequent inter-picture, another from a previous picture, the other from a linear averaged signal of two motion compensated prediction signals.

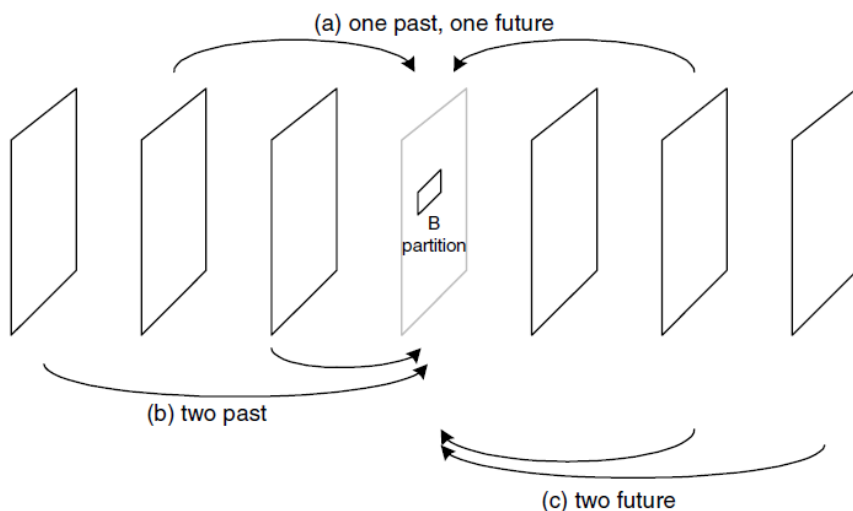


Figure 2.16 Partition prediction examples in a B macroblock type: (a) past/future, (b) past, (c) future [1]

H.264 supports forward/backward prediction pair and also supports forward/forward and backward/backward prediction pair [1]. Figure 2.16 (a) and Figure 2.16 (b) describe the scenario where bidirectional prediction and multiple reference frames respectively are applied and a macroblock is thereby predicted as a linear combination of multiple reference signals using weights as described in equation 2.1. Considering two forward references for prediction is beneficial for motion compensated prediction of a region just before scene change. Considering two backward reference frames is beneficial for frames just after scene change. H.264 also allows bi-directionally predictive-coded slice which may also be used as references for inter-coding of other pictures. Except H.264, all the existing standards consider equal weights for reference pictures. Equal weights of reference signals are averaged and the prediction signal is obtained. H.264 also uses weighted prediction [1]. It can be used for a macroblock of P slice or B slice. Different weights can be assigned to two different reference signals and the prediction signal is calculated as follows:

$$p = w1 * r1 + w2 * r2 \quad (2.1)$$

In (2.1), p is the prediction signal, $r1$ and $r2$ are the reference signals and $w1$ and $w2$ are the prediction weights. (Note * - simple multiplication)

2.4 H.264 Decoder

The H.264 decoder works similar in operation to the local decoder of H.264 encoder. An encoded bit stream is the input to the decoder. Entropy decoding (CABAC or CAVLC) takes place on the bit stream to obtain the transform coefficients. These coefficients are then inverse scanned and inverse quantized. This gives residual block data in the transform domain. Inverse transform is performed to obtain the data in the pixel domain. The resulting output is 4x4 blocks of residual signal. Depending on inter predicted or intra-predicted, an appropriate prediction signal is added to the residual signal. For an inter-coded block, a prediction block is constructed depending on the motion vectors, reference frames and previously decoded pictures. This prediction block is added to the residual block to reconstruct the video frames. These reconstructed frames then undergo deblocking before they are stored for future use for prediction or being displayed. Figure 2.17 illustrates the decoder.

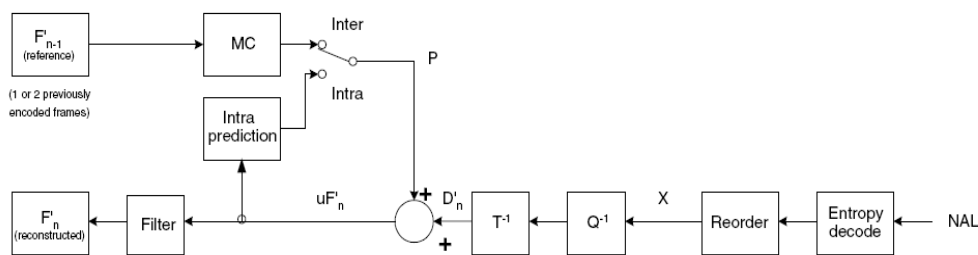


Figure 2.17 H.264 Decoder block diagram [1]

2.5 Summary

This chapter outlines the coding tools of H.264 codec. The next chapter describes the coding tools used in VC-1.

CHAPTER 3

VC-1 VIDEO CODING STANDARD

3.1 Introduction

VC-1 [40] [42] is the informal name of the SMPTE 421M video codec standard initially developed by Microsoft. It was released on April 3, 2006 by SMPTE. It is now a supported standard for blu-ray discs, and Windows media video 9 (WMV9). VC-1 is an evolution of the conventional DCT-based video codec design also found in H.261 [76], H.263 [77], MPEG-1 [75] and MPEG-2 [74]. It is widely characterized as an alternative to the latest ITU-T and MPEG video codec standard known as H.264/MPEG-4 AVC [14]. VC-1 contains coding tools for interlaced video sequences as well as progressive encoding. The main goal of VC-1 development and standardization is to support the compression of interlaced content without first converting it to progressive, making it more attractive to broadcast and video industry professionals.

The VC-1 codec is designed to achieve state-of-the-art compressed video quality at bit rates that may range from very low to very high. The codec can easily handle 1920 pixel × 1080 pixel resolution at 6 to 30 megabits per second (Mbps) for high-definition video. VC-1 is capable of higher resolutions such as 2048 pixels × 1536 pixels for digital cinema, and of a maximum bit rate of 135 Mbps. An example of very low bit rate video would be 160 pixel × 120 pixel resolution at 10 kilobits per second (Kbps) for modem applications.

The basic functionality of VC-1 [42] involves a block-based motion compensation and spatial transform scheme similar to that used in other video compression standards such as MPEG-1 and H.261. However, VC-1 includes a number of innovations and optimizations that make it distinct from the basic compression scheme, resulting in excellent quality and efficiency. VC-1 advanced profile is also transport independent. This provides even greater flexibility for device manufacturers and content services.

3.2 Profiles and levels in VC-1

3.2.1 Profiles in VC-1

Profiles of a video codec define a subset of tools and algorithms (use of bi-predictive pictures, start codes, intensity compensation etc) used, and levels within a profile place constraints on the parameters that define a particular profile. VC-1 defines three profiles, as listed below

1. The Simple profile targets low-rate internet streaming and low-complexity applications such as mobile communications, or playback of media in personal digital assistants. There are two levels in this profile.
2. The Main profile targets high-rate internet applications such as streaming, movie delivery via IP, or TV/VOD over IP. This profile contains three levels.
3. The Advanced profile targets broadcast applications, such as digital TV, HD DVD for PC playback, or HDTV. It is the only profile that supports interlaced content. In addition, this profile contains the required syntax elements to transmit video bit streams. This profile contains five levels.

Table 3.1 lists all the profiles and levels, and the label associated to each of them and Table 3.2 outlines the features in the different profiles in VC-1.

Table 3.1 List of profiles and levels in VC-1 [10]

Profile	Level	Label
Simple	Low	SP@LL
	Medium	SP@ML
Main	Low	MP@LL
	Medium	MP@ML
	High	MP@HL
Advanced	L0	AP@L0
	L1	AP@L1
	L2	AP@L2
	L3	AP@L3
	L4	AP@L4

Table 3.2 Features in VC-1 profiles [10]

Compression Feature	Simple	Main	Advanced
Baseline intra frame compression	X	X	X
Variable-sized transform	X	X	X
16-bit transform	X	X	X
Overlapped transform	X	X	X
4 motion vectors per macroblock	X	X	X
Quarter-pixel motion compensation Y	X	X	X
Quarter-pixel motion compensation U, V		X	X
Start codes		X	X
Extended motion vectors		X	X
Loop filter		X	X
Dynamic resolution change		X	X
Adaptive macroblock quantization		X	X
Bidirectional (B) frames		X	X
Intensity compensation		X	X
Range adjustment		X	X
Interlace: Field and frame coding modes			X
Self descriptive fields / flags			X
GOP layer			X
Display metadata			X

3.2.2 Levels in VC-1

There are several levels for each of the profiles. Each level limits the video resolution, frame rate, Hypothetical Reference Decoder bit rate (HRD), HRD buffer requirements, and the motion vector range. These limitations are shown in Table 3.3

Table 3.3 Levels in VC-1 [41]

Profile	Level	Maximum Bit Rate	Resolutions by Framerate
Simple	Low	96 kbit/s	176 x 144 / 15 (QCIF)
	Medium	384 kbit/s	240 x 176 / 30 352 x 288 / 15 (CIF)
Main	Low	2 Mbit/s	320 x 240 / 24 (QVGA)
	Medium	10 Mbit/s	720 x 480 / 30 (480p) 720 x 576 / 25 (576p)
	High	20 Mbit/s	1920 x 1080 / 30 (1080p)
Advanced	L0	2 Mbit/s	352 x 288 / 30 (CIF)
	L1	10 Mbit/s	720 x 480 / 30 (NTSC-SD) 720 x 576 / 25 (PAL-SD)
	L2	20 Mbit/s	720 x 480 / 60 (480p) 1280 x 720 / 30 (720p)
	L3	45 Mbit/s	1920 x 1080 / 24 (1080p) 1920 x 1080 / 30 (1080i) 1280 x 720 / 60 (720p)
	L4	135 Mbit/s	1920 x 1080 / 60 (1080p) 2048 x 1536 / 24

3.3 VC-1 Codec structure

The internal color format for VC-1 is 8-bit 4:2:0. The codec uses a block-based motion compensation and spatial transform scheme which, at a high level, is similar to all popular video compression standards. VC-1 performs block-by-block motion compensation from the previous reconstructed frame using a two-dimensional parameter called the motion vector (MV) to signal spatial displacement. A prediction of the current block is formed by looking up a same-sized block in the previous reconstructed frame that is displaced from the current position by the motion vector. Subsequently, the displaced frame difference, or residual error, is computed as the difference between the actual block and its motion-compensated prediction. This residual error is transformed using a linear energy-compacting transform, then quantized and entropy coded.

On the decoder side, quantized transform coefficients are entropy decoded, de-quantized and inverse transformed to produce an approximation of the residual error, which is then added to the motion-compensated prediction to generate the reconstruction. The high level description of the encoder is shown in Figure 3.1.

VC-1 has intra (I), predicted (P) and bi-directionally predicted (B) frames. Intra frames are those which are coded independently and have no dependence on other frames. Predicted frames are frames that depend usually on one frame in the past. Decoding of a predicted frame can occur only after all reference frames prior to the current frame starting from the most-recent intra frame. B frames are frames that have two references - one in the temporal past and one in the temporal future. B frames are transmitted subsequent to their reference, which means that B frames are sent out of order to ensure that their references are available at the time of decoding. B frames in VC-1 are not used as a reference for subsequent frames. This places B frames outside of the decoding loop, allowing shortcuts to be taken during the decoding of B frames without drift or long-term visual artifacts. This definition of I, P and B frames holds for both progressive and interlaced sequences.

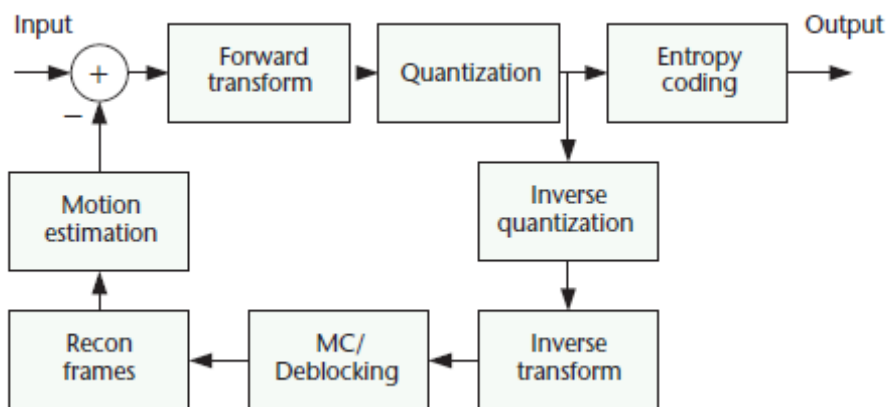


Figure 3.1 Block diagram of VC-1 encoder [42]

3.4 Coding concepts of VC-1

3.4.1 Color Space

VC-1 codes a sequence of images in the YUV 4:2:0 color space.

3.4.2 Macroblocks, Blocks, and Sub-blocks

When VC-1 codes an image, it divides the image into macroblocks. Each 16x16 macroblock is comprised of size 8x8 sample blocks (four Y blocks, one U block, and one V

block). Further, the coding method may divide an individual 8x8 block into two 8x4 blocks, two 4x8 blocks, or four 4x4 blocks.

3.4.3 Transform Coding

VC-1 uses a variation of the discrete cosine transform to convert blocks of samples into a transform domain to facilitate more efficient coding. The transform may operate on the full 8x8 block or any of the 3 supported sub-block sizes (8x4, 4x8, or 4x4). Unlike many codec standards preceding VC-1, the specification defines a bit-accurate transform method that all implementations are expected to conform to so as to minimize transform error. Figure 3.2 and Figure 3.3 illustrate the integer 8x8 and 4x4 inverse transforms used respectively.

$$T_8 = \begin{bmatrix} 12 & 12 & 12 & 12 & 12 & 12 & 12 & 12 \\ 16 & 15 & 9 & 4 & -4 & -9 & -15 & -16 \\ 16 & 6 & -6 & -16 & -16 & -6 & 6 & 16 \\ 15 & -4 & -16 & -9 & 9 & 16 & 4 & -15 \\ 12 & -12 & -12 & 12 & 12 & -12 & -12 & 12 \\ 9 & -16 & 4 & 15 & -15 & -4 & 16 & -9 \\ 6 & -16 & 16 & -6 & -6 & 16 & -16 & 6 \\ 4 & -9 & 15 & -16 & 16 & -15 & 9 & -4 \end{bmatrix}$$

Figure 3.2 8x8 Transform in VC-1 [79]

$$T_4 = \begin{bmatrix} 17 & 17 & 17 & 17 \\ 22 & 10 & -10 & -22 \\ 17 & -17 & -17 & 17 \\ 10 & -22 & 22 & -10 \end{bmatrix}$$

Figure 3.3 4x4 Transform in VC-1 [79]

3.4.4 Zigzag

After transforming sample data into the transform domain, VC-1 reorders the transformed data in a zigzag pattern which makes certain successive coding techniques

more effective. VC-1 has 13 different zigzag patterns depending on various parameters (block size, interlacing, prediction mode and intra/inter).

In VC-1, there are three different zigzag patterns used for Intra blocks depending on Alternating current (AC) prediction on/off and Direct current (DC) prediction direction as shown in Figure 3.4 – (a) Normal/ (b) Horizontal/ (c) Vertical scan types. However, there is only one scan pattern for each type of Inter blocks – one each for (a) 8x8 Inter, (b) 4x8 Inter, (c) 4x4 Inter, and (d) 8x4 Inter blocks as shown in Figure 3.5. There are more scan patterns defined in the VC-1 standard, depending on Profile/ Levels. Note that Intra DC is specially treated like in most video coding standards.

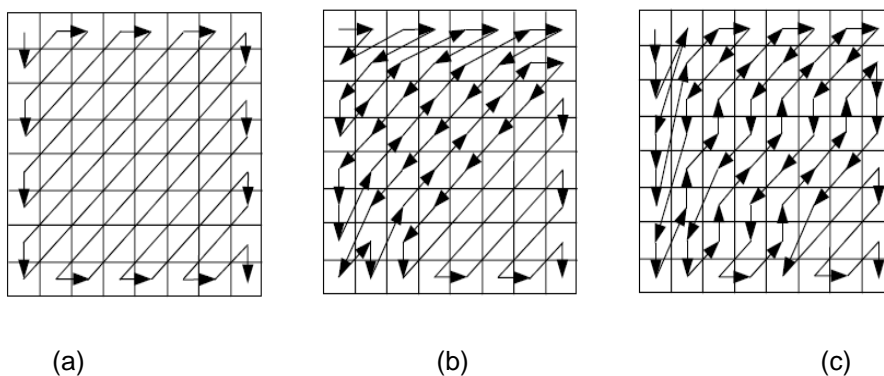


Figure 3.4 Intra Zig Zag scan in VC-1 (a) Normal (b) Horizontal (c) Vertical [79]

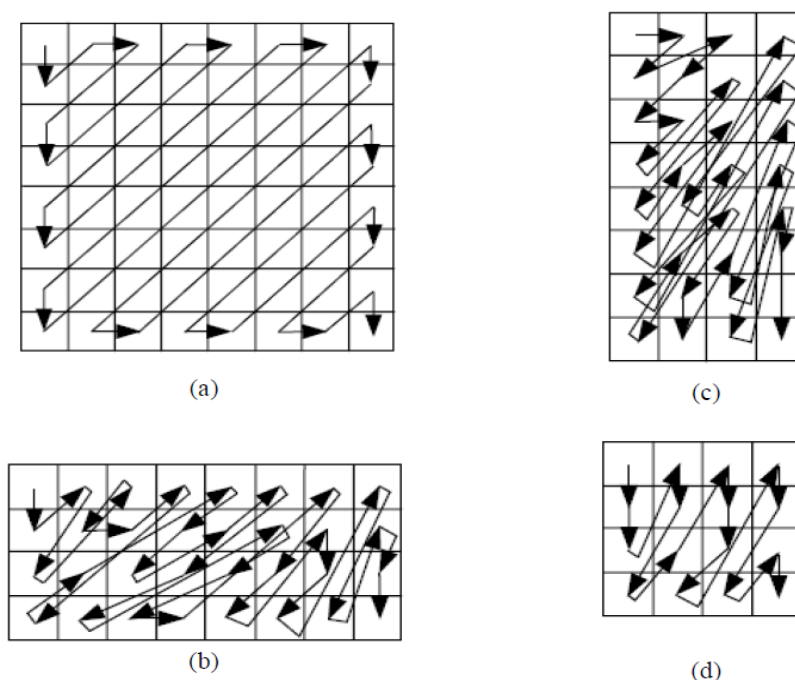


Figure 3.5 Inter Zig Zag scan in VC-1 (a) 8x8 (b) 4x8 (c) 8x4 (d) 4x4 [79]

3.4.5 Quantization

Quantization is the compression step that potentially loses the most information in a lossy compression scheme such as VC-1. This codec (unlike many others) defines a direct way to scale DC/AC coefficients using the quantization parameter instead of specifying quantization matrices.

Different macroblocks can use different quantization parameters (QP) in different ways - all macroblocks may have different quantization parameters, edge macroblocks only may have a different QP, two adjacent edges macroblocks may have a different QP, macroblocks from one particular edge may have a different QP or all macroblocks may have the same quantizer. For second to fourth case, there is a second quantizer for selected edge macroblocks, for the first case difference value between main and real quantizer is stored.

3.4.6 Bit plane Coding

VC-1 uses a number of bit planes which are simply maps of ones and zeros that specify properties for the macroblocks in an image. For example, a particular bit plane codes information about which macroblocks are not coded in a frame. These bit planes are coded into the final bitstream using a number of methods as follows:

- raw (data from bit plane is actually stored in macroblock header)
- row skip / column skip (each row or column are either zero - '0' bit is sent or coded - '1' bit and raw data bits are sent)
- tiling (bit plane is split into 2x3 or 3x2 blocks, each block is coded with own codeword, remainder is coded with row skip and column skip method)

Bit plane may be coded in inverted mode which is signaled by additional bit before bit plane data.

3.4.7 VC-1 Intra Prediction

Intra MBs are in I frames and optionally in the P/B frames of the VC-1 standard. VC-1 has no spatial intra prediction. Recent video compression standards take advantage of

Intra prediction methods such as DC/ AC prediction from adjacent blocks. The VC-1 has DC / AC prediction in the transform domain like in MPEG-4 part 2 [78]. DC prediction is always mandatory, while AC prediction is optional with AC prediction flag. Luma and chroma data perform independent Intra prediction as shown in Figure 3.6

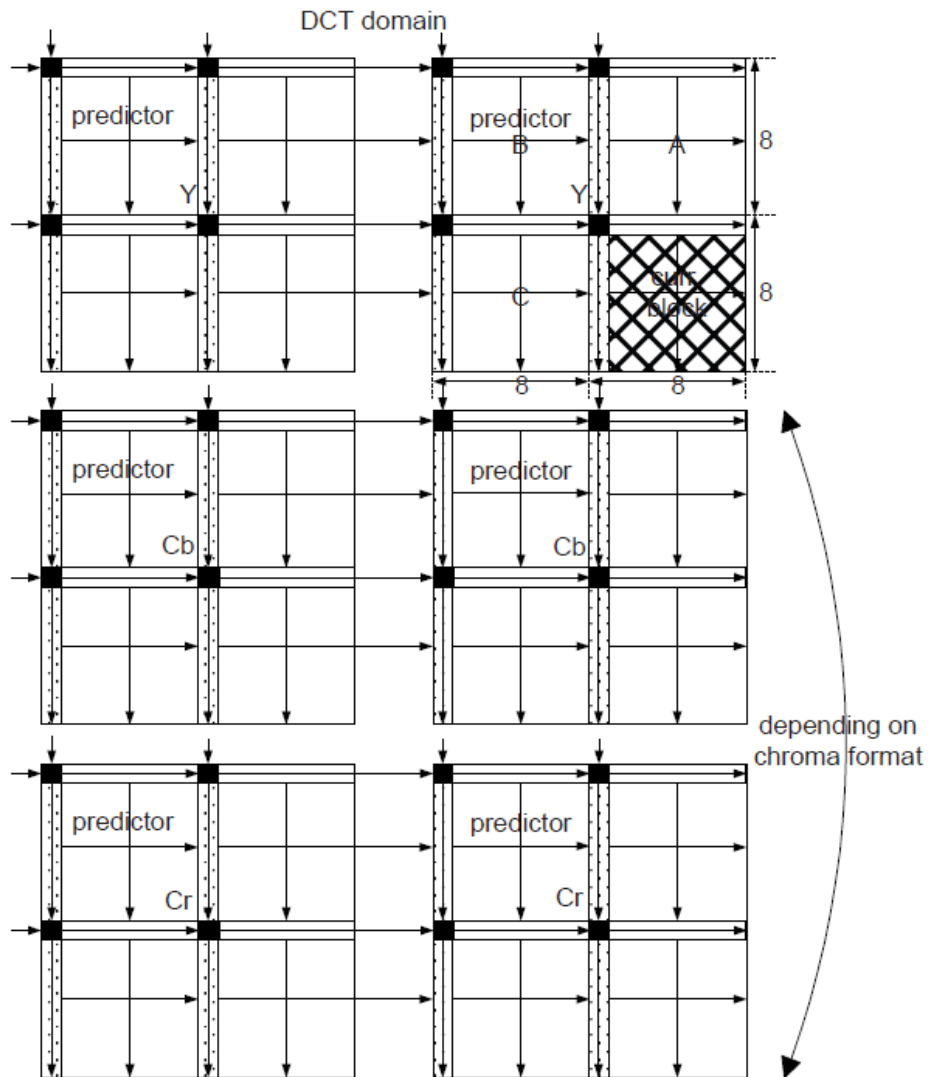


Figure 3.6 Intra AC/DC Prediction of 2 MBs in VC-1 [79]

3.4.8 Motion Compensation

As shown in Figure 3.7, a 16x16 luma can be broken into four 8x8 sub blocks for individual motion compensation (MC). The 8x8 size MC is pretty effective when the area to cover is not uniform in motion of texture such as in object boundaries.

For example, one of four 8x8 blocks in an object boundary MB can move in a different direction when it falls in the background with three other blocks occupied in the foreground. In such cases where motion is not uniform, smaller regions for MC can provide better performance in compression. This has been adopted in previous standards such as MPEG-4 Part 2. VC-1 moves forward one more step. Any of the 8x8 blocks in a MB can be coded as Intra mode as shown in Figure 3.7. For example, when a new pattern in the video is part of the frame now from the background (previously hidden) as the foreground content is moving, the Intra coding option can be considered for the area.

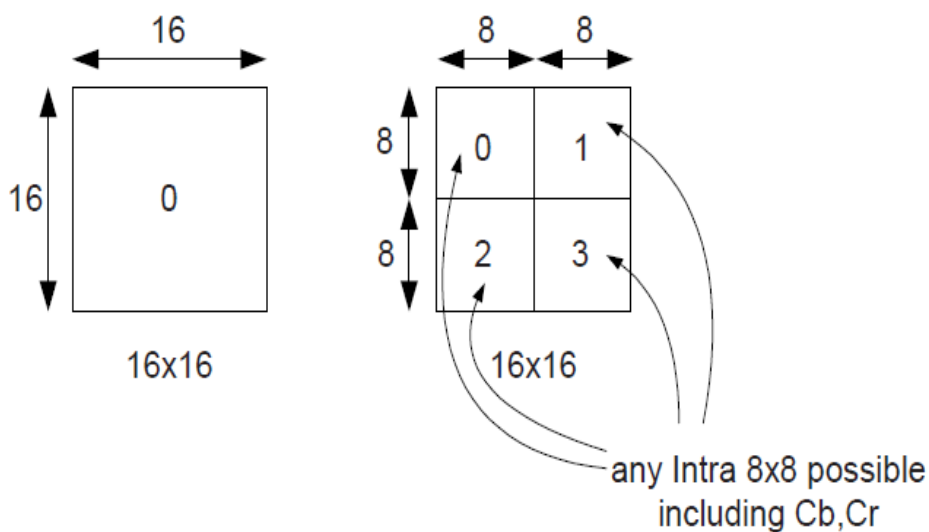


Figure 3.7 Partition for Motion Compensation in VC-1 [79]

VC-1 uses half-pel and quarter-pel inter frame motion compensation with either bilinear (like in H.264) or bicubic (extended version of motion compensation employed in Windows Media 2) interpolation. VC-1 proposes four ME methods as follows:

1. 16x16 block size (1MV) $\frac{1}{2}$ -pel bilinear,
2. 16x16 block size (1MV) $\frac{1}{2}$ -pel bi-cubic,
3. 16x16 block (1MV) $\frac{1}{4}$ -pel bi-cubic,
4. 8x8 block (4MV) $\frac{1}{4}$ -pel bi-cubic.

Note that the order of arrangement is in order of complexity and quality. Each application scenario can select a motion estimation (ME) method. For example, mobile handheld devices have relatively low computational power and typically do not need high quality

reproduction of video. In such a case, 16x16 block size ½-pel bilinear interpolation-MC is a good candidate for an encoder to select since this does not require high computational power and produces good quality. For HD-DVD, 8x8 block ¼-pel bi-cubic interpolation-MC might be a good choice since this would produce excellent quality through sophisticated predictors. When only ½-pel resolution is required, interpolation up to ½-pel is performed. When ¼-pel resolution is demanded, interpolation up to ¼-pel is performed. Figure 3.8 illustrates the integer, half and quarter pel positions used in VC-1.

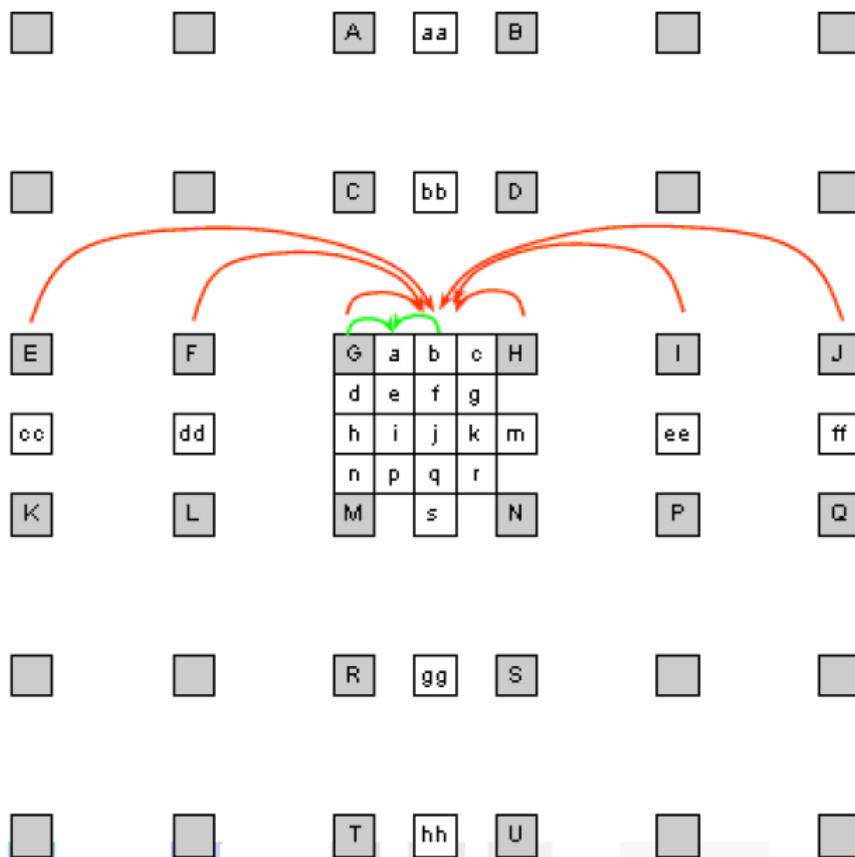


Figure 3.8 Integer, half and quarter pel positions [42]
(A-Q Integer, aa-hh half, a-s quarter pel positions)

3.4.9 Huffman Coding

All essential data in frames (like motion vectors, block coefficients) are encoded using static Huffman codes. Usually there are several sets of codes for each data type such as motion vectors, block coefficients. One set of codes is used throughout the whole frame.

The set index is usually defined in the frame header or derived from some parameters (like quantization or frame intra/inter).

3.4.10 Intensity Compensation

Intensity compensation is a special mode when the reference frame luma and chroma data are scaled before using it for motion compensation. When video scenes fade in and out, the content and textures are typically the same without movement. In this case, mainly the magnitude of luminance increases or decreases. Figure 3.9 illustrates this scenario. The lower part of the figure means that the scene is fading out. To effectively handle fade-in and fade-out scenes, the Intensity Motion Compensation (IMC) technique is used in VC-1. The key idea is to map luma and chroma data of the reference into an intensity-decreased or intensity-increased domain of the reference data as shown in Figure 3.9. If a typical motion compensation is applied, pretty big residual data would be produced as shown in Figure 3.9. However, if the same motion compensation is applied for a remapped reference picture, almost minimal amounts of residual data are produced. This technique is only defined for P pictures in VC-1. B pictures are not considered for this tool.

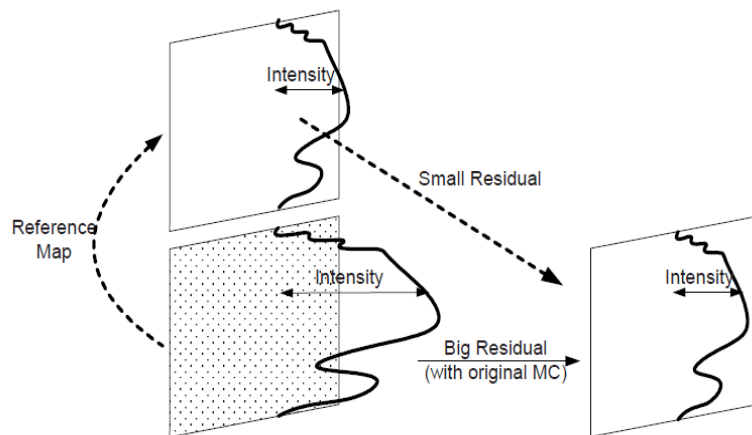


Figure 3.9 Intensity compensation in VC-1 [79]

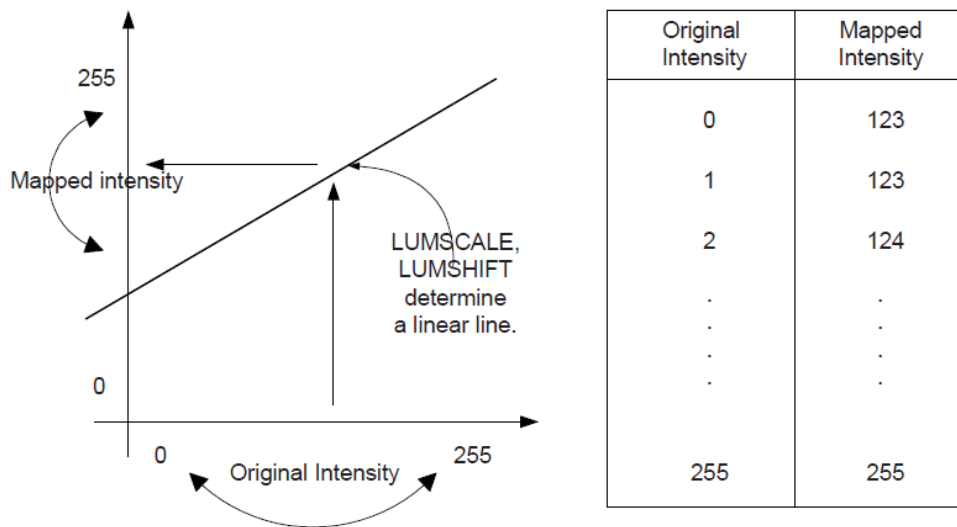


Figure 3.10 Intensity mapping and look up table [79]

3.4.11 Range Reduction

Range reduction is used when the values of luma and chroma are neither too low nor too high. The luma and chroma data range (0-255) is scaled down twice to 64-192 (with center = 128), and is expanded back before displaying (used for prediction in simple and main profiles).

3.4.12 Overlap transform smoothing

There are two techniques used in VC-1 to eliminate blocky effects – Overlapped Transform (OLT) smoothing and in loop filtering (ILF). To clearly tell the difference between the two operations in VC-1, the term deblocking filter (DBF) is defined to refer to OLT and ILF together as one operation. There are three scenarios to tackle in VC-1 deblocking operation. High-low quality discontinuity as shown in Figure 3.11 (a) is enhanced with the OLT tool, while low-low quality discontinuity and high-high quality discontinuity as shown in Figure 3.11 (b) and Figure 3.11 (c) are enhanced with the ILF tool. Note that this kind of discontinuity comes from different quantization effects at neighboring blocks in different frequency components.

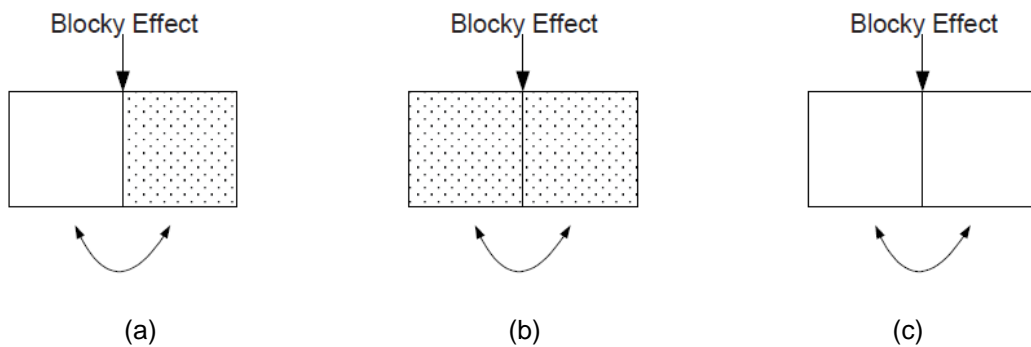


Figure 3.11 Blocky effect due to quality discontinuity [79] (a) High quality – Low quality (b) Low quality – Low quality (c) High quality – High quality

The basic idea of the OLT is to switch the edge data of two adjacent blocks, where both of them are of original quality as shown in Figure 3.12. When two such adjacent blocks undergo Transform/Quantization and inverse Quantization/ inverse Transform, quantization error and/or blocky effects can be introduced in one block more severely than the other in certain cases. At the decoder, two edge data should be switched over again due to recovering original data topology. Then, a good quality block contains bad quality edges, while a bad quality block contains good quality edges. In other words, good quality and bad quality blocks diffuse each other.

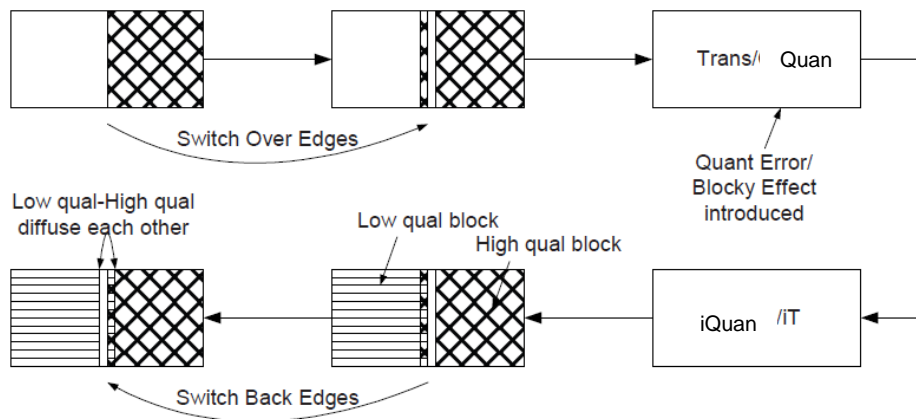


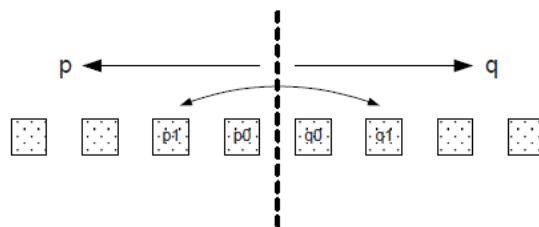
Figure 3.12 Concept of overlapped transform smoothing [79]

When the idea of Figure 3.12 is directly applied, high frequency components can be introduced due to edge exchange. Therefore, a filtering operation is defined as an OLT instead of simple data switch-over. The filtering operation required at the decoder is described in Figure 3.13; the inverse matrix of Figure 3.13 should be applied at the encoder

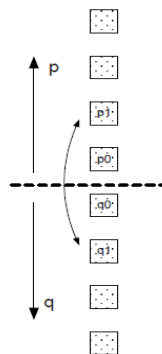
side. Note that the matrix is implementing a kind of low pass filter distributing original data around the edges of two adjacent blocks as was explained earlier. If some texture is lost in one block in the quantization, missing texture can be gained back due to inverse distribution operation at the decoder. When data is exchanged, Intra block and Inter block should not be exchanged (i.e., Inter is about residual data). Therefore, OLT is applied only to Intra coded blocks, which is always 8x8 size. In addition, when the data of two blocks is almost saturated at 255, filtering might introduce overflow due to the linear property of the operation. To avoid overflow, 128 level shift is defined for the OLT in the standard. Figure 3.14 illustrates the edge pixels used in the overlap transform.

$$\begin{bmatrix} p1' \\ p0' \\ q0' \\ q1' \end{bmatrix} = \begin{bmatrix} 7 & 0 & 0 & 1 \\ -1 & 7 & 1 & 1 \\ 1 & 1 & 7 & -1 \\ 1 & 0 & 0 & 7 \end{bmatrix} \begin{bmatrix} p1 \\ p0 \\ q0 \\ q1 \end{bmatrix} + \begin{bmatrix} r_0 \\ r_1 \\ r_0 \\ r_1 \end{bmatrix} \gg 3$$

Figure 3.13 Over lapped transform [79]



(a) Vertical boundary



(b) Horizontal boundary

Figure 3.14 Definition of edge pixels in OLT in VC-1 [79]

3.4.13 In loop filtering

For I or B pictures, in loop filtering (ILF) is performed at all 8x8 block boundaries. All the horizontal boundary lines in the frame shall be filtered first, followed by the vertical boundary lines. For P pictures, blocks may be Intra or Inter-coded. Intra-coded blocks shall use an 8x8 inverse transform to reconstruct the samples, whereas Inter-coded blocks shall use an 8x8, 8x4, 4x8 or 4x4 inverse transform. The boundary between transform blocks or sub-blocks shall be filtered, unless the following exception holds: when the transform blocks (or sub-blocks) on either side of the boundary are both inter-coded, and when the MVs of these blocks (or sub-blocks) are identical, and when both blocks (or sub-blocks) have all transform coefficients equal to zero, filtering will not be performed. The reason for not filtering in this case is due to copying over an already filtered reference picture as explained in Figure 3.15 Typical boundary selection in P pictures for ILF is depicted in Figure 3.16.

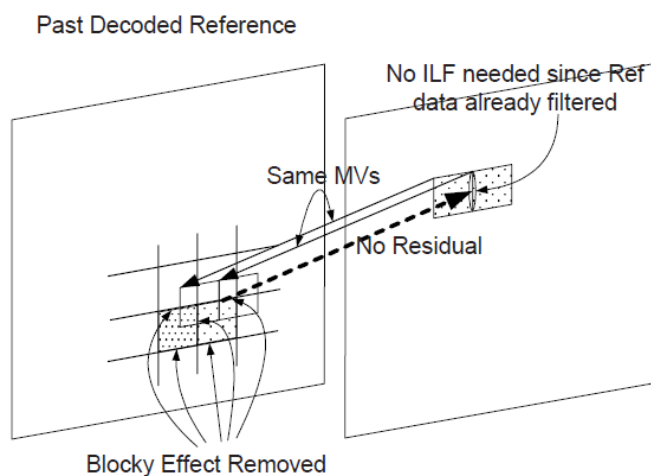


Figure 3.15 Filtering exception of in loop filtering

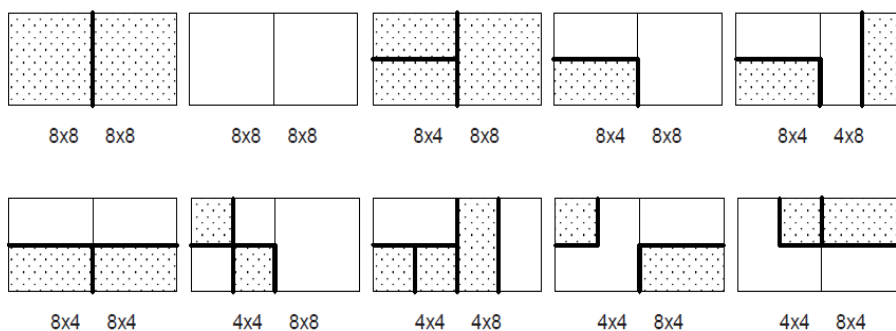
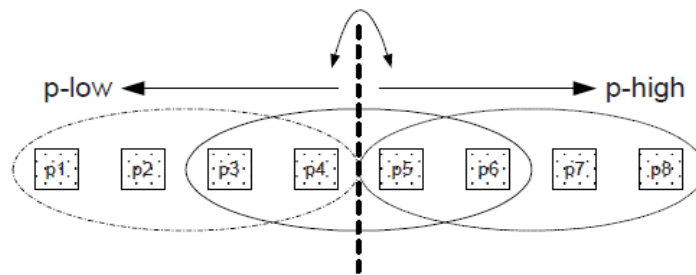
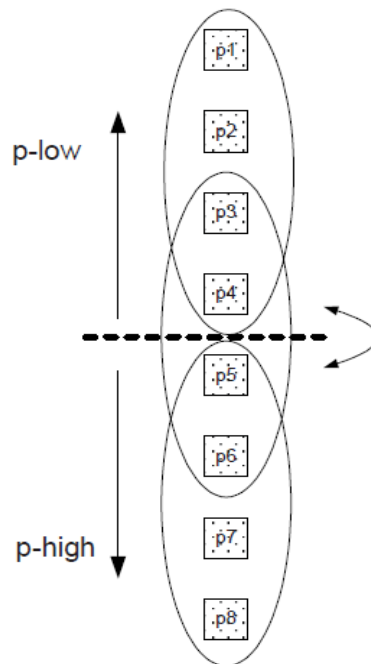


Figure 3.16 Typical block boundaries in P frames for In loop filtering in VC-1 [79]

A filtering algorithm is applied with inputs $p1$, $p0$, $q0$, $q1$, producing smoothed outputs $p0'$ and $q0'$. Only two pixels are adjusted for progressive video. The filtering algorithm is composed of three sub-steps – 1st sub-step of testing whether the video input is originally blocky looking, 2nd sub-step of comparing activities to see if the blocky effect on the edge is not serious compared with other two sides, and 3rd sub-step of actual adjustment at $p0$ and $q0$ when the filtering decision is made.



(a) Vertical boundary



(b) Horizontal boundary

Figure 3.17 Definition of edge pixels in in loop filtering [79]

3.4.14 Intra and Inter macroblock encoding process

Figure 3.18 and Figure 3.19 illustrates the basic steps used for encoding an intra and an inter macroblock respectively. The compression process uses block-based motion predictive coding to reduce temporal redundancy and transform coding to reduce spatial redundancy.

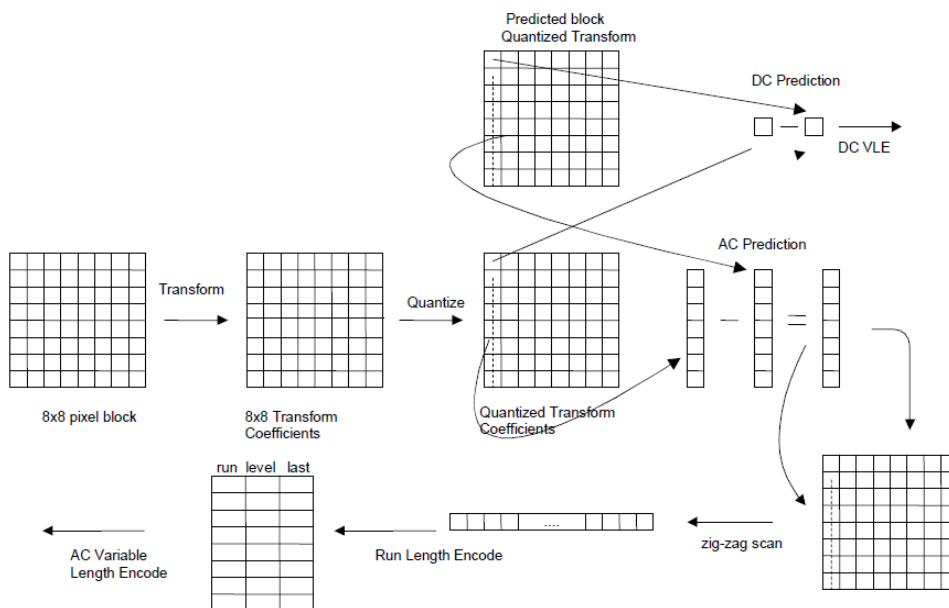


Figure 3.18 Coding of Intra blocks [10]

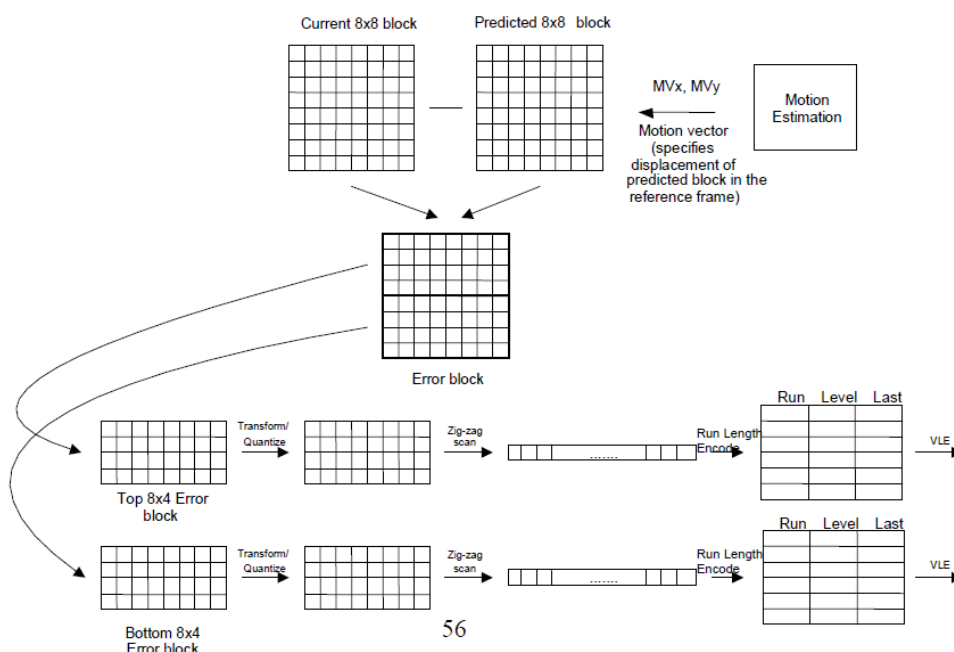


Figure 3.19 Coding of inter blocks [10]

3.5 Innovations in VC-1

The important innovations that distinguish VC-1 from prior standards are:

- Adaptive block size transform,
- 16-bit implementation of the transform,
- Multiple precision modes for motion compensation,
- Uniform and non-uniform quantization,
- Loop-filtering, and
- Overlap smoothing.

3.5.1 Adaptive block size transform

Traditionally, 8x8 transforms have been used for image and video coding [45] [46]. The 8x8 size has the advantage of being dyadic, and large enough to capture and preserve trends and periodic structures. However, it is known that smaller transforms are better in areas with discontinuities because they produce fewer ringing artifacts [43] [44].

VC-1 takes the approach of allowing 8x8 blocks to be encoded using either 8x8, 8x4, 4x8 or 4x4 transforms as shown in Figure 3.21. This allows the VC-1 codec to choose the transform size and shape that is best suited for the underlying data. The specific transform configuration used must be signaled as part of the bitstream. This signaling is performed in an efficient manner as well, as outlined below.

The transform type can be signaled at the frame or at the macroblock or block level to optimize overhead for the bit rate and content type. If the signal is sent at the frame level, all blocks within the frame use the same transform type. Likewise, if the signal is sent at the macroblock level, all blocks within a macroblock (there are six 8x8 blocks in all, including four luminance and two chrominance blocks) use the same transform type.

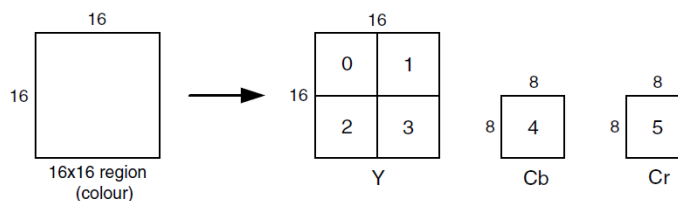


Figure 3.20 Macroblock (4:2:0) [1]

Block level signaling is specific to the current block. Macroblock and block level signaling can be mixed across macroblocks within one frame. This allows for coarse and fine level specification of the transform type, which is useful when the data is non-stationary. Frame level signaling helps in low-rate situations where the transform type coding overhead may be excessive.

When macroblock or block level signaling is used there is a possibility of saving a few bits in static or perfectly predicted areas. When for the chosen transform type all quantized transform coefficients over a macroblock or block are zero, there is no need to send the transform type information, since all varieties of inverse transform will produce all-zero blocks for an all-zero input. This allows the overhead to be reduced for static areas or areas that can be generated purely by motion compensation, and is a key factor for improved performance at low rates or for low-motion sequences such as talking heads.

Intra frames and intra blocks/macroblocks in predicted frames use 8x8 transforms. The use of adaptive block transform achieves significant rate-distortion benefits. More importantly, from the point of subtle texture preservation (such as keeping film details and grain noise), adaptive transform sizes also provide significant subjective quality benefits [47] [48].

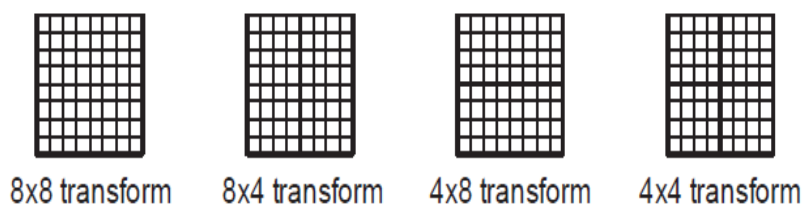


Figure 3.21 Transform sizes allowed in VC-1 [42]

3.5.2 16-Bit Transforms

The important motivation for the transform design is to minimize the computational complexity of the decoder while preserving the compression efficiency. The VC-1 inverse transforms are designed to be implemented in 16-bit fixed point arithmetic which facilitates software as well as hardware realizations. It is well known that 32 bit operations are expensive to implement on 16 bit processors which form a large percentage of commonly

used DSPs. Thus, the computational complexity of the decoder is significantly reduced, if the inverse transform can be implemented exactly in 16-bit fixed point arithmetic. The inverse transform requiring 32-bit integer or floating point arithmetic is more complicated.

At the same time, the transform approximates the DCT [18], and thus retains the favorable energy compaction properties of the DCT on intra and residual video data. The VC-1 transforms are designed to meet a list of constraints enumerated below. The transforms are separable, which allows the constraints to be defined for each one dimensional transform stage. The constraints for both the one dimensional 4 and 8 point transforms are:

- a) Transform coefficients are small integers.
- b) The transform is a 16 bit operation – where both sums and products of two 16 bit values produce results within 16 bits.
- c) Forward and inverse transforms form an orthogonal pair.
- d) The transform approximates a DCT.
- e) Norms of basis functions within one transform type are identical so as to eliminate the need for any coefficient-indexed renormalization in the de-quantization process.
- f) Norms of basis functions between transform types are identical. In other words the 4 and 8 point basis functions have the same norm. This allows the use of same quantization parameter between various transform types while maximizing the rate-distortion performance.

It is impossible to find an 8x8 transform that meets all of these constraints simultaneously. The key innovation in VC-1 is to mildly relax constraints e and f, allowing the basis of the transform coefficients to be very close, though not identical, in norm. These small discrepancies between basis function norms are accounted for entirely on the encoder side with no loss in compression efficiency.

Although the 4 and 8 point transforms (Figure 3.2 and Figure 3.3 respectively) are essentially described by the transform matrices, their two-dimensional implementations have additional restrictions. First, the rows of the de-quantized transform coefficients are

inverse transformed. This is followed by a rounding operation which is followed by inverse transformation of the columns, followed by another rounding operation. In order to operate within 16 bits with sufficient headroom and maximum accuracy, rounding is distributed between the horizontal and vertical transforms. In addition, the second transform stage exploits the specific transform matrix entries to achieve an extra bit of precision. The output of the inverse transform is in the range 10 bits which allows for headroom for quantization error beyond the theoretically possible 9 bits.

3.5.3 Motion Compensation

Efficiency of a video codec is closely related to the ability of the motion compensator to generate a good set of predictors. The quality of motion compensation is determined by three factors:

- 1) Sub-pixel resolution,
- 2) Size of predicted area and
- 3) Filters used for interpolation.

While sub-pixel resolution improves the quality of prediction, the benefits of going to finer pixel resolutions are offset by the increased cost of coding motion vectors to higher degrees of precision. At low rates, higher precision in motion vectors is a liability since the percentage of bits used in coding motion vectors is significant. Motion vector resolution is either $\frac{1}{2}$ or $\frac{1}{4}$ pixel in existing profiles in standard video codecs. VC-1 allows a maximum resolution of $\frac{1}{4}$ pixels.

The second factor influencing the ability to generate good predictors is the size of the predicted area. Typically in the older formats, a single motion vector is used for a macroblock, which is a 16x16 pixel area in the luminance plane. MPEG-4 allows definition of motion vectors for 16x16 or 8x8 blocks; this choice is made for each macroblock being coded. H.264/AVC permits motion vectors to reference areas as small as 4x4. While this level of freedom can be useful at high bit rates, smaller areas impose higher computational overhead on the codec. Smaller blocks with randomly distributed motion vectors cause increased cache access, and they need more filtering steps on a per-pixel basis. Thus

increase the computational complexity of the decoder. VC-1 uses 16x16 block size by default, but allows for 8x8 block size in frames which are signaled as containing mixed motion vector resolution.

Finally, the filter used for generating sub-pixel predictors is the third key determinant of the quality of motion compensation. Shorter filters are computationally simpler but have poor frequency response and are adversely influenced by noise. Longer filters using more pixels are computationally more difficult to implement. Moreover, images have strong local and transient characteristics that tend to get blurred with long filters. VC-1 trades off these considerations by using two sets of filters for motion compensation. The first is an approximate bicubic filter with four taps, and the second is a bilinear filter with two taps.

Further, VC-1 combines the motion vector modes derived from the three criteria (MV resolution, size of predicted area and filter type) into a single mode. This combined mode is one of the following:

- Mixed block size (16x16 and 8x8), $\frac{1}{4}$ pixel, bicubic – Mode 1
- 16x16, $\frac{1}{4}$ pixel, bicubic
- 16x16, $\frac{1}{2}$ pixel, bicubic
- 16x16, $\frac{1}{2}$ pixel, bilinear

This combined motion vector mode is signaled at the frame level. In general, higher bit rates tend to use the modes at the top of the list, and lower bit rates use modes towards the bottom of the list. Only when mode 1 is chosen, the predicted block size is allowed to be either 16x16 or 8x8, and the size is signaled at a macroblock level. In all other modes, only 16x16 block sizes are used. The consolidation of these three criteria into one leads to a more compact decoder implementation, with no significant performance loss.

3.5.3.1 Sub-pixel Filters

In the existing codec standards, sub-pixel interpolation in two dimensions is performed by filtering in one dimension, rounding and clamping the intermediate value back to the input range of 8 bits, followed by filtering in the second direction, rounding and clamping. It is possible to achieve additional accuracy by retaining a higher precision result after the first stage of filtering. Another advantage is that since the clamping operation is

non-linear it may be more difficult to implement on certain processors (though clamping can be done easily on hardware and specific DSPs).

VC-1 exploits this observation by defining the two dimensional filtering process as follows. First, filtering is performed in the vertical direction. Then, a rounding factor is added to the result and some bits are shifted out. The intermediate result is possibly outside of the [0 255] range. Next, the intermediate result is filtered in the horizontal direction. Finally, a second rounding parameter is added to the result which is shifted and clamped to within the range [0 255]. The two shifts are chosen so as to (a) add up to the required shift for normalizing the filters and (b) to allow for a 16 bit implementation – where the intermediate values in the second filtering operation are within 16 bits. The four-tap bicubic filters used in VC-1 for $\frac{1}{4}$ and $\frac{1}{2}$ pixel shifts are: $[-4\ 53\ 18\ -3]/64$ and $[-1\ 9\ 9\ -1]/16$ [42].

3.5.3.2 Chrominance Channel

Since chrominance motion vectors are implicitly derived from co-located luminance motion vectors, their accuracy is limited and offers scope for simplification. Also, the chroma channels have a strong low-pass component. VC-1 uses bilinear filters for chroma motion interpolation. In general, chroma motion vectors are obtained by dividing the co-located luminance motion vectors by 2 and rounding the result to a $\frac{1}{4}$ pixel position. In addition, there is a sequence level 1 bit field that controls the rounding of chroma motion vectors. If this bit is set, then the chroma motion vectors that are at quarter pixel offsets are rounded to the nearest full pixel positions – in effect only allowing $\frac{1}{2}$ and full pixel locations for chroma motion vectors. The purpose of this mode is speed optimization of the decoder.

The motivation for this optimization is the significant difference between the complexities of interpolating pixel offsets that are at a) integer pixel; b) half pel; c) at least one coordinate (of x and y) at a quarter pel; and d) both coordinates at quarter pel positions. The ratio of a:b:c:d is roughly 1:4:4.7:6.6. By applying this mode one can favor a) and b), thus cutting down on decoding time. Since this is done only for chroma interpolation, the coding and quality losses (especially subjective quality) are both negligible [42].

3.5.4 Quantization

Quantization and de-quantization of transform coefficients are key steps that can critically affect the rate-distortion performance of a video codec. Earlier standards such as MPEG-2 use quantization with a dead-zone, where all quantization intervals except the dead-zone are of the same size—the dead-zone being typically larger. The quantization intervals and reconstruction levels of a dead-zone quantizer are shown in Figure 3.22 a. The use of a dead-zone leads to substantial bit savings at low bit rates.

In regular uniform de-quantization, the reconstruction levels are all equally spaced. The quantization intervals and reconstruction levels of a regular uniform quantizer are shown in Figure 3.22 b. This kind of a quantizer performs better at higher bit rates.

VC-1 allows for the use of both dead-zone and regular uniform quantization. The type of quantizer is signaled at the frame level, and the appropriate de-quantization rule is applied to all coefficients within the frame by the decoder. The encoder typically uses dead-zone quantizer at larger step sizes, and uses uniform quantizer at lower step sizes. Although this rule works well across a variety of sequences, other factors such as the noise within a sequence and rate control parameters may be used in more sophisticated encoders to fine-tune the choice of quantizer type. The flexibility to switch between dead-zone and regular uniform quantization is a key factor in the superior performance of VC-1 at both low and high bit rates.

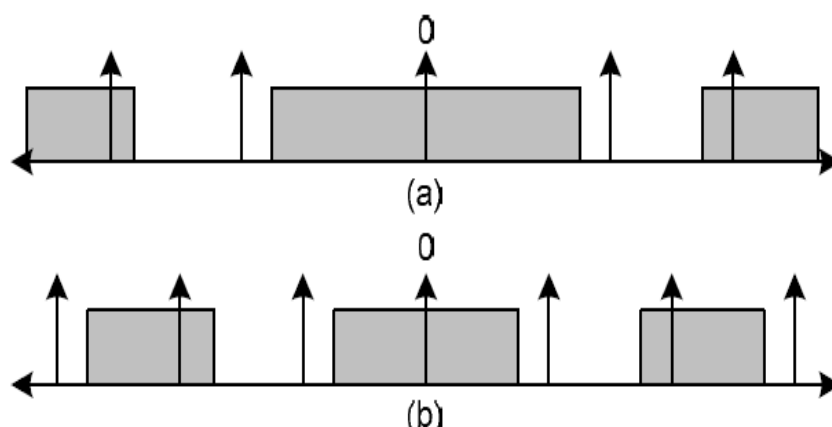


Figure 3.22 VC-1 quantization and dequantization rules showing (a) dead-zone and (b) regular uniform quantization – arrows are reconstruction levels, and gray boxes are recommended quantization bins (for alternate intervals) [42]

Differential quantization is an encoding method in which multiple quantization steps are used within a single frame. Rather than quantize the entire frame with a single quantization level, macroblocks are identified within the frame that might benefit from lower quantization levels and greater number of preserved AC coefficients. Such macroblocks are then encoded at lower quantization levels than the one used for the remaining macroblocks in the frame. The simplest and typically most efficient form of differential quantization involves only two quantizer levels (bi-level dquant), but VC-1 supports multiple levels, also.

3.5.5 Loop Filtering

Compression can induce discontinuities at block boundaries, and these discontinuities show up as visible 'blocky' artifacts. Moreover, these artifacts affect the quality of the reconstructed frame as a predictor for future frames. To mitigate these effects, the VC-1 scheme uses an in-loop deblocking filter. The filtering is performed on the reconstructed frame prior to its use as a reference frame for the subsequent frame(s). Therefore, the encoder and decoder must perform the same filtering operation.

In P pictures, the boundary between a block and a neighboring block is not filtered if both blocks have the same motion vector and if both blocks have no residual error. This prevents over-smoothing of block boundaries where quantization or motion compensation induced discontinuities are unlikely to occur. This constraint also lowers the complexity of the loop-filtering process in P pictures.

Due to the various condition checks involved, loop filtering is a computationally expensive process. VC-1 uses a shortcut to reduce computations. Determination of whether to smooth across an edge or not is made only once every four pixels. This shortcut helps speed up the loop filtering process with little rate-distortion or visual detriment.

3.5.6 Overlap transform

While loop-filtering can mitigate artifacts due to boundary discontinuities, it cannot distinguish between both block-aligned true edges and apparent (quantization induced) block edges. This drawback is the result of loop-filtering being purely a decoder (and

reconstruction loop on the encoder) process since there is no accounting for the fact that a deblocking filter is being applied to the forward process of encoding. Moreover, the loop-filter is a conditional non-linear operation, and hence may be disabled in the less complex profiles due to its computational requirements.

It is simpler to use a lapped transform to minimize block boundary artifacts in intra-coded macroblocks, while avoiding the drawbacks of loop-filtering. The use of lapped transform is motivated by the following observation: Intra coding in video codecs is performed by partitioning the picture into tiles, performing a linear transform, and quantizing the transform coefficients. At higher levels of quantization, fewer coefficients are quantized to non-zero values. Since the spatial support of the transform basis is restricted to the block (typically 8x8 pixels), the influence of any given non-zero coefficients is circumscribed by this support. This causes apparent edge artifacts at the block boundaries.

A lapped transform is a transform whose input spans, besides the data elements in the current block, a few adjacent elements in neighboring blocks. On the reconstruction side the inverse transform influences all data points in the current block as well as a few data points in neighboring blocks. In two dimensions, the lapped transform is a function of the current block, together with select elements of blocks to the left, top, right, bottom and possibly top-left, top-right, bottom-left and bottom-right. The number of data points in neighboring blocks that are used to compute the current transform is referred to as the overlap.

A spatial domain realization of the lapped transform is used in VC-1. The key advantage of the spatial domain realization of the lapped transform is that an existing block based codec can be retrofitted with a pre and post processing stage to derive the benefits of the lapped transform. The post-processing step is a linear smoothing filter applied to the inverse transform reconstruction, within the decode loop. The pre-processing step is the inverse of post-processing.

Certain critical design issues emerge when the spatial domain lapped transform is implemented by means of pre-processing and post-processing stages in an existing codec. The key issues are range expansion, need for higher precision arithmetic, and reduced

quality in high contrast regions. VC-1 handles these issues by a combination of techniques. First, the lapped transform is used only for a certain quantization and higher, i.e. at the lower bitrates where blocking artifacts are more apparent. Second, the pre and post-processing operations are not true inverses of each other – the post-processing smoothing operation is heavier than the pre-processing sharpening operation. Third, the range of intermediate data is clamped to 9 bits to prevent underflow or overflow.

Further, VC-1 allows for the signaling of the overlap smoothing at the macroblock level for an intermediate range of bit rates. Thus, overlap can be enabled in high texture areas and disabled in smooth regions providing additional perceptual benefits.

3.5.7 Interlaced Coding

Interlaced video content is widely used in television broadcasting. When encoding interlaced content, the VC-1 codec can take advantage of the characteristics of interlaced frames to improve compression. This is achieved by using data from both fields to predict motion compensation in interpolated frames.

3.5.8 Advanced B Frame Coding

A bi-directional or B frame is a frame that is interpolated from data both in previous and subsequent frames. B frames are distinct from I frames (also called key frames), which are encoded without reference to other frames. B frames are also distinct from P frames, which are interpolated from previous frames only. VC-1 includes several optimizations that make B frames more efficient. VC-1 does not have a fixed group of pictures (GOP) structure and the number of pictures in a GOP can vary.

3.5.9 Fading Compensation

Fading compensation is used by VC-1 to improve the performance of motion compensation on video sequences that include fading effects such as fade-to-black, fade-from-black and cross-fades. This tool helps in the generation of better predictors since luminance change cannot be modeled by motion compensation. Fading detection comprises computing an error measure for the current video image relative to the original reference

video image, and comparing the error measure with a threshold. If fading is detected, the encoder computes the fading parameters which specify a pixel-wise linear first order transform of the reference image. The fading parameters are quantized and signaled to the decoder. The encoder and decoder use the quantized fading parameters to transform the original reference frame into a new reference frame, which is used for motion compensation. This process allows motion compensation to find better predictors for each block, and thus code more blocks as inter-blocks. Thus fading compensation improves the overall compression efficiency of VC-1 on sequences with fading, or other global illumination changes.

3.6 Summary

This chapter outlines the coding tools used in VC-1. Having an understanding of H.264 from chapter two and VC-1 from this chapter, the next chapter compares the two standards and brings out the similarities and differences between the two.

CHAPTER 4

COMPARISON OF H.264 AND VC-1

4.1 Introduction

The high definition video adoption has been growing rapidly for the last five years. The high definition DVD format blue ray has mandated MPEG-2 [13] [16], H.264 [9] and VC-1 [10] as video compression formats. The coexistence of these different video coding standards creates a need for transcoding. As more and more end products use these standards, transcoding from one format to another adds value to the product's capabilities.

It is very important to be able to understand the difference between the two encoding standards (H.264 and VC-1) before the process of transcoding between these two standards is described. This chapter describes the comparison of performance between these two codecs for comparable profiles. The two profiles looked at are VC-1 simple profile and H.264 baseline profile.

4.2 Comparison of features and coding tool

H.264 [1] is a set of encoding tools to provide high quality video at low bit rates. A lot of encoding tools employed in order to achieve that include significant computation to reduce the bit rate. Hence H.264 is significantly computationally involved compared to other codecs. The main advantage that VC-1 standard on the other hand is that it can keep the encoding process as simple as possible, thereby achieving processor friendliness. VC-1 stresses the processor significantly less compared to H.264. As a result there is a lot of difference between different encoding tools employed in the two codecs. The Table 4.1 provides a high level overview of the difference between the various features of the two standards – VC-1 and H.264.

Table 4.1 Comparison of H.264 baseline and VC-1 simple profile tools

Feature	H.264 Baseline	VC-1 Simple
Picture type	I, P	I, P
Transform Size	4x4	4x4, 4x8, 8x4, 8x8
Transform	Integer DCT	Integer DCT
Intra Prediction	4x4, 16x16 spatial, IPCM	Frequency domain DC and AC Prediction
Motion Compensation Block Size	16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4	16x16, 8x8
Total MB Modes	7 inter + (9 + 4) intra	3
Motion Vector resolution	¼ pixel	¼ pixel
In loop filter	Deblocking	Deblocking, Overlap transform
Reference Frames	Single, Multiple	Single
Entropy coding	CAVLC	Adaptive VLC

For the P-frames VC-1 does not make use of multiple frames for motion estimation / motion compensation as compared to H.264 (refer Figure 2.12). Due to the presence of multiple prediction frames in H.264 the prediction frame buffers need to be larger. Also the presence of more than one prediction frame allows weighted prediction. VC-1 does not support weighted prediction. VC-1 needs only one frame buffers for prediction based on previous frame.

H.264 has up to 9 intra-prediction modes (Section 2.3.1). The intra-prediction modes in H.264 make use of adjacent pixel redundancy. Intra-prediction contributes to significant increase in complexity of H.264. There is no spatial intra prediction in VC-1; it only has a low cost DC and an AC prediction as noted earlier in Section 3.4.7. However VC-1 cannot achieve the significant redundancy reduction that H.264 achieves for intra-prediction. The intra-prediction modes in H.264 increase the complexity by roughly 2-16 times.

Inter-prediction in H.264 can be carried out for multiple MB and sub-MB sizes including 4x4, 4x8, 8x4, 8x8, 8x16, 16x8 and 16x16 (Section 2.3.2). This helps H.264 achieve much finer prediction thereby reducing the bit rate. VC-1 supports 16x16 and 8x8 MB sizes for motion estimation. A major amount of complexity in an encoder comes from motion estimation and compensation. Hence, lowering the complexity in the motion estimation process in VC-1 reduces the CPU utilization to a very large extent.

Both the codecs support up to 1/4 pixel accuracy for motion vectors. In order to generate sub-pixel values for sub-pixel motion search, interpolation filters are required. Whereas H.264 used 6-tap filtering (refer Figure 2.10 and Figure 2.11) for this VC-1 makes use of bilinear and bi-cubic filters which give better subjective quality.

H.264 has an in-loop deblocking filter (Section 2.3.4). VC-1 has an in loop deblocking filter that is less complex compared to H.264. In H.264, deblocking decision is taken for all 4 sets of pixels for every 4x4 boundary (refer Figure 2.14). In VC-1, the complexity is reduced to one single decision per boundary. Also, H.264 filters up to 6 pixels in each set of 8 pixels, where as VC-1 filters only 2 pixels in each set of 8 pixels.

The DCT used in H.264 is an integer DCT. This DCT is described in Section 2.3.3. VC-1 uses a different integer DCT which is adaptive and aims to club as many zero coefficients together as possible. To achieve this VC-1 makes use of different scan ordering for inter and intra. These processes are described in Section 3.4.3 and 3.4.4. Whereas for most of the profiles, H.264 has only 4x4 integer DCTs, VC-1 has 8x8, 8x4, 4x8 and 4x4 integer DCT in all profiles.

4.3 Performance comparison

It is well understood that the newer video bit-stream formats such as VC-1 and H.264 allow for superior compression performance (in terms of rate-distortion plots) compared with the older formats such as MPEG-2 [74] and MPEG-4 part 2 (visual) [78]. The more interesting question that remains to be answered is the relative performance of VC-1 versus H.264. In general, video formats are difficult to compare because of several reasons. Primarily, formats define syntax, and not a specific implementation. Any experimental comparison therefore compares individual encoder implementations rather than the format itself. Secondly, the most meaningful comparison is a subjective one. However, subjective tests are difficult to conduct in a statistically meaningful manner. Finally, objective comparisons such as PSNR etc collapse the time varying encoder performance into a single data point. This glosses over aspects of rate control (which may be very different between encoder implementations), error patterns and temporal artifacts.

Having provided the above disclaimer, a rate-distortion result comparing VC-1 (Microsoft's WMV 9 implementation [50]) with H.264/AVC (Nokia implementation [51]) is presented. The results in Figure 4.1 and Figure 4.2 suggest that the PSNR performance of both codecs is data dependent and relatively similar. It must also be borne in mind that the complexity of H.264 is noticeably higher than that of VC-1.

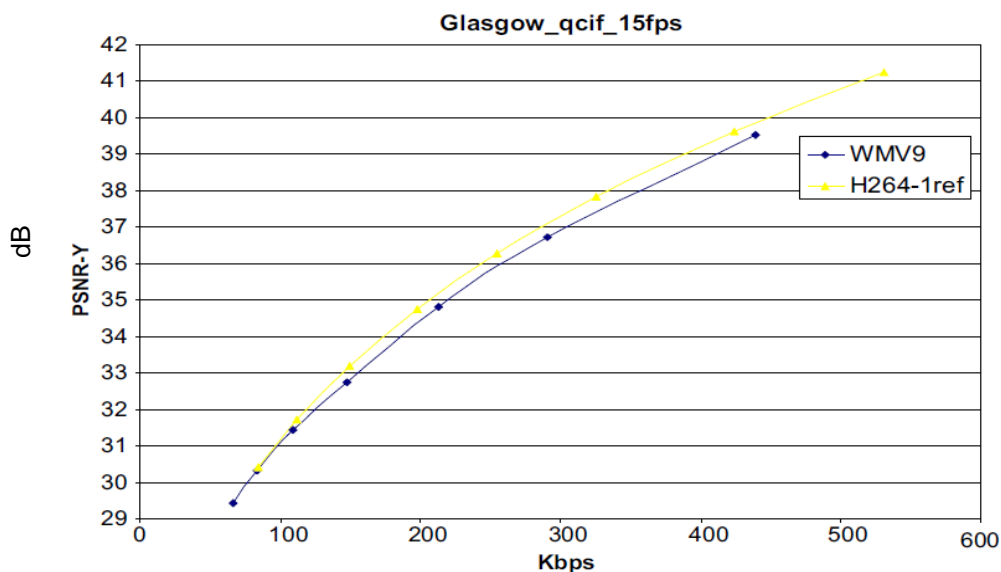


Figure 4.1 Glasgow - PSNR vs. Bitrate for WMV-9 Main (without B frames) and H.264/AVC Baseline, in favor of the former [42]

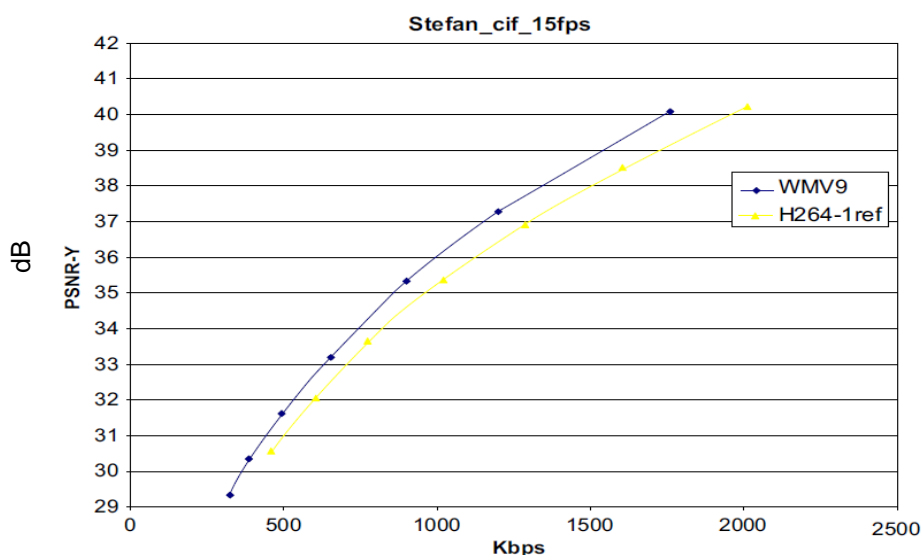


Figure 4.2 Stefan - PSNR vs. Bitrate for WMV-9 Main (without B frames) and H.264/AVC Baseline, in favor of the former [42]

When it comes to subjective quality, however, VC-1 (WMV 9) has equaled or outperformed optimized implementations of H.264/AVC. There have been a variety of studies that have independently evaluated the compression efficiency of WMV-9 and H.264/AVC. For example, Tandberg Television evaluated WMV-9 and the H.264/AVC (baseline and main profiles) implementations (version 6.0a). They concluded that the visual quality achieved by H.264/AVC main profile and WMV-9 was comparable, and that these two codecs provide the best quality among the competing codecs in their comparison. In another independent study from Computer Technique (C' T) Magazine [52], Germany's premiere AV/Computer magazine, the performance of WMV-9 and codecs based on H.264/AVC MPEG-4 (e.g., Dicas, DivX, XVID, etc.) were evaluated subjectively and objectively (using Sarnoff's JND metric, which is also integrated in Tektronix video analysis equipment [49]). In this study, WMV-9 was selected as producing the best subjective and objective qualities. In addition, WMV-9 also achieved the highest perceptual quality in each of the video clips tested at the latest independent test performed by the DVD Forum, which also evaluated optimized implementations of H.264/AVC main profile, MPEG-4 ASP, and MPEG-2.

The technical description of VC-1 in Section 0 provides sufficient reasons for the conclusions favoring VC-1, especially for high definition content. For instance, adaptive block transforms used in VC-1 retain the fine structure, textures and film grain of content more faithfully than H.264/AVC, which tends to smooth out fine details. This effect has been recently demonstrated by other researchers [48] [53]. As another example, the loop filter of VC-1 is weaker than that of H.264/AVC and this is a design choice that affects both complexity and subjective performance in favor of VC-1.

4.4 Results from the DVD Forum video codec tests [42]

The latest video codec tests of the DVD Forum provide further evidence of the strong capability of VC-1 for compressing high definition content. The Forum tested the performance of multiple video codecs (e.g., MPEG-2, MPEG-4 ASP, H.264, VC-1, etc.) in six film clips of time length of 90s and resolution of 1920x1080. These clips were selected

for their variety and complexity by major Hollywood studios. To be more concrete, the sequences 1, 2, 3, 4, 5, and 6, were of segments from the movies “Dick Tracy”, “Titan A.E”, “Harry Potter and the Sorcerer’s stone”, “Stuart Little 2”, “Seven”, and “Monsters, Inc”, respectively. Over 35 experts from the Studios and Consumer Electronic companies performed carefully crafted blind tests (e.g., using appropriate lighting conditions, randomizing the clips, displaying the results in a variety of high end monitors, etc.) and compared these codecs to the industry reference D-VHS (MPEG-2 at 24 Mbps) and the original D5 master (near-lossless compression at 235Mbps). Some key results from the DVD Forum test in regards of VC-1 are shown in Figure 4.3 and Table 4.2. The values in Figure 4.3 corresponds to the “Overall Impression” perceptual scores for the final round of tests at Panasonic Hollywood Labs. The scoring method used the typical 1–5 scale, where a score of 5 indicates that the given clip is perceived to be equal to the D5 reference. The Forum averaged and rounded the scores to the nearest one decimal point, and the D-VHS and D5 references were included at random in the testing set to eliminate bias. This is why not even the D5 clips achieved an average of 5. The D5 reference was always shown side-by-side with each of the compressed clips. The bit streams were cross-verified by independent companies. Both pre-processing and post-processing were not allowed in this final round of tests. In Figure 4.3, observe that, with only 7.7 Mbps, VC-1 achieved similar perceptual scores as the references. In fact, VC-1’s score was even higher than that of D-VHS in one sequence, and tied with the scores of the original D5 and D-VHS in another. Initially, there were nine video codecs participating in the DVD Forum codec tests, which included VC-1 and several professional (optimized) implementations of MPEG-2, MPEG-4 ASP, and H.264/AVC. Table 4.2 shows the final codec ranking per sequence for VC-1 and the best H.264/AVC and MPEG-2 implementations. The focus is only on these three codecs here, because they were the ones that achieved the highest scores. The codec ranking was obtained by averaging all the perceptual scores for that specific video sequence to the nearest one decimal point, and then ordering the codecs from higher to lower score. The numbers in parentheses indicate the difference in the average perceptual scores from VC-1. It was observed that VC-1 ranked first for each of the video sequences, and hence received

the highest scores and most consistent performance. H.264/AVC's scores were lower in sequences that included fine textures (e.g., Seq 4 is from a segment in "Stuart Little 2" that contains fine artificial textures), while MPEG-2 suffered with fast motion (e.g., Seq 6 is from a fast action scene in "Monsters, Inc").

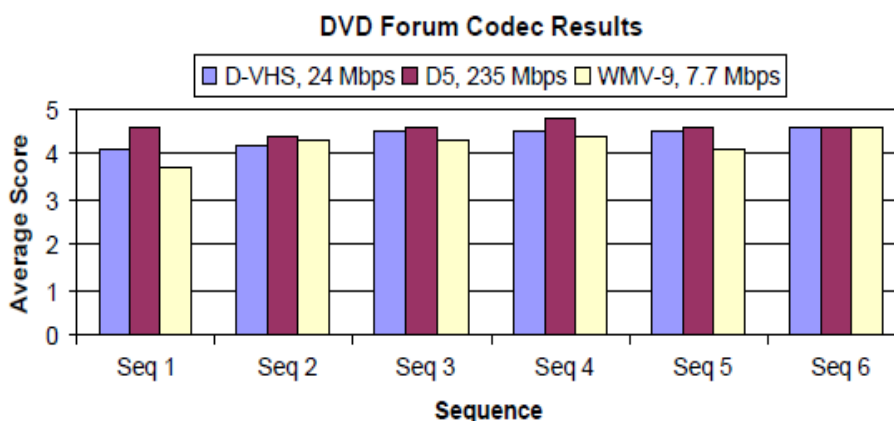


Figure 4.3 Average perceptual scores of VC-1 in comparison to D5 and D-VHS references, as performed by experts in the DVD Forum [42]

Table 4.2 Codec ranking for the three codecs that performed best in the DVD Forum - final set of tests. [42]

Codec Ranking	WMV-9	H.264/AVC	MPEG-2
Sequence 1 – Dick Tracey	1	1 (0)	4 (-0.4)
Sequence 2 – Titan	1	2 (-0.3)	3 (-0.4)
Sequence 3 – Harry Potter	1	2 (-0.2)	2 (-0.2)
Sequence 4 – Stuart Little 2	1	3 (-0.4)	2 (-0.1)
Sequence 5 – Seven	1	3 (-0.1)	1 (0)
Sequence 6 – Monsters Inc	1	2 (-0.1)	3 (-0.6)

4.5 Computational complexity

VC-1 is more complex to decode than MPEG-2 and MPEG-4 simple profile. However, a comparison of computational complexity is only meaningful between codecs that achieve similar levels of compression, which means that the only other codec of interest here is H.264/AVC. A concern with H.264/AVC is the high computational complexity required for encoding and decoding. For example, a preliminary study shows that the decoder complexity of H.264/AVC (main profile) is about three times higher than MPEG-4 simple profile [54]. On the other hand, the decoding complexity of VC-1 (main profile) is

relatively close to that of the MPEG-4 simple profile codec. To be more specific, decoding with VC-1 is about 1.4 times slower, which can be verified easily by using both codecs in the Windows Media Player (or other MPEG-4 simple profile decoders). Even though one cannot derive strong conclusions or draw parallels on such complexity analyses, this information suggests that H.264/AVC decoding complexity is likely to be twice that of the VC-1 codec, or at least that there is a significant computational benefit of VC-1 over H.264/AVC on the decoder side. Finally, the superficial computational complexity analysis is backed up with some experimental results. The decoder performance of VC-1 and H.264/AVC is compared on a non-PC ARM processor (clock cycles measured through the Armulator). Table 4.3 shows a table comparing the clock cycles required for VC-1 and H.264/AVC decode on 10s long clips encoded with the main profile of VC-1 and baseline profile of H.264/AVC. The numbers for H.264/AVC decoder are as optimized and reported by Nokia [51]. It is clear that there is a significant computational advantage with VC-1, i.e., VC-1 Main profile decoding requires 2–3 times less computation than H.264/AVC Baseline. Observe that H.264/AVC Main decoding is quite more complex than baseline (e.g., Main includes arithmetic coding).

Table 4.3 Comparison of WMV-9 main and H.264/AVC baseline decoder complexity in an ARM chip. [42]

Sequence	Millions of ARM cycles/sec	
	WMV-9	H.264
Foreman	27	70
News	17	45.9
Container	19	45.5
Silent	18	50.8
Glasgow	25	48.5
Average	21.2	52.14

4.6 Summary

The chapter gives an overview of the difference between the encoding tools and compares the complexity for both the codecs. Further performance comparison is given with an objective and subjective quality assessment. Having had a good understanding of both H.264 and VC-1, the next chapter details the similarities and dissimilarities between the two standards.

CHAPTER 5

TRANSCODING

5.1 Introduction

As described in Section 1.3 video transcoding is an operation of converting video from one format to another [12]. This format conversion includes a range of operations such as bit rate reduction, conversion of one compression format to another, altering video container format or changing the header descriptions and others. Apart from this basic format conversion, a transcoder can be used for other functions such as adjustment of coding parameters of compressed video, spatial and temporal resolution conversions, and insertion of new information such as digital watermarks or company logos and even enhanced error resilience [12].

Figure 5.1 shows different multimedia devices; these devices operate at different bit rates, picture resolutions and video coding algorithms. Due to the scope of communications in current times there is extensive exchange of multimedia data between various such systems of different configurations over networks of varied capacities. It is very important to be able to play the same video content for all the systems to ensure access to a widespread audience. Transcoding can enable multimedia devices of diverse capabilities and formats to exchange video content on heterogeneous network platforms [56].

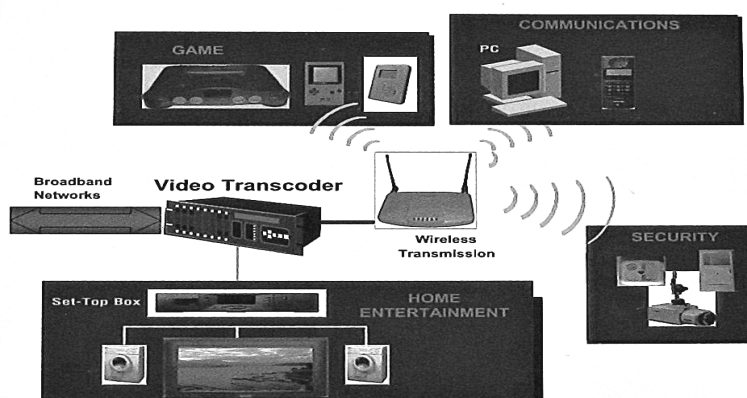


Figure 5.1 Communication between various multimedia devices [55]

Figure 5.2 is a high level look on transcoder functionalities. High bit rate video from high quality sources such as DVD and HDTV broadcasting can be made available on hand held devices such as smart phones, PDAs, etc. In video conferencing transcoding can enable the adjustment of bit rate to support the bandwidth requirements and also any format conversions if required. This attempt to ensure availability to all media content over diverse networks and systems with varied capabilities forms an important aspect of universal multimedia access [11] [12].

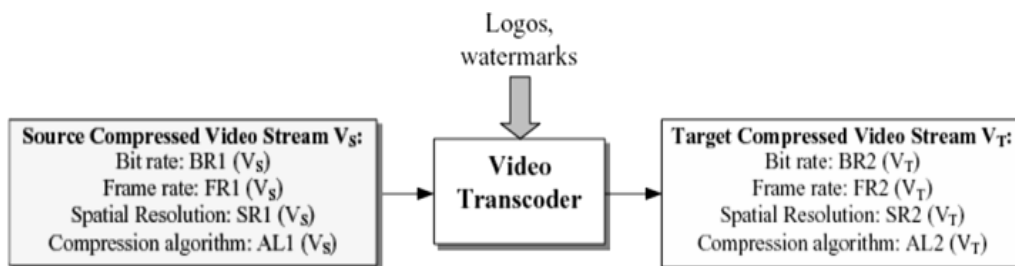


Figure 5.2 Video transcoding operations [12]

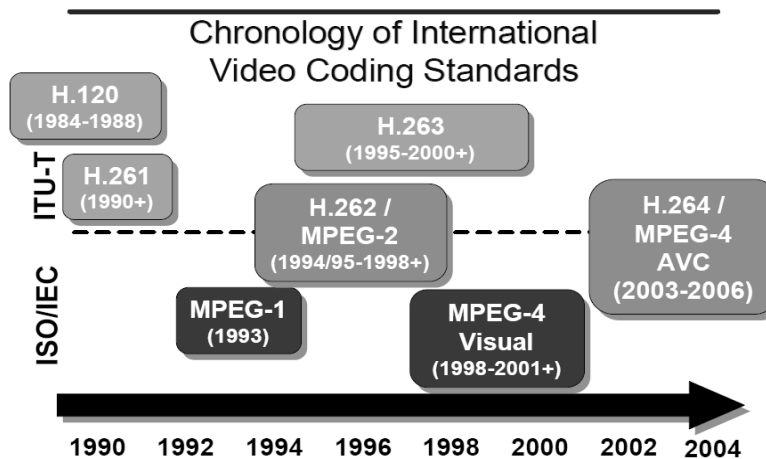
Several video coding standards exist currently [9] [10] [13] [15]. Each of them suits a diverse range of applications, but is optimized for certain kind of applications. This results into use of varied standards over diverse applications. Table 5.1 lists the different codecs used for different multimedia applications. Some of the standards are proprietary and the IP owners prefer using their standards. Some of the common standards can be listed as - H.261, H.262 H.263, H.263+ designed by ITU (International Telecommunication Union); they are aimed at low-bit-rate video applications such as videophone and videoconferencing. MPEG standards are defined by ISO (International Organization for Standardization). MPEG-2 is aimed for high bit rate high quality applications such as digital TV broadcasting and DVD, and MPEG-4 is aimed at multimedia applications including streaming video applications on mobile devices [13]. The new and highly popular H.264 standard is used for high resolution video content and a range of broadcast applications. This standard addresses a very wide variety of applications [9]. Apart from that, Microsoft developed the Windows Media Video standard WMV9 which is adopted by SMPTE as VC-1 standard and is popular [10]. VC-1 along with H.264 and MPEG-2 is also used for high definition content in Blu-ray Disc standard [24]. Thus a wide range of standards exist.

Figure 5.3 provides an overview of how these standards have grown over the years and the increase in complexity along with that. Transcoding one video format to another is increasingly significant in the current scenario of diversifying and increasing multimedia applications. The current research on H.264 and VC-1 standards is aimed at the interoperability of the two popular standards supported in Blue Ray Discs [24].

Table 5.1 Different multimedia applications and corresponding video standards [14]

Application	Bitrate	Video standard
Digital TV broadcasting	2 to 6 Mbps	MPEG-2
DVD Video	6 to 8 Mbps	MPEG-2
Internet video streaming	20 to 200 kbps	Flash – Sorrenson spark, VP6 & H.264; Silverlight uses VC-1; MPEG-4 Part 2 (Visual)
Video conferencing and video-telephony	20 to 320 kbps	H.261, H.263.
Video over 3G wireless	20 to 100 kbps	H.263, MPEG-4 Part 2 (Visual)
High definition – Blu-ray	36 to 54 Mbps	H.264, VC-1 and MPEG-2
Handheld and mobile devices	64 kbps - 2 Mbps	H.264 baseline (ATSC/A153)

Apart from this, transcoding is useful in a range of other applications. In statistical multiplexing, the bit rate increases as various multi-bit-rate video streams are multiplexed. A transcoder can be used to adapt the bit-rates of the video streams when the aggregated bit-rate exceeds the channel bandwidth [56]. A transcoder can also be used to insert new information including company logos, watermarks, as well as error-resilience features into a compressed video stream. Transcoding techniques are also useful for supporting VCR trick modes, i.e., fast forward, reverse play, etc., for on-demand video applications [56]. For adaptive video content delivery, object based transcoding techniques can be used [56].



(a)

- *Intraframe coding*: only spatial correlation exploited
 → DCT [Ahmed, Natarajan, Rao 1974], JPEG [1992] Complexity increases
- *Conditional replenishment, DPCM, scalar quantization*
 → H.120 [1984]
- *Frame difference coding*
 → H.120 Version 2 [1988]
- *Motion compensation: integer-pel accurate displacements*
 → H.261 [1991]
- *Half-pel accurate motion compensation*
 → MPEG-1 [1993], MPEG-2/H.262 [1994]
- *Variable block-size motion compensation*
 → H.263 [1996], MPEG-4 [1999]
- *Variable block size Intra directional prediction, motion compensation, in loop filtering (2003)*
 → H.264/MPEG-4 Part 10

(b)

Figure 5.3 Video coding standards (a) Timeline [58] (b) Increase in complexity [57]

Transcoding can be broadly categorized into homogenous and heterogeneous transcoding techniques based on applications. Figure 5.4 shows the classification of different transcoding techniques.

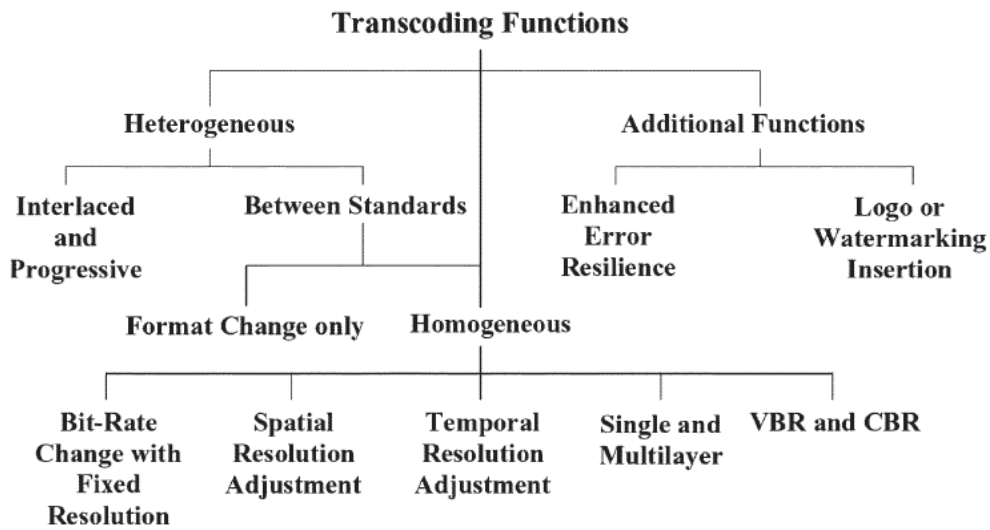


Figure 5.4 Video transcoding operations and classification [12]

Homogenous transcoding performs conversion between video bit streams of the same standard. Bit rate conversion is one such application. A simple technique to achieve bit rate reduction can be to increase the quantization step at the encoder part of the transcoder [12] [59] [60] (Section 5.2.2.3). As the quantization resolution decreases, the number of non-zero coefficients decreases, thereby resulting in bit rate reduction. The complexity of such an application is less, although the reconstructed image quality can be affected; it is considered as a good trade-off. Another method of fixed resolution bit rate reduction can be selective transmission (Section 5.2.2.2). Since most of the energy is concentrated in the lower frequency bands of an image, discarding (truncating) some of the higher frequency coefficients [12] [55] can be used. This can preserve picture quality, but can introduce blocking artifacts in the reconstructed target video. Various other methods for bit rate transcoding at architectural levels are discussed in Sections 5.2 and further.

Other application of homogenous transcoding is spatial or temporal transcoding. As shown in Figure 5.5 [12], spatial transcoding can be implemented in various ways which can both achieve spatial as well as bit rate adjustments. There can be multiple aims of performing spatial transcoding such as to sub sample or even to extract sections of the image to user's interest as shown in Figure 5.6 [12]. This requires the use of meta information. In sub sampling, filtering and pixel averaging to reduce spatial resolution [12]

problems arise when passing motion vectors directly from the decoder to the encoder. Thus, motion vectors need to be refined [12] (Section 5.2.5.4). In [61] Shanabelah proposed a filter that can be used in both horizontal and vertical directions for luminance and chrominance; the image is then down-sampled by dropping every alternate pixel in the both horizontal and vertical directions (Section 5.2.5.1). In pixel-averaging (Section 5.2.5.2) [61], MxM pixels are represented by averaging their values to a single pixel. It is a very simple method, but the reconstructed pictures may become blurred. In [62] spatial resolution reduction is achieved by performing decimation in DCT domain by discarding the higher order DCT coefficients (Section 5.2.5.3).






	WorkStation/LAN	PC/Dialup	TV Browser	Gray PDA	B/W PDA
Video					
Color	24 bit color	24 bit color	256 colors	4 bit gray	B/W
Size	352x288	176x144	176x144	160x120	128x96
Bits	64KB	33KB	8KB	4KB	0.6KB

Figure 5.5 Various ways of spatial transcoding (Bits = bits/frame required for transmission) [12]

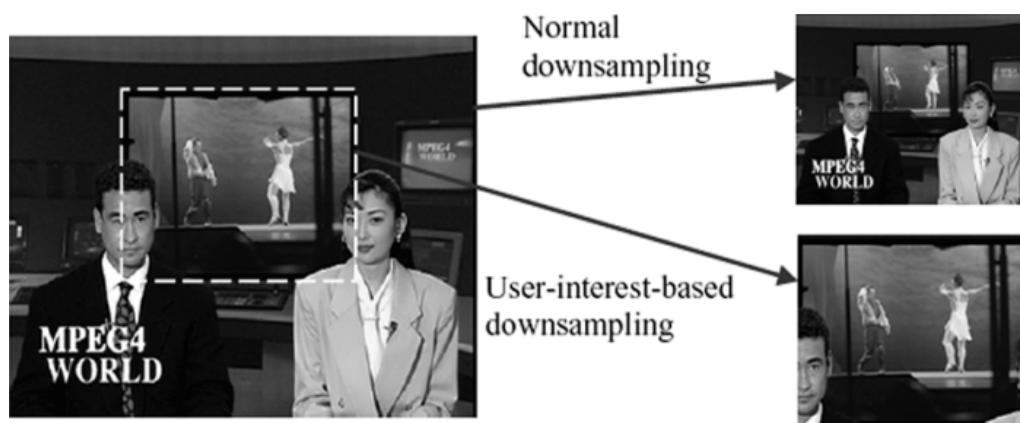


Figure 5.6 Transcoding with normal down sampling and user-interest-based down sampling [12]

Frame-rate conversion is needed when the end-system supports only a lower frame-rate. Reduction in frame rate may save bits that can be used in the remaining frames

to maintain acceptable overall picture quality for each frame. However this requires motion vector recalculation. With the dropped frames, incoming MVs may not be valid since they point to frames that do not exist. In [63] a method to estimate the new MVs by using bilinear interpolation is described. Given that the MVs between every adjacent dropped frames are known, bilinear interpolation (Section 5.2.6.1) is used to calculate the new MVs between the current and previous non-skipped frame. Another method proposed in [64] known as the Forward Dominant Vector Selection (FDVS) (Section 5.2.6.2) selects dominant MV from the four neighboring macroblocks. A dominant MV is defined as the MV carried by a macroblock that has the largest overlapping segment with the block pointed by the incoming MV. The Telescopic Vector Composition (TVC) technique (Section 5.2.6.3) described in [61] sums up all the MVs of the corresponding macroblocks of the dropped frames and adds the resulting combined MV to its corresponding MV in the current frame. This technique also carries out new macroblock decision and MV refinement. Another algorithm known as Activity-Dominant Vector Selection (ADVS) (Section 5.2.6.4) is described in [65]. It utilizes the activity of the macroblock to decide the choice of the MV. The activity information of a macroblock is represented by counting the number of nonzero quantized DCT coefficients of covered 8x8 residual blocks, other statistics, such as the sum of the absolute values of DCT coefficients, etc.

A heterogeneous video transcoder provides conversions between two different video coding standards. This involves basic syntax conversion between the two standards as the first step. Further it can also provide all the other functionalities of a homogenous transcoder.

5.2 Video transcoding architectures

5.2.1 Cascaded decoder and encoder model

The most straightforward transcoding architecture is to cascade the decoder and encoder directly as shown in Figure 5.7 [12]. In this architecture, the incoming source video stream (V_S) is fully decoded, and then the decoded video is re-encoded into the target video stream (V_T) with desirable bit-rate or format; the process of transcoding does not introduce

any degradation into the visual quality. The more detailed manifestation of the cascaded transcoder is shown in Figure 5.7 (b) [12].

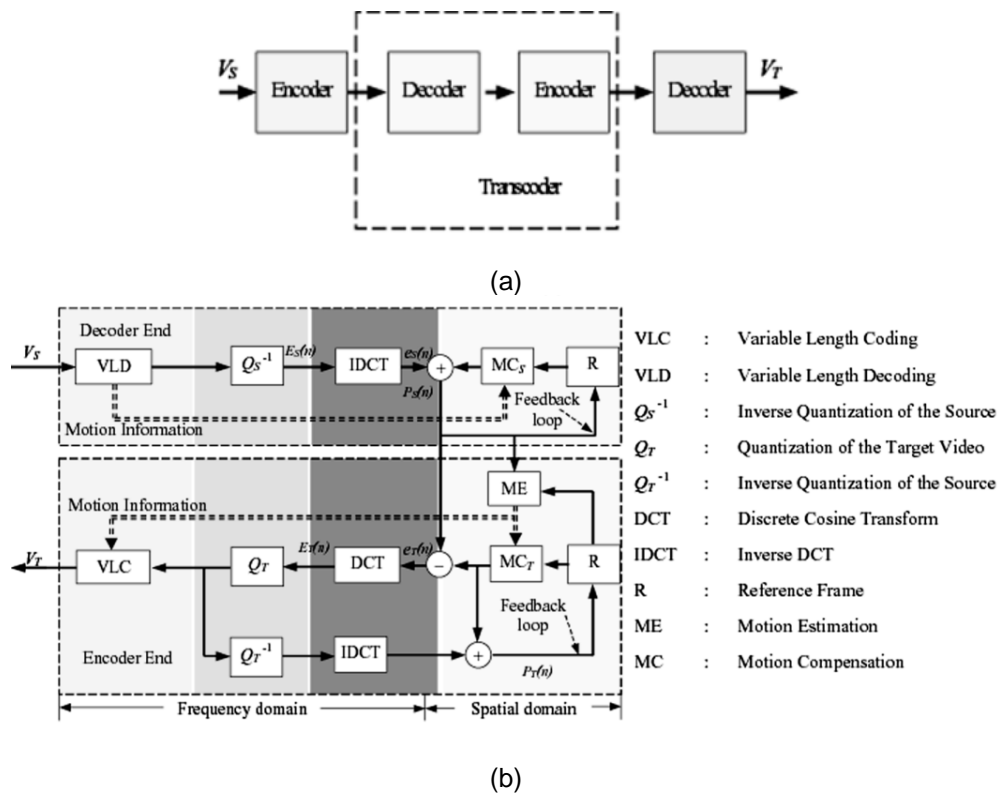


Figure 5.7 Cascaded decoder and encoder model [12] (a) block level diagram (b) detailed diagram

This type of implementation involves complete decoding and re-encoding of the incoming compressed video stream. It has to perform full decoding followed by the resizing / re-ordering of the decoded sequence before re-encoding it. Due to the complete re-encoding operation complex frame reordering and full-scale motion re-estimation are required. Motion estimation has the highest complexity in the encoder. So such an implementation involves the maximum complexity and also high processing time and power consumption leading to significant delay [55]. Also the pictures / frames exhibit increased error due to re-encoding being performed on decoded pictures which have lower quality than original frames. The error is due to propagation. Lossy encoding process inserts errors; when such a bitstream is decoded, the decoded pictures have errors which propagate on further encoding which inserts more errors. This error is not the same as the drift error in

open loop transcoders described in Section 5.2.2.1. Due to all these reasons such a transcoding model needs a lot of optimization. Different methods are described further to optimize transcoder performance and reduce complexity.

5.2.2 Open loop transcoding architecture

The simplest method to reduce the complexity of the cascaded decoder encoder transcoding model is to use open-loop transcoding architecture. Such an architecture aims to use minimum transcoding complexity by only modifying the encoded DCT coefficients. Figure 5.8 shows an open loop transcoder. Since only the DCT coefficients are modified in order to adjust the bit rate, other video parameters remain unaffected; the DCT encoded error coefficients are decoded while the rest of the parameters are encoded using variable length coding (VLC). After the DCT coefficients are decoded, operations may be performed on them to reduce the bit rate. Bit rate reduction can be achieved by either throwing away the high frequency components or coarser re-quantization of the decoded coefficients. These schemes are discussed in Sections 5.2.2.2 and 5.2.2.3 respectively. This architecture is so called because it does not have any feedback loop which can compensate the drift errors (Section 5.2.2.1 describes drift errors and reduction of the drift errors using a feedback loop). The picture drift occurs from the mismatch between the locally reconstructed pictures at the encoder and the transcoded pictures in the system.

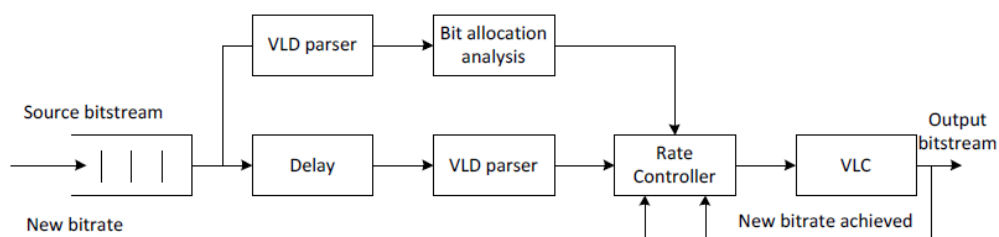


Figure 5.8 Open loop transcoder architecture [55]

5.2.2.1 Picture drift error

A cumulative effect occurs due to the mismatch between the reconstructed images of the originally encoded video frames and the transcoded video frames. Such errors propagate along the video sequence. The errors resulting from mismatch between the video

frames at the original encoder and the eventual decoder are known as drift errors and the effect is known as the picture drift effect in a transcoder [12]. The following sets of equations [12] develop drift errors mathematically. All the symbols in the equations below have reference in the Figure 5.7 (b).

$$P_S(n) = e_S(n) + MC_S(P_S(n-1)) \quad (5.1)$$

$$e_T(n) = P_S(n) - MC_T(P_T(n-1)) \quad (5.2)$$

$$\begin{aligned} E_T(n) &= DCT(e_T(n)) \\ &= DCT(P_S(n) - MC_T(P_T(n-1))) && \text{[from equation 5.2]} \\ &= DCT(e_S(n) + MC_S(P_S(n-1)) - MC_T(P_T(n-1))) && \text{[from equation 5.1]} \\ &= DCT(e_S(n)) + DC_T(MC_S(P_S(n-1)) - MC_T(P_T(n-1))) \\ &= E_S(n) + DCT(\Delta MC) \end{aligned} \quad (5.3)$$

where

$$\Delta MC = MC_S(P_S(n-1)) - MC_T(P_T(n-1))$$

Equation 5.3 shows that if ΔMC is non-zero i.e. the video frames at the original encoder and the eventual decoder are different; the transcoder output has errors compared to the original input. These errors build up as the video frames progress. As observed from equation 5.3, the errors come up during motion compensation (MC). Since there is no motion compensation in intra frames, drift errors do not occur in intra-frames. However as each predicted frame (P-frame) makes use of another P-frame which already has drift errors, the errors build up. So drift error keeps increasing with each frame until a new I-frame is reached. B-frames are not used for prediction. So they do not contribute to further increase the drift errors [12].

The feedback loop in Figure 5.7 (b) makes sure the building up of drift errors can be avoided. The feedback loop can ensure that for further prediction the MC block at the encoder uses the same frame as the MC block of the decoder thereby preventing the building up of drift errors due to erroneous prediction frames.

5.2.2.2 Truncation of high-frequency coefficients

Figure 5.9 shows an open-loop transcoder which reduces the bitrate by discarding the high frequency coefficients. The variable length decoder (VLD) decodes the DCT coefficients and based on the target bit-rate defines a scaled bit usage profile to meet the target rate. The rate controller simply has to discard the coefficients that exceed the scaled profile. For this most often the VLD only needs to decode the codeword lengths. This method does not need to perform inverse quantization and re-quantization as the method described in Section 5.2.2.3. It is the simplest implementation of a bit-rate transcoder.

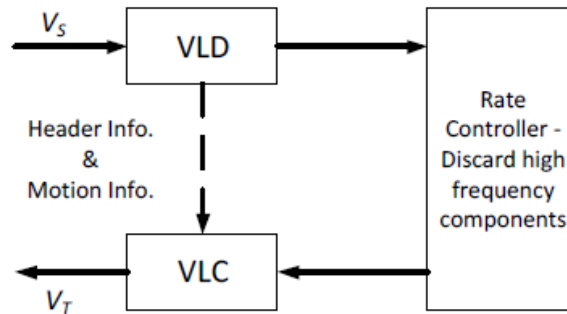


Figure 5.9 Bit rate reduction by truncation of high frequency coefficients [12]

5.2.2.3 Re-quantization to reduce bit rate

Figure 5.10 shows an open-loop transcoder which employs the method of coarser re-quantization of DCT coefficients in order to achieve reduction in bit rate. Compared to the architecture in Section 5.2.2.2 it has a complete VLD block, an inverse quantizer and a quantizer. All these contribute towards a slight increase in the complexity. The DCT coefficients are decoded first, inverse quantized, re-quantized with a coarser quantization value and variable length coded (VLC) again. This architecture also exhibits drift errors as there is no feedback loop. The equations below give an analysis of the drift error in open-loop architecture. All the symbols used in the equations are in reference with Figure 5.10

$$\begin{aligned}
 E'_T(n) &= E_S(n) \\
 E_T(n) &= E'_T(n) + \text{DCT}(\Delta\text{MC}) && \text{[from equation 5.3]} \\
 d_e(n) &= \text{DCT}(\Delta\text{MC}) = E_T(n) - E'_T(n) && (5.4)
 \end{aligned}$$

It can be seen from equation 5.4 that here $d_e(n)$ is the drift error which is introduced by the process of re-quantization. Similarly in the scheme in Section 5.2.2.2 such an error is introduced by the discarding on non-zero high frequency coefficients.

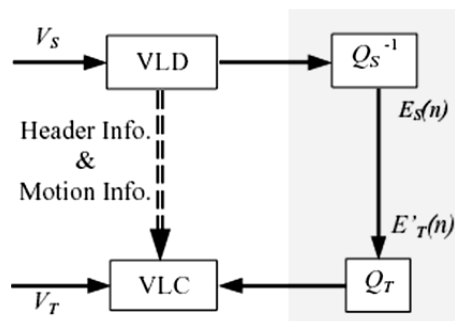


Figure 5.10 Transcoding with re-quantization scheme [12]

5.2.3 Spatial domain transcoding architecture

Figure 5.7 (b) shows spatial-domain transcoding architecture (SDTA). It is the most basic architectures to perform bit-rate reduction; but as described in Section 5.2.1, it has a very high complexity. Figure 5.11 shows a slightly modified SDTA which makes reuse of the motion information from the video decoder. This results in significant reduction in complexity since motion estimation alone contributes 60-70% of the encoder time complexity [12]. As it can be observed, there are optional functional blocks between the decoder and the encoder in Figure 5.11. Motion vector composition and refinement (MVCR) and spatial/temporal resolution reduction (STR) can be performed at these blocks. Different methods to achieve these are explained in Section 5.2.5.4 and Section 5.2.5 respectively. If the only aim of the SDTA is bit-rate reduction the transcoder can be further simplified as shown in Figure 5.12 [12].

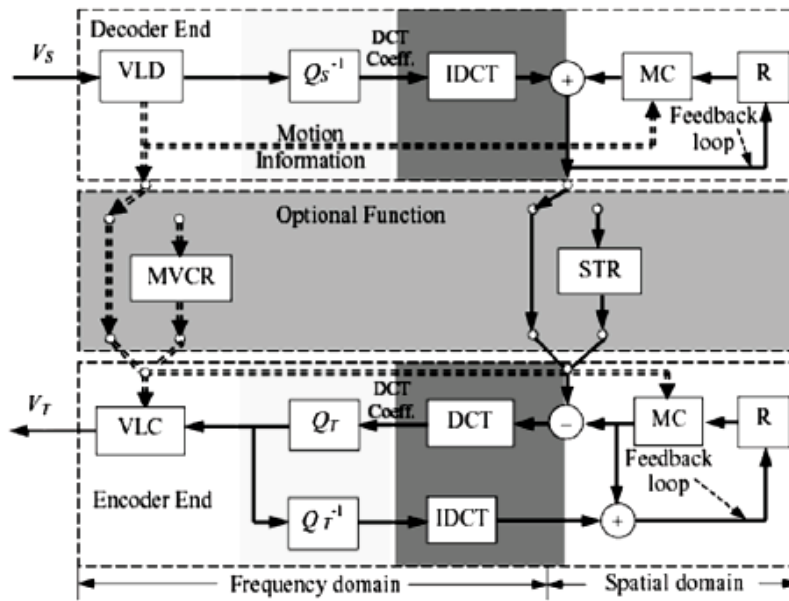


Figure 5.11 Spatial domain transcoding architecture (SDTA) with MV reuse and STR [12].

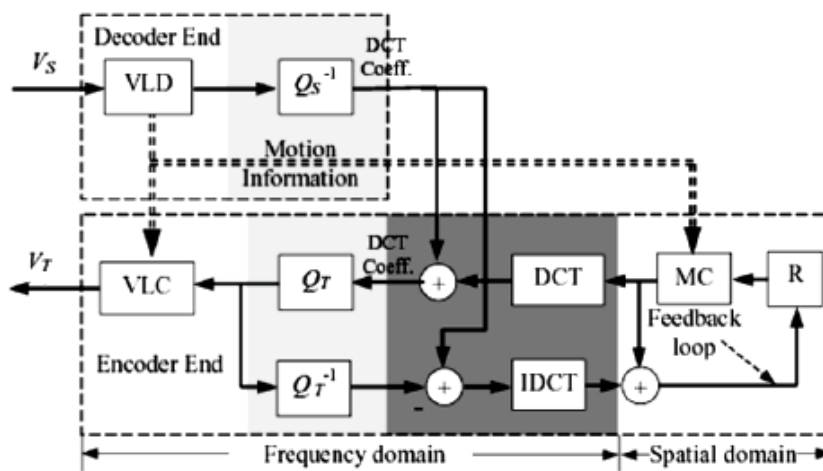


Figure 5.12 Simplified SDTA without STR [12]

5.2.4 Frequency domain transcoding architecture

Some operations in the decoder part of Figure 5.11 could be reduced for a simplified SDTA in Figure 5.12. Further simplification can be achieved if the DCT/IDCT operations can be removed in the encoder by performing motion compensation in frequency domain. Figure 5.13 describes simplified frequency domain transcoding architecture (FDTA) which can achieve this. Here motion compensation (MC) is achieved in the frequency domain. FDTA gives significant reduction in complexity over the cascaded

scheme but it can have drift since sub-pixel motion compensation cannot be done at the frequency level [12].

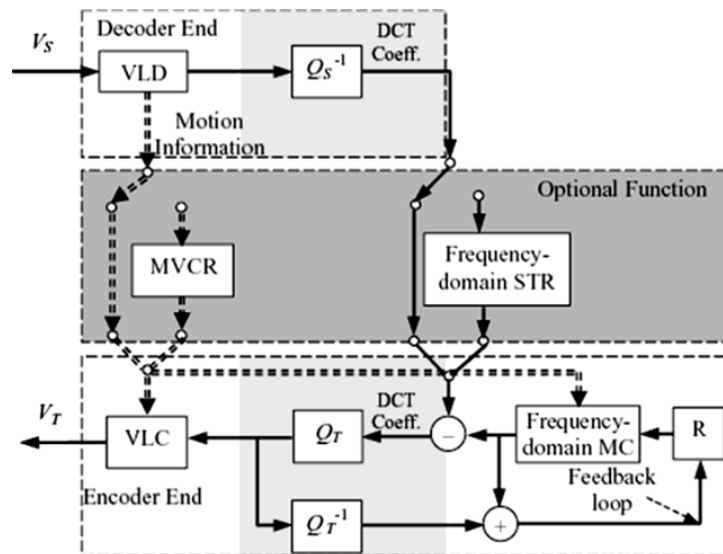


Figure 5.13 Frequency domain transcoding architecture (FDTA) [12]

5.2.5 Spatial resolution reduction

Spatial resolution reduction (SRR) can be achieved using either SDTA or FDTA described in Sections 5.2.3 and 5.2.4 respectively. Spatial resolution reduction comes along with bit rate reduction. With the spatial resolution reduction, motion vectors need to be recalculated; Section 5.2.5.4 described techniques for motion vector composition and refinement. Also with spatial resolution reduction, the MB mode decisions need to be made again; Section 5.2.5.5 discusses MB mode decisions in such scenarios. Sections 5.2.5.1 through 5.2.5.3 describe different techniques for SRR.

5.2.5.1 Filtering and sub-sampling

Sub-sampling is a method to reduce spatial resolution in the spatial domain. Sub sampling needs a decimation filter before dropping the alternate pixels. The decimation filter is applied in both horizontal and vertical directions for luminance and chrominance after which down sampling is performed by dropping alternate pixels as shown in Figure 5.14.

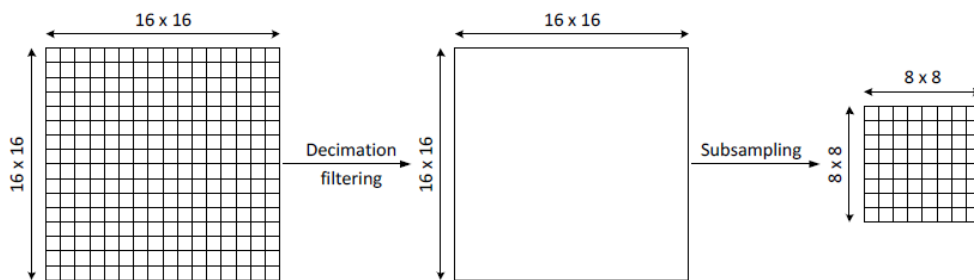


Figure 5.14 Decimation on 16 x 16 MB [61]

5.2.5.2 Pixel averaging

The simplest method to reduce spatial resolution is pixel averaging as show in Figure 5.15. Here every $M \times M$ pixels are represented by 1 pixel in order to achieve $M:1$ spatial resolution reduction. Most often the calculated value for this replacement pixel is the average of the $M \times M$ original pixels. However this can clearly introduce blurring in the picture [61].

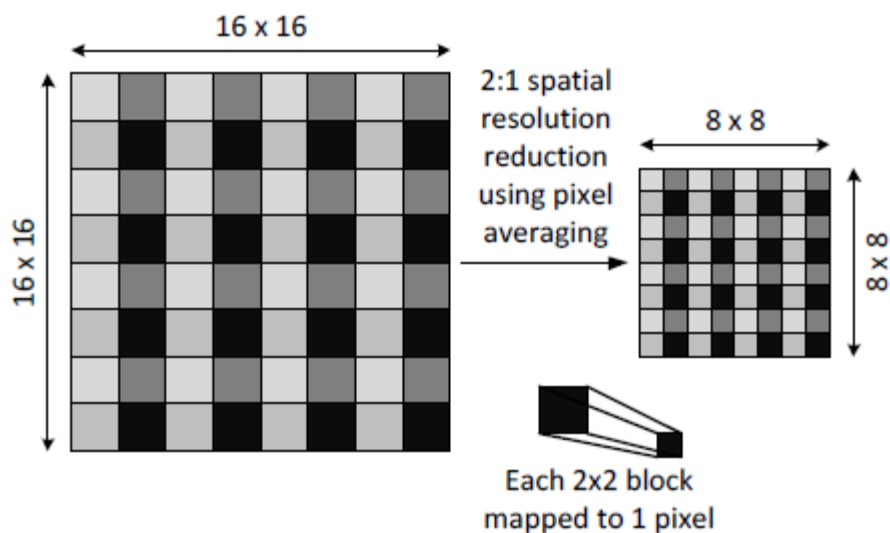


Figure 5.15 Pixel averaging [61]

5.2.5.3 Discarding high frequency DCT coefficients

A more efficient method to achieve spatial resolution reduction is discarding the higher order DCT coefficients. This method performs the scaling operation in the DCT domain so it can be used in the FDTA where the DCT domain MC is performed. It is much

simpler since it simply involves discarding all the higher order frequency coefficients as shown in Figure 5.16. According to a comparative study carried out by the authors in [61] DCT domain decimation technique performs the best of the three SRR techniques described.

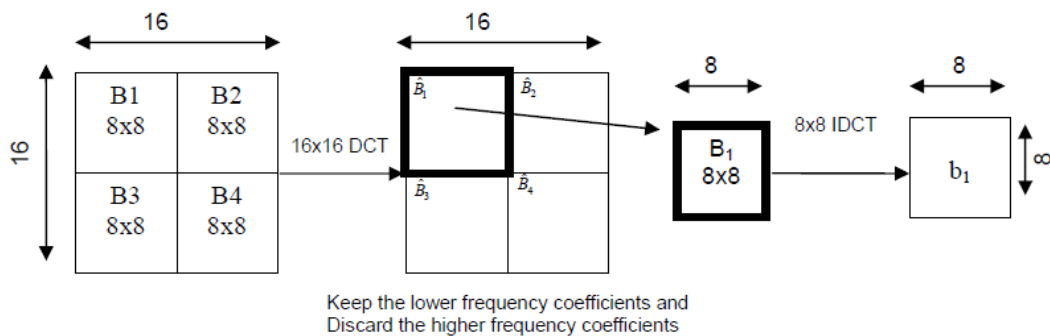


Figure 5.16 DCT domain decimation for SRR [61]

5.2.5.4 Motion vector composition and refinement

For SRR the motion vectors from the source cannot be passed directly to the encoder. When the resolution is reduced by 2:1 as shown in Figure 1.1, multiple MVs need to be merged into a single MV. This process is motion vector composition.

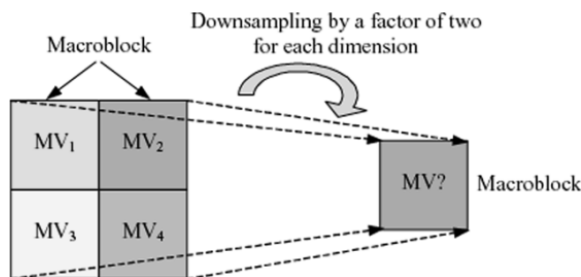


Figure 5.17 Four motion vectors being down sampled to one [12]

Numerous simple and involved methods have been proposed. Some of them are as follows.

- Random selection: A fast method of MV composition is to select any random MV from the incoming MVs to replace all of them. However this is a very inefficient method [12].
- Mean: An average value of all the incoming motion vectors can be used; this method however can be used only if all the motion vectors are in the same

direction. Most often there is no significant difference between the MV values of neighboring MBs. But this method is ineffective if one of the original MVs is much larger compared to the rest of the MVs [12].

- Median is let to represent the four adjacent motion vectors. The distance between each vector and the rest is calculated as the sum of their euclidean distances as follows:

$$d_i = \sum_{\substack{j=1 \\ j \neq i}}^4 \|v_i - v_j\|.$$

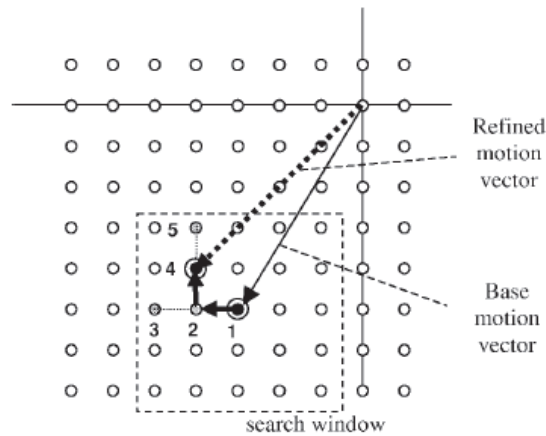
The median vector is defined as one of these vectors that has the least distance from all, i.e.

$$\text{med}(V) = v_k \in V \quad \text{such that} \quad \min d_i = d_k.$$

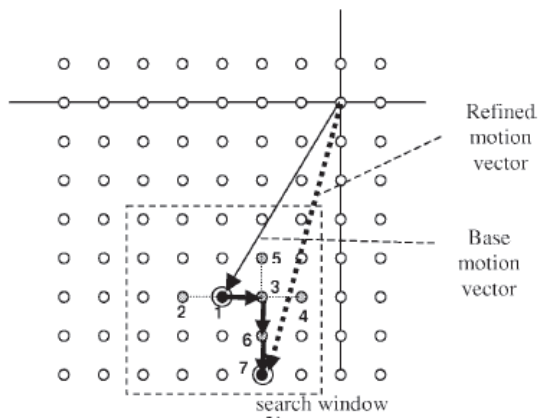
This method extracts the motion vector situated in the middle of the rest of the motion vectors. The magnitude of the selected motion vector is then scaled to reflect the reduction in the spatial resolution.

- DCMax: A technique used for MV composition makes use of the incoming MV with the maximum DC coefficients of the residual block in the source video. This method, according to [12], is more complicated but gives better results compared to taking mean or random selection.

The motion vector composition schemes are sub-optimal and introduce degradation in quality at the output. So MV refinement techniques are proposed in [64]. The recalculated MVs will not differ very much from actual motion vectors; so refinement of this MV in a small search window gives better results. Figure 5.18 describes a MV refinement scheme proposed in [64].



(a)



(b)

Figure 5.18 Motion vector refinement using search window (a) best case – small search (b) worst case – long search [64]

5.2.5.5 MB coding mode decision

With the reduced spatial resolution the MB coding modes need to be reevaluated. For 2:1 downscaling four incoming MBs are coded into a single MB. There can be two scenarios – all four of them have the same MB coding mode or all four of them have a different coding mode as shown in figure 5.19. In [66] the following scheme is proposed when all four incoming MBs have the same coding mode.

- If the incoming MBs are coded are INTRA MBs re-encode the reduced MB as INTRA.

- If the incoming MBs are SKIPPED (a macroblock which has no information sent regarding it; all the required parameters are derived from previously encoded macroblocks) again code the reduced MB as SKIPPED.
- If the incoming MBs are coded as INTER, check to see if all the coefficients in the reduced MB are zero; than it should be coded as SKIPPED. If not it should be encoded as either INTRA or INTER.

For the second scenario as shown in figure 5.19, the authors in [67] have described the following scheme.

- At the transmitter: If one of the four is an INTRA coded MB, than pass the new MB as intra. If one of them is INTER and none of them INTRA, pass the new MB as INTER MB. If all MBs are SKIPPED pass the new MB as SKIPPED.
- At the encoder: Re-evaluate the MBs as applicable.

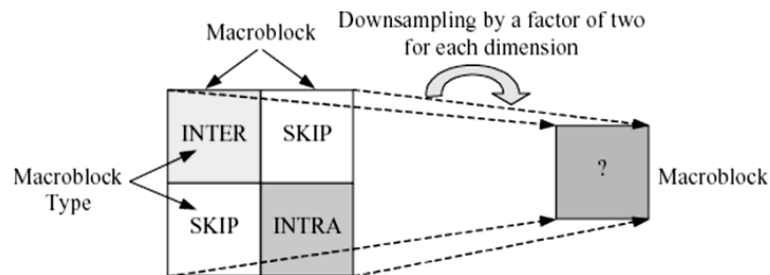


Figure 5.19 Four macroblock types down sampled to one [12]

5.2.6 Temporal resolution reduction

With frames dropped in order to achieve temporal resolution reduction (TRR) the incoming MVs are not valid as they point to the frames that do not exist in the transcoded bitstream. New MVs need to be derived. Sections 5.2.5.1 through 5.2.5.4 describe different techniques to derive new MVs from the MVs of the dropped frames.

5.2.6.1 Bilinear Interpolation

In [68] a bilinear interpolation method to estimate the new MVs is proposed. The new MV is calculated using interpolation of MVs between every adjacent frame between the current frame and previous non-skipped frame. The new location position based on this interpolated MV servers as the search center to calculate the actual value of the new MV;

thereby reducing the complexity in the new MV search. The search area is calculated from the number of skipped frames and accumulated magnitudes of their MVs.

5.2.6.2 Forward Dominant Vector Selection (FDVS)

The method proposed by the authors in [64] selects the dominant MV from the four neighboring macroblocks as shown in Figure 5.20. A dominant MV is defined as the MV carried by the macroblock that has the largest overlapping segment with the block pointed by the incoming MV. The best-matched area pointed by the MV of the current macroblock occurring after a dropped frame overlaps with at most four macroblocks in the previous dropped frame. The MV of the macroblock with the largest overlapping portion is selected and added to the current MV. This process is repeated each time a frame is dropped until a new set of MVs is composed for the last non skipped frame.

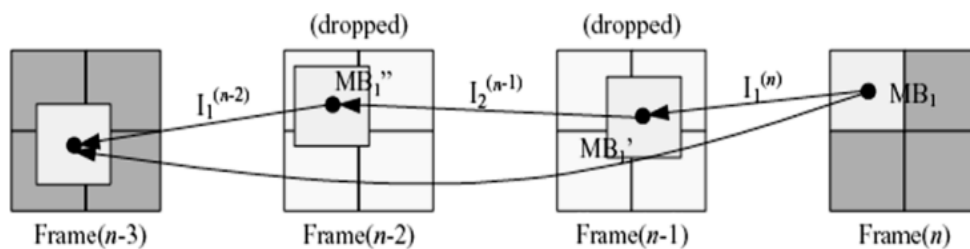


Figure 5.20 FDVS motion vector composition scheme for TRR [12]

5.2.6.3 Telescopic Vector Composition (TVC)

The TVC technique is described in [61]. It accumulates all the MVs of the corresponding macroblocks of the dropped frames and adds each resultant composed MV to the corresponding MV in the current frame. This technique also carries out mode decision and MV refinement.

5.2.6.4 Activity-Dominant Vector Selection (ADVS)

The authors in [65] describe this technique which makes use of the activity of the macroblock to choose the new MV. The activity information of a macroblock is represented by counting the number of nonzero quantized DCT coefficients of covered 8 x 8 residual blocks; other statistics, such as the sum of the absolute values of DCT coefficients, etc.

These quantities are proportional to the spatial-activity measurement. The higher the activity of the macroblock, the more significant will be the motion of the macroblock. Since the quantized DCT coefficients of prediction errors are available in the incoming bitstream of transcoder, the computation for counting the nonzero coefficients is very little.

5.2.7 Heterogeneous Transcoding

A heterogeneous video transcoder provides conversions between various standards, for instance, MPEG-2 to H.264 transcoder [29], VC-1 to H.264 [17], etc. Further, a heterogeneous video transcoder may also provide the functionalities of homogeneous transcoding [61]. Several techniques aimed for homogeneous transcoding can also be exploited in heterogeneous transcoding.

5.2.7.1 Main Issues in Heterogeneous Transcoding

A heterogeneous transcoder needs a syntax conversion module, and may change the picture type, picture resolution, directionality of MVs, and picture rate. A heterogeneous transcoder must adjust the features of the incoming video to enable the features of the outgoing video. Due to spatial-temporal sub sampling, and different encoding format of the output sequence, the encoder and decoder motion compensation loops in a heterogeneous transcoder are more complex [12].

5.2.7.2 Generic Heterogeneous Transcoder

A generic heterogeneous transcoder is showed in Figure 5.21. In this architecture, syntax conversion (SC) is needed to convert the syntax of source video to that of the target video. A higher resolution decoder decodes the incoming bitstream. The extracted MVs are then post-processed according to the desired output encoding structure, and if required, they are properly scaled down to suit the lower spatial-temporal resolution encoder. In case post-processing is not sufficient, the extracted MVs are refined to improve the encoding efficiency. The decoded pictures are accordingly down-sampled spatially or temporally, and the down-sampled images are encoded with the new MVs. Since the incoming MVs are re-employed, and other encoding decisions, such as macroblock types can be extracted from

the incoming bit stream, the architecture of this transcoder can be further simplified. In this architecture, the MVs of the incoming bitstream are employed in the outgoing one. So the extracted MVs have to be converted to be compatible with the encoding nature of the output bitstream. Note that the nature of extraction of the MVs and their usage depend on the picture type. The algorithm proposed in [61] assumes the motion between the pictures is uniform, such that the forward and the reverse MVs are images of each other, or an inter-frame MV is a scaled version of a larger picture distance and so on. In case no MV is found, one might either use a (0, 0) MV or in the worst-case intra-frame code the underlying macroblock. In [61], encoding format of MPEG-1 and MPEG-2 is first transcoded into H.261/H.263, the algorithm adopted the incoming motion parameters of a sub GOP of up to three frames to produce several candidate MVs for the outgoing picture. Then all the estimated MVs are compared, and the one that gives the least coding error in terms of sum of absolute differences (SAD) is chosen. The best MV was then refined by half-pixel (or one pixel) motion estimation to produce near-optimum results. Second, transcoding from the encoding format of H.261/H.263 into H.263 predictive/ bi predictive (P/B) frames. The new MVs of both P and B frames were calculated in a similar manner to that used in the first case. After the new MVs are obtained, and spatial or temporal reduction is performed, the encoder in the heterogeneous transcoder can code every picture according to the picture type of the new format.

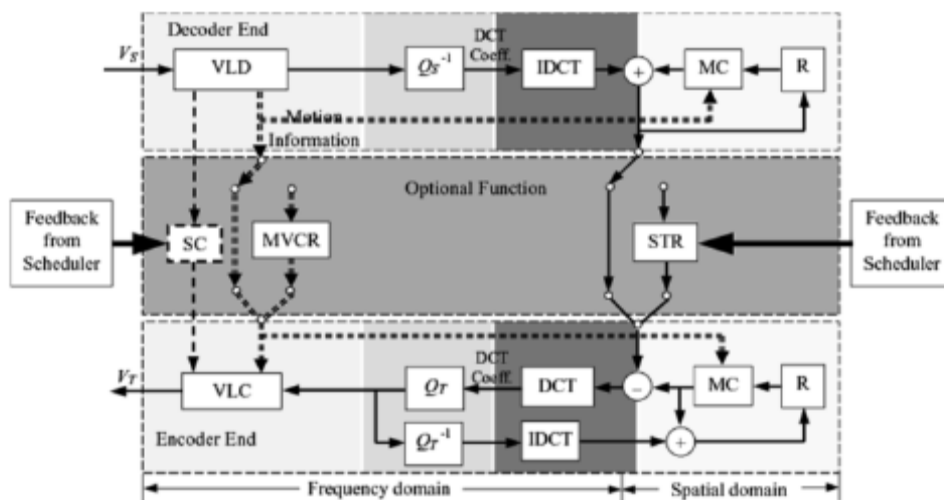


Figure 5.21 Heterogeneous video transcoder [12]

5.3 Summary

This chapter describes different video transcoding architectures and the issues related to efficient transcoding. The next chapter describes the proposed architecture and the reasons for the choice of the same.

CHAPTER 6

PROPOSED TRANSCODER

6.1 Introduction

Transcoding H.264 to VC-1 is a heterogeneous transcoding operation. Hence, it is important to map the differences between the two standards. The reference transcoder, the reasons for choosing the proposed transcoder architecture, and the mapping between the two standards are described in the following sections.

6.2 Cascaded reference transcoder

The simplest implementation of a transcoder as described in chapter 0 is to cascade the decoder and encoder to get the new bitstream. Since it involves complete decoding and re-encoding the complexity of such an implementation is very high. However the only error it has is from lossy encoding of already degraded video output for the decoder. This implementation is devoid of drift errors (Section 5.2.2.1). Being the simplest implementation of a transcoder, this architecture is used for the basis of complexity and output quality comparison. In the current research the first step is the implementation of a cascaded transcoding as described in Figure 6.1

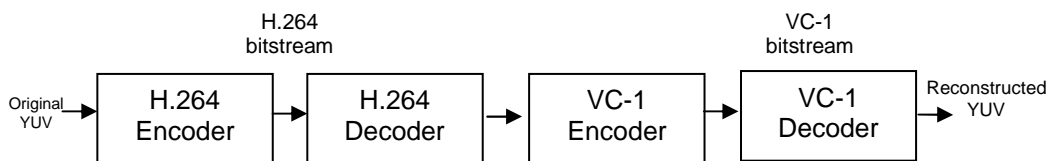


Figure 6.1 Cascade decoder and encoder

6.3 Choice of transcoding architecture

Transcoding can be performed in either spatial domain or frequency domain. Frequency domain transcoders (Section 5.2.4) have the main drawback that motion compensation in transform domain is very computationally intensive. Frequency domain transcoders are also less flexible compared to spatial domain transcoders [56] (Section

5.2.3). For H.264 to VC-1 standards transcoding, it is required to implement several changes in order to accommodate the mismatches between the two standards. For instance, for motion estimation and compensation, H.264 supports 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 macroblock partitions (Section 2.3.2), but VC-1 supports 16x16 and 8x8 only (Section 3.4.8). The transform size and type - 8x8 and 4x4 in H.264 (Section 2.3.3) and 8x8, 4x8, 8x4 and 4x4 in VC-1 (Section 3.4.3) are different and make transform domain transcoding prohibitively complex. Hence, the use of frequency domain transcoders is poor for heterogeneous transcoding and pixel domain is preferred.

6.4 Proposed transcoder

The transcoding algorithm considered in this research assumes full H.264 decoding down to the pixel level, followed by a reduced complexity VC-1 encoding. The data gathered during the H.264 decoding stage is used to accelerate the VC-1 encoding stage. It is assumed that the H.264 encoded bitstream is generated with a rate-distortion (R-D) optimized encoder. The picture coding types used are similar in both the standards since the profiles considered in this research are limited to the baseline profile in H.264 and the simple profile in VC-1. The transform size and type in VC-1 are different from those used in H.264 and makes transform domain transcoding prohibitively complex. The semantics of intra MBs are similar except for the intra directional prediction allowed in H.264 and the mixed MBs in VC-1. The inter prediction has significant differences including the block size of the MC and the block size of the transform. These similarities and dissimilarities between the codecs can be exploited in reducing the transcoding complexity. Figure 6.2 illustrates the proposed transcoder architecture.

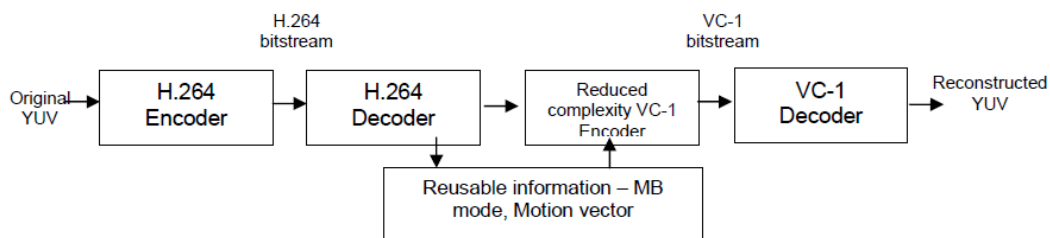


Figure 6.2 Proposed transcoder architecture

6.4.1 Intra MB Mode Mapping

An intra MB in the incoming H.264 bitstream is coded as a VC-1 intra MB. An H.264 intra MB can be coded as Intra 4x4 (9 different directional modes) or Intra 16x16 (4 different modes). But a VC-1 intra MB has four 8x8 blocks and has no prediction modes. Since intra MB in VC-1 uses an 8x8 transform, irrespective of the block size (16x16 or 4x4) in H.264, we need not carry over the information of the intra prediction type in H.264. Table 6.1 shows the proposed intra MB mapping.

Table 6.1 H.264 and VC-1 Intra MB mapping

H.264 Intra MB	VC-1 Intra MB
Intra 16x16 (Any mode)	Intra MB 8x8
Intra 4x4 (Any mode)	Intra MB 8x8

6.4.2 Inter MB Mode Mapping

An inter coded MB in the incoming H.264 bitstream is coded as inter MB in VC-1. The inter MB in H.264 has 7 different motion compensation sizes – 16x16, 16x8, 8x16, 8x8, 4x8, 8x4, 4x4. The inter MB in VC-1 has 2 different motion compensation sizes 16x16 and 8x8. Another significant difference is that H.264 uses 4x4 (and 8x8 in fidelity range extensions) transform sizes where as VC-1 uses 4 different transform sizes – 8x8, 4x8, 8x4 and 4x4. The 16x16 motion compensation sizes are usually selected in H.264 for areas that are relatively uniform and will be mapped to inter 16x16 MB in VC-1 with a transform size of 8x8. Motion compensation sizes 8x16, 16x8 have small non-uniform motion and hence they are mapped to inter 8x8 MB in VC-1 since 16x16 MB size will yield worse quality due to the non-uniform motion. Using the selected H.264 block size as a measure of homogeneity in the block, the transform size is determined and applied in VC-1. In other words, the H.264 block size determines the transform size used for that particular block. This method eliminates the need to compute the half sum and half difference values of each 8x8 block to determine the transform size.

The 8x8, 8x4, 4x8 and 4x4 modes are usually selected in H.264 for areas that have non-uniform motion. The 16x16 mode in VC-1 is eliminated for such non-uniform MBs. The

MB is then mapped to an 8x8 block size in VC-1 with the H.264 block size determining the transform size to be used in VC-1.

Table 6.2 describes the decision making for mapping the inter MBs and the type of transform to be used in VC-1.

Table 6.2 H.264 and VC-1 Inter MB mapping and VC-1 transform type

H.264 Inter MB	VC-1 Inter MB	Transform size in VC-1
Inter 16x16	Inter 16x16	8x8
Inter 16x8	Inter 8x8	8x4
Inter 8x16	Inter 8x8	4x8
Inter 8x8	Inter 8x8	8x8
Inter 4x8	Inter 8x8	4x8
Inter 8x4	Inter 8x8	8x4
Inter 4x4	Inter 8x8	4x4

6.4.3 Motion vector mapping

Re-use of motion vectors selected in H.264 can significantly reduce the complexity of VC-1 encoding. Section 5.2.5.4 describes motion vector selection and refinement. Since the proposed transcoder does format conversion from H.264 to VC-1, median motion vectors (Section 5.2.5.4) are selected when there is more than 1 motion vector in the incoming H.264 macroblock. Table 6.3 describes the selection of motion vectors.

Table 6.3 H.264 and VC-1 Inter MB motion vector mapping

H.264 Inter MB	VC-1 Inter MB	Motion Vector Re-use
Inter 16x16	Inter 16x16	Same motion vectors for 16x16 block
Inter 16x8	Inter 8x8	Median of motion vectors for each 8x8 block
Inter 8x16	Inter 8x8	Median of motion vectors for each 8x8 block
Inter 8x8	Inter 8x8	Same motion vectors for each 8x8 block
Inter 4x8	Inter 8x8	Median of motion vectors for each 8x8 block
Inter 8x4	Inter 8x8	Median of motion vectors for each 8x8 block
Inter 4x4	Inter 8x8	Median of motion vectors for each 8x8 block

Except for the case of inter 16x16 mode and inter 8x8, It is necessary to choose one motion vector per 8x8 block from a number of available motion vectors. The median motion vector of the available motion vectors is chosen in this research. Figure 6.3 illustrates the different possible combinations.

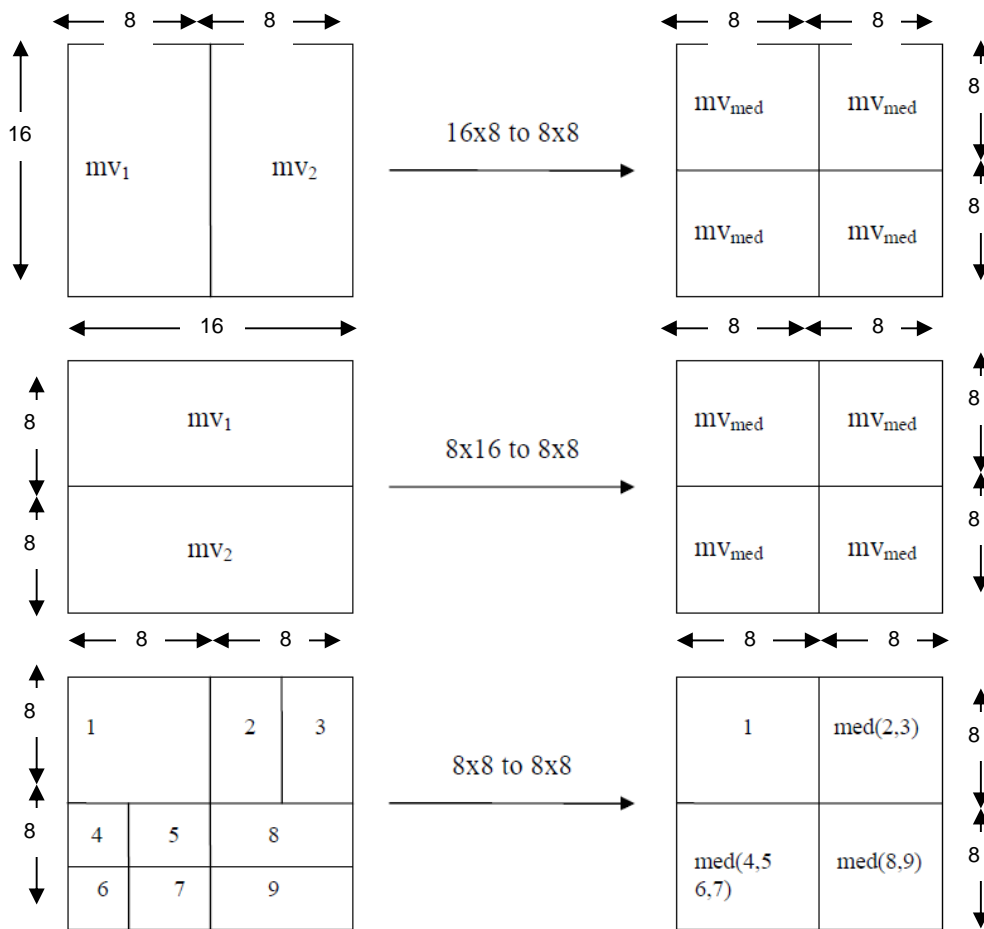


Figure 6.3 Selection of motion vectors in different cases

In the 16x8 and 8x16 motion vector selection, mv_{med} is the median motion vector of mv_1 and mv_2 . In the 8x8 to 8x8 motion vector selection case, the median of the sub block's motion vector is used.

6.4.4 Reference Pictures:

The H.264/AVC standard defines the use of up to sixteen reference pictures for motion estimation, while VC-1 uses only one or two, according to the slice type P or B respectively. The reuse of motion vectors implies using the same reference pictures to maintain their meaning. Since the profiles considered are baseline profile for H.264 (single reference picture) and simple profile for VC-1 (single reference picture), the same reference picture as the incoming bitstream is used and no re-scaling of motion vectors is required.

6.4.5 Skipped Macroblock

When a skipped macro block is signaled in the bit stream, no further data is sent for that macro block. The mode conversion of skipped macroblocks in H.264 to skipped macroblocks in VC-1 is a straight forward process. Since the skip macro block definition of both standards is fully compatible, a direct conversion is possible.

6.4.6 Flowchart of proposed transcoder

Figure 6.4 illustrates the proposed transcoder flow. The macroblock coding modes mapping is described.

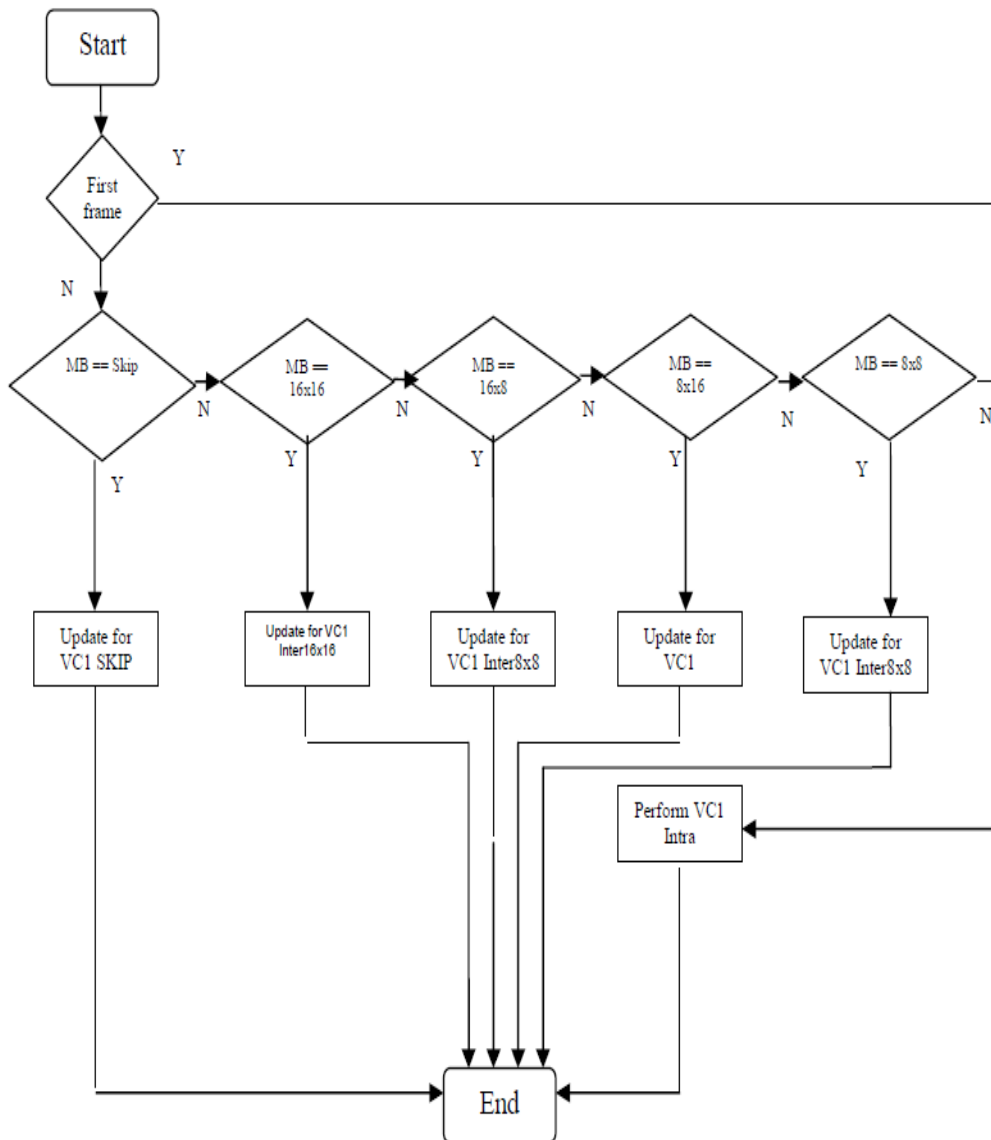


Figure 6.4 Proposed transcoder flowchart

6.4.7 Extraction of re-usable information

The H.264 bitstream has the macroblock type, sub-macroblock type, reference picture index and motion vectors (if applicable). These details are extracted out of the bitstream and written to a data file. This information is used while encoding in VC-1.

6.4.7.1 Extracted details

H.264 bitstream contains information about

1. Macroblock type
 - P16x16 – P block type 16x16
 - P16x8 – P block type 16x8
 - P8x16 – P block type 8x16
 - P8x8 – P block type 8x8
 - I4MB – I block type 16x16
 - I16MB – I block type 16x16
2. Macroblock sub block type
 - SMB8x8 – sub macroblock type 8x8
 - SMB8x4 – sub macroblock type 8x4
 - SMB4x8 – sub macroblock type 4x8
 - SMB4x4 – sub macroblock type 4x4
3. Reference picture index
4. Motion vector x, y

Figure 6.5 gives a sample screen shot of the extracted information.

```

transcoder_mb_type - WordPad
File Edit View Insert Format Help
mb_no 0
mb_type 9
mb_no 1
mb_type 9
mb_no 2
mb_type 9
mb_no 3
mb_type 9
mb_no 4
mb_type 9
mb_no 5
mb_type 9
mb_no 6
mb_type 9
mb_no 7
mb_type 9
mb_no 8
mb_type 9
mb_no 9
mb_type 9
mb_no 10
mb_type 9
mb_no 11
mb_type 9
mb_no 12
mb_type 9
mb_no 13
mb_type 9
mb_no 14
mb_type 9
mb_type _

```

(a)

```

transcoder_mb_type - WordPad
File Edit View Insert Format Help
mb_no 0
mb_type 3
mb_ref_pic_idx 0
mb_motion_vector_x 0
mb_motion_vector_y 1
mb_ref_pic_idx 0
mb_motion_vector_x 1
mb_motion_vector_y 0
mb_no 1
mb_type 1
mb_ref_pic_idx 0
mb_motion_vector_x 0
mb_motion_vector_y 0
mb_no 2
mb_type 1
mb_ref_pic_idx 0
mb_motion_vector_x 0
mb_motion_vector_y 0
mb_no 3
mb_type 1
mb_ref_pic_idx 0
mb_motion_vector_x 0
mb_motion_vector_y 0
mb_no 4
mb_type 1
mb_ref_pic_idx 0
mb_motion_vector_x 0
mb_motion_vector_y 0
mb_no 5

```

(b)

Figure 6.5 Sample extracted information from H.264 bitstream (a) I frame (b) P frame

6.4.8 Simplified VC-1 Encoder

A pseudo code that described how the complexity of the VC-1 encoder is reduced by re-using the extracted information from the H.264 bit stream.

```
#ifdef H264VC1TRANSCODER
    if((mbType == I4MB) || (mbType == I16MB))
    {
        // update block types for each 8x8 block
    }
    else if((mbType == P16x16) || (mbType == P16x8) || (mbType == P8x16) || (mbType
== PSKIP))
    {
        if(mbType == P16x16)
            // update all 4 8x8 block types as 1 MV MB
            // update the motion vectors from the input file
        if(mbType == P16x8)
            // update all 4 8x8 block types as 4 MV MB
            // compute the median MV from input file
            // update the motion vectors as the median MV
        if(mbType == P8x16)
            // update all 4 8x8 block types as 4 MV MB
            // compute the median MV from input file
            // update the motion vectors as the median MV
        if(mbType == P8x8)
            // update all 4 8x8 block types as 4 MV MB
            // compute the median MV from input file depending on the sub
            macroblock type
            // update the motion vectors as the median MV
        if(mbType == PSKIP)
            // update all 4 8x8 block types as skip
            // update MV as the predicted MV
    }
#endif
```

6.4.9 Implementation and Results

6.4.9.1 Implementation

The implementation of the transcoder consisted of two steps; modify the H.264 decoder to output the required information and then modify the VC-1 encoder to re-use this information. The H.264 codec used was the joint module implementation of the H.264 standard [70] and the VC-1 software was the SMPTE implementation [71]. Both the codecs were implemented in the C programming language.

6.4.9.2 Results

Five test sequences are chosen for testing purposes. They are Akiyo (low motion QCIF sequence), Miss America (low motion, QCIF sequence), Foreman (medium-high

motion, CIF sequence), Football (high motion, CIF sequence) and Mobile (high detail, high motion CIF sequence).

6.4.9.2.1 Comparison of proposed transcoder with cascaded transcoder with respect to H.264 video

A comparison of the MSE, PSNR, SSIM, bit stream file size and encoding times between the proposed architecture and the cascaded re-encoding transcoder are tabulated for various quantization parameters (QP) and illustrated in Sections 6.4.9.2.1.1 through 6.4.9.2.1.5. The reference video used for all the comparison purpose is the H.264 decoded video. This is the available video to compare in real time applications when the original video may or may not be available.

6.4.9.2.1.1 Test sequence: Akiyo, QCIF Resolution, 10 frames

Table 6.4 Transcoder results for Akiyo QCIF sequence, cascade transcoder Vs proposed transcoder

QP	Open loop bitstream (kb)	Transcoded bitstream (kb)	% change in file size	Metrics type	Open loop	Transcoder	% Change
10	28551.00	28419.00	-0.46	Y MSE	2.29	2.26	-1.31
				U MSE	1.77	1.73	-2.26
				V MSE	1.38	1.37	-0.72
				Y PSNR	44.54	44.59	0.11
				U PSNR	45.65	45.76	0.24
				V PSNR	46.75	46.76	0.02
				Y SSIM	0.99	0.99	0.00
				U SSIM	0.99	0.99	0.00
15	7847.00	7378.00	-5.98	V SSIM	0.99	0.99	0.00
				Y MSE	5.08	5.15	1.38
				U MSE	3.41	3.40	-0.29
				V MSE	2.50	2.48	-0.80
				Y PSNR	41.07	41.02	-0.12
				U PSNR	42.80	42.82	0.05
				V PSNR	44.16	44.18	0.05
				Y SSIM	0.98	0.98	0.00
20	6510.00	5829.00	-10.46	U SSIM	0.98	0.98	0.00
				V SSIM	0.98	0.98	0.00
				Y MSE	8.66	8.87	2.42
				U MSE	5.73	5.55	-3.14
				V MSE	3.87	3.80	-1.81
				Y PSNR	38.75	38.65	-0.26
				U PSNR	40.55	40.69	0.35
				V PSNR	42.25	42.33	0.19
28	3517.00	2405.00	-31.62	Y SSIM	0.96	0.96	0.00
				U SSIM	0.97	0.97	0.00
				V SSIM	0.97	0.97	0.00
				Y MSE	17.88	18.72	4.70
				U MSE	11.56	10.62	-8.13
				V MSE	6.45	6.17	-4.34
				Y PSNR	35.61	35.41	-0.56
				U PSNR	37.50	37.87	0.99
35	3353.00	1563.00	-53.39	V PSNR	40.04	40.23	0.47
				Y SSIM	0.95	0.94	-1.05
				U SSIM	0.96	0.96	0.00
				V SSIM	0.95	0.95	0.00
				Y MSE	30.13	31.74	5.34
				U MSE	20.07	17.92	-10.71
				V MSE	10.85	10.33	-4.79
				Y PSNR	33.35	33.12	-0.69
35	3353.00	1563.00	-53.39	U PSNR	35.11	35.60	1.40
				V PSNR	37.78	37.99	0.56
				Y SSIM	0.92	0.91	-1.09
				U SSIM	0.94	0.95	1.06
35	3353.00	1563.00	-53.39	V SSIM	0.94	0.94	0.00

Table 6.5 Transcoder results for Akiyo QCIF sequence, Time saving, cascade transcoder Vs proposed transcoder

QP	Open loop encoding time (s)	Transcoded encoding time (s)	Time saving %
10	9	1	89
15	10	1	90
20	11	1	91
28	11	1	91
36	12	1	92

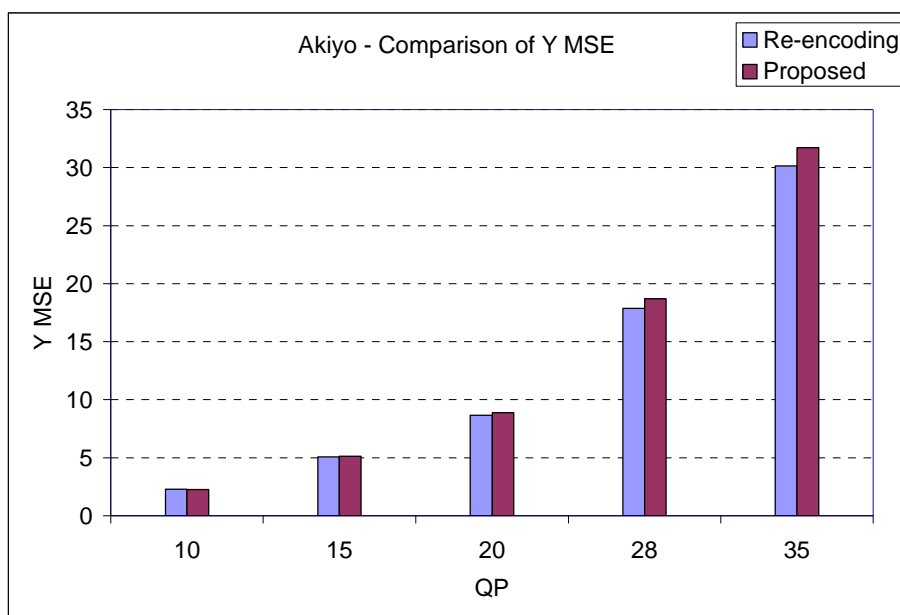


Figure 6.6 Transcoder Results - Akiyo, QCIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder

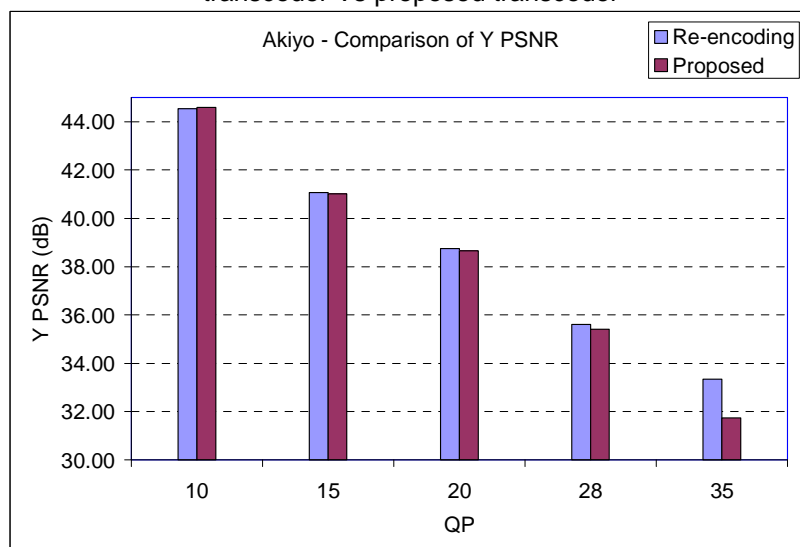


Figure 6.7 Transcoder Results - Akiyo, QCIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder

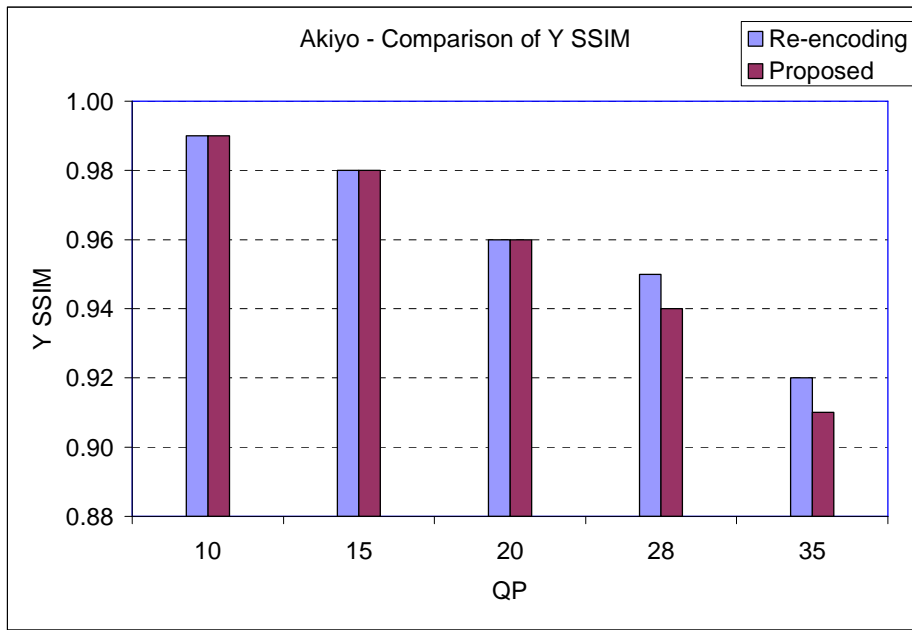


Figure 6.8 Transcoder Results - Akiyo, QCIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder

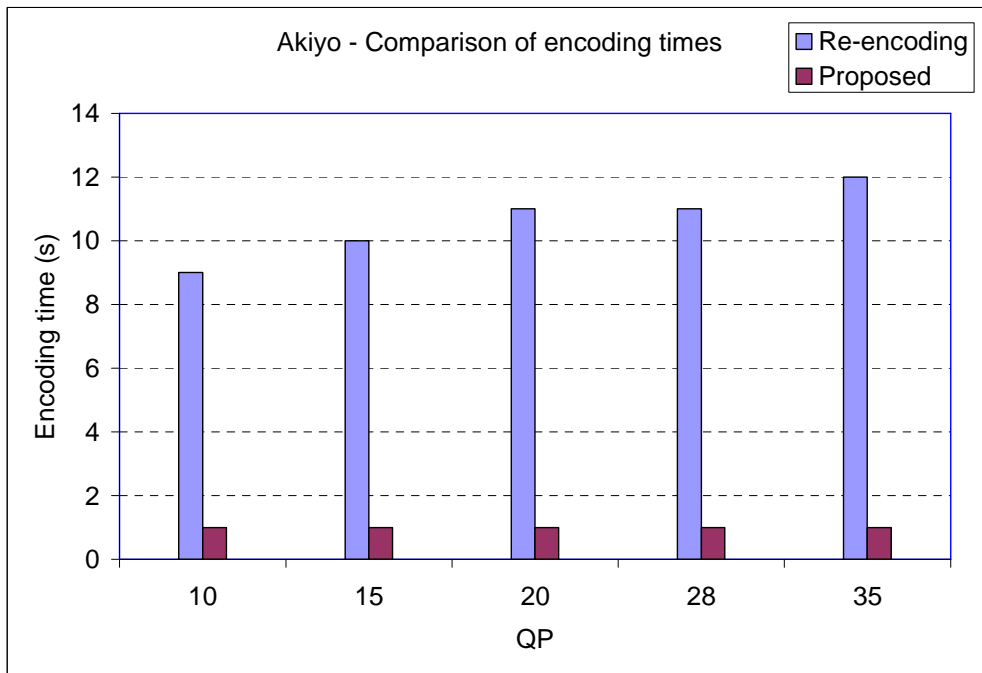


Figure 6.9 Transcoder Results - Akiyo, QCIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder

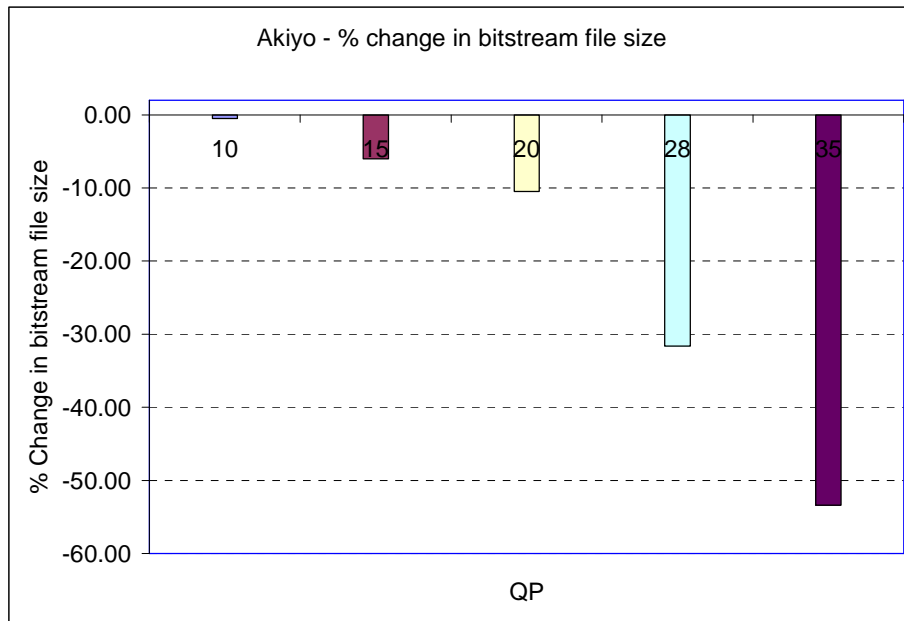


Figure 6.10 Transcoder Results - Akiyo, QCIF, Percentage change in bitstream file size, cascade transcoder Vs proposed transcoder

Figure 6.11 illustrates the test sequences results.

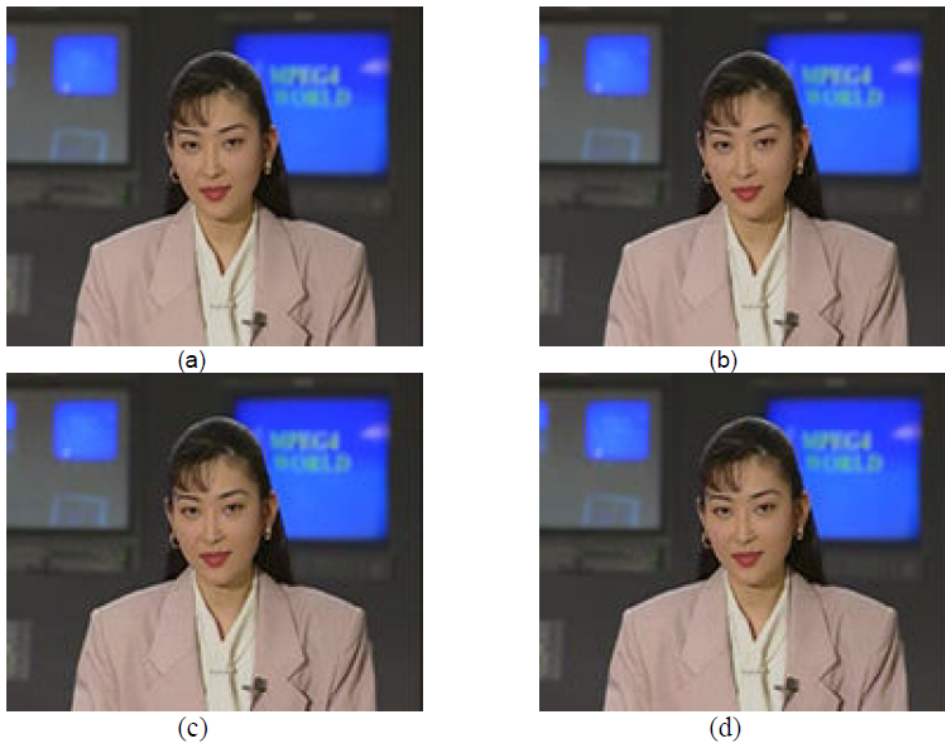


Figure 6.11 Akiyo sequence (a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10

6.4.9.2.1.2 Test sequence: Miss America, QCIF Resolution, 10 frames

Table 6.6 Transcoder results for Miss America QCIF sequence, cascade transcoder Vs proposed transcoder

QP	Open loop bitstream (kb)	Transcoded bitstream (kb)	% change in file size	Metrics type	Open loop	Transcoder	% Change
10	35626.00	34652.00	-2.73	Y MSE	1.95	1.86	-4.62
				U MSE	2.86	2.71	-5.24
				V MSE	2.23	2.14	-4.04
				Y PSNR	45.24	45.44	0.44
				U PSNR	43.57	43.81	0.55
				V PSNR	44.65	44.83	0.40
				Y SSIM	0.98	0.98	0.00
				U SSIM	0.96	0.97	1.04
				V SSIM	0.98	0.98	0.00
15	12867.00	12831.00	-0.28	Y MSE	3.37	3.16	-6.23
				U MSE	4.46	4.36	-2.24
				V MSE	3.68	3.44	-6.52
				Y PSNR	42.85	43.14	0.68
				U PSNR	41.66	41.75	0.22
				V PSNR	42.48	42.77	0.68
				Y SSIM	0.98	0.98	0.00
				U SSIM	0.95	0.95	0.00
				V SSIM	0.97	0.97	0.00
20	7948.00	7981.00	0.42	Y MSE	5.13	4.63	-9.75
				U MSE	4.88	4.58	-6.15
				V MSE	5.41	5.10	-5.73
				Y PSNR	41.03	41.48	1.10
				U PSNR	41.25	41.53	0.68
				V PSNR	40.80	41.06	0.64
				Y SSIM	0.97	0.97	0.00
				U SSIM	0.95	0.95	0.00
				V SSIM	0.97	0.97	0.00
28	2633.00	2608.00	-0.95	Y MSE	9.20	7.85	-14.67
				U MSE	4.78	4.75	-0.63
				V MSE	7.86	8.21	4.45
				Y PSNR	38.49	39.19	1.82
				U PSNR	41.33	41.37	0.10
				V PSNR	39.18	38.99	-0.48
				Y SSIM	0.96	0.96	0.00
				U SSIM	0.96	0.96	0.00
				V SSIM	0.96	0.96	0.00
35	1371.00	1374.00	0.22	Y MSE	14.49	12.35	-14.77
				U MSE	3.91	3.99	2.05
				V MSE	13.43	14.55	8.34
				Y PSNR	36.52	37.22	1.92
				U PSNR	42.22	42.15	-0.17
				V PSNR	36.86	36.51	-0.95
				Y SSIM	0.95	0.95	0.00
				U SSIM	0.97	0.97	0.00
				V SSIM	0.94	0.93	-1.06

Table 6.7 Transcoder results for Miss America QCIF sequence, Time saving, cascade transcoder Vs proposed transcoder

QP	Open loop encoding time (s)	Transcoded encoding time (s)	Time saving %
10	15	1	93
15	15	1	93
20	15	1	93
28	15	1	93
36	15	1	93

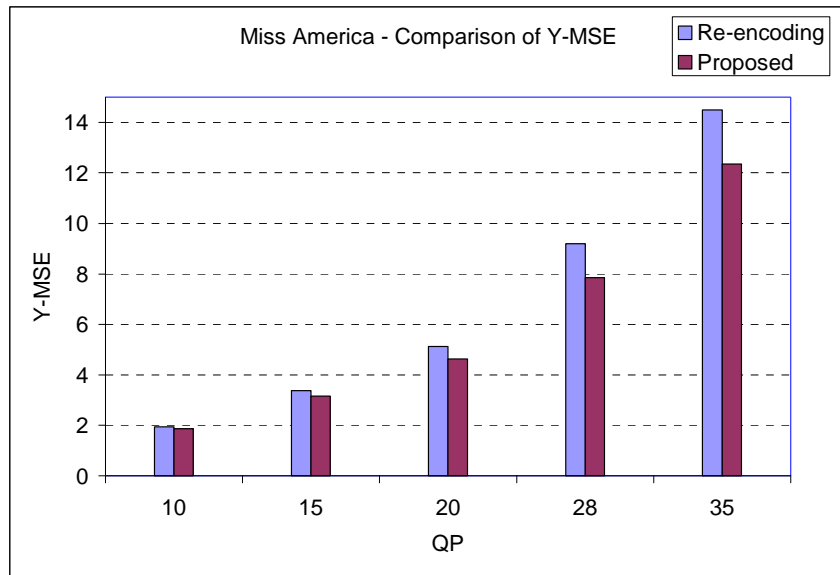


Figure 6.12 Transcoder Results – Miss America, QCIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder

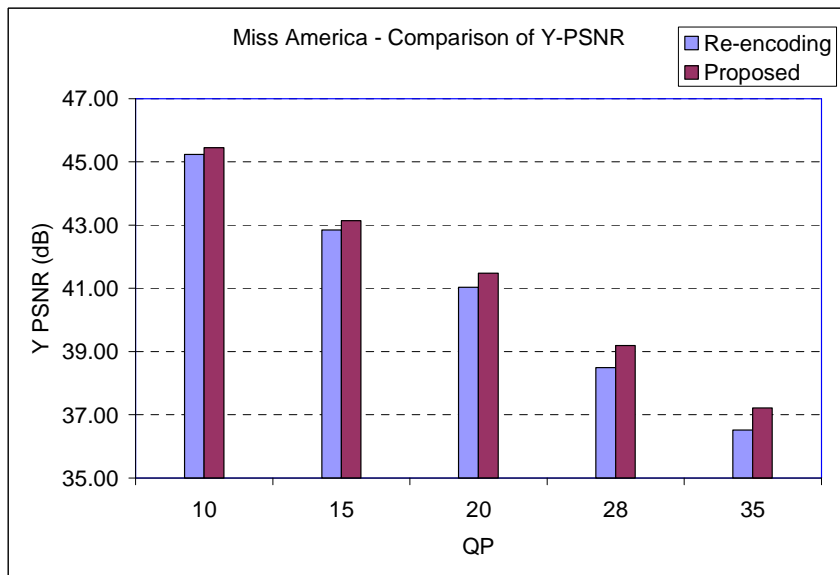


Figure 6.13 Transcoder Results – Miss America, QCIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder

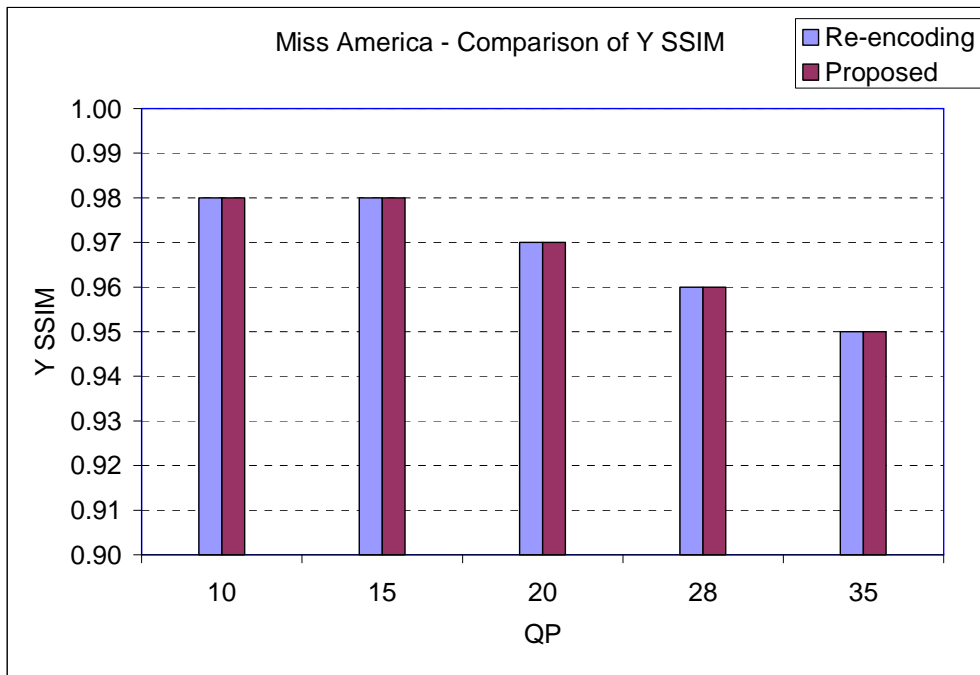


Figure 6.14 Transcoder Results – Miss America, QCIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder

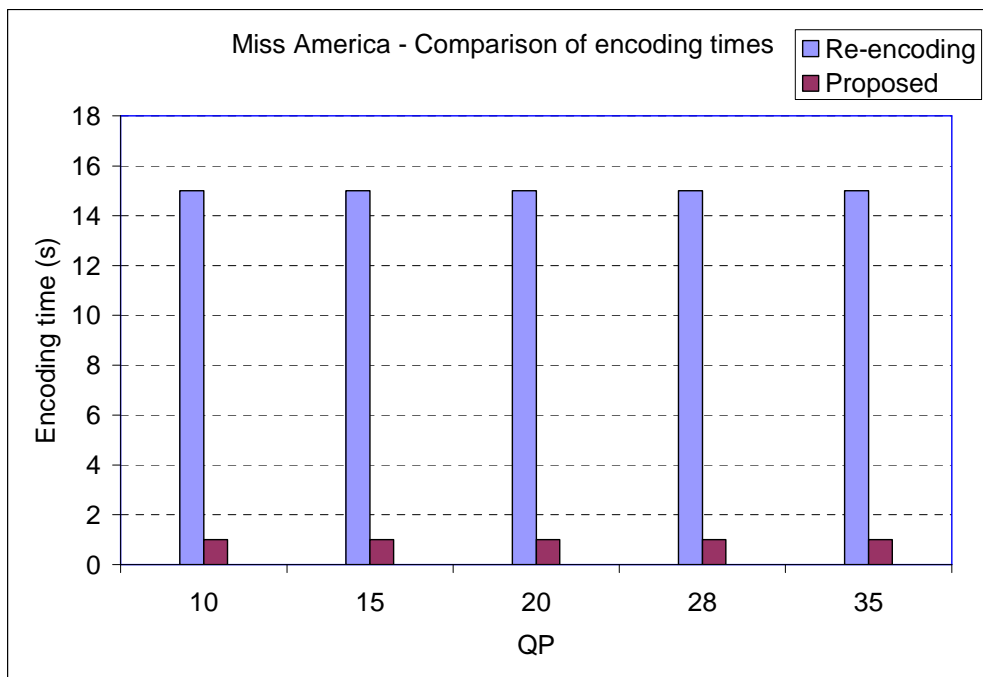


Figure 6.15 Transcoder Results – Miss America, QCIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder

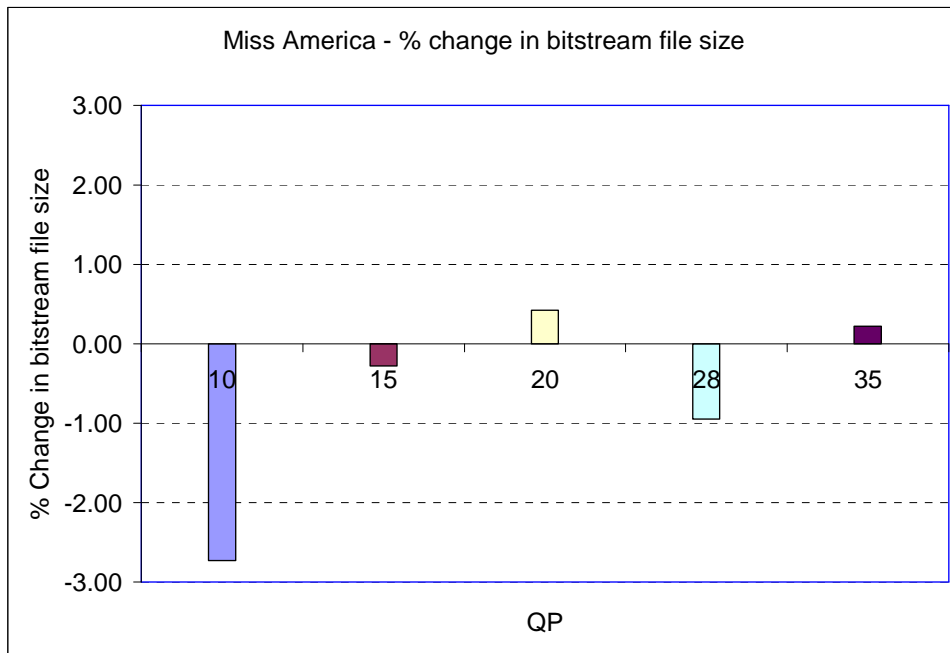


Figure 6.16 Transcoder Results – Miss America, QCIF, Percentage change in Y bitstream file size, cascade transcoder Vs proposed transcoder

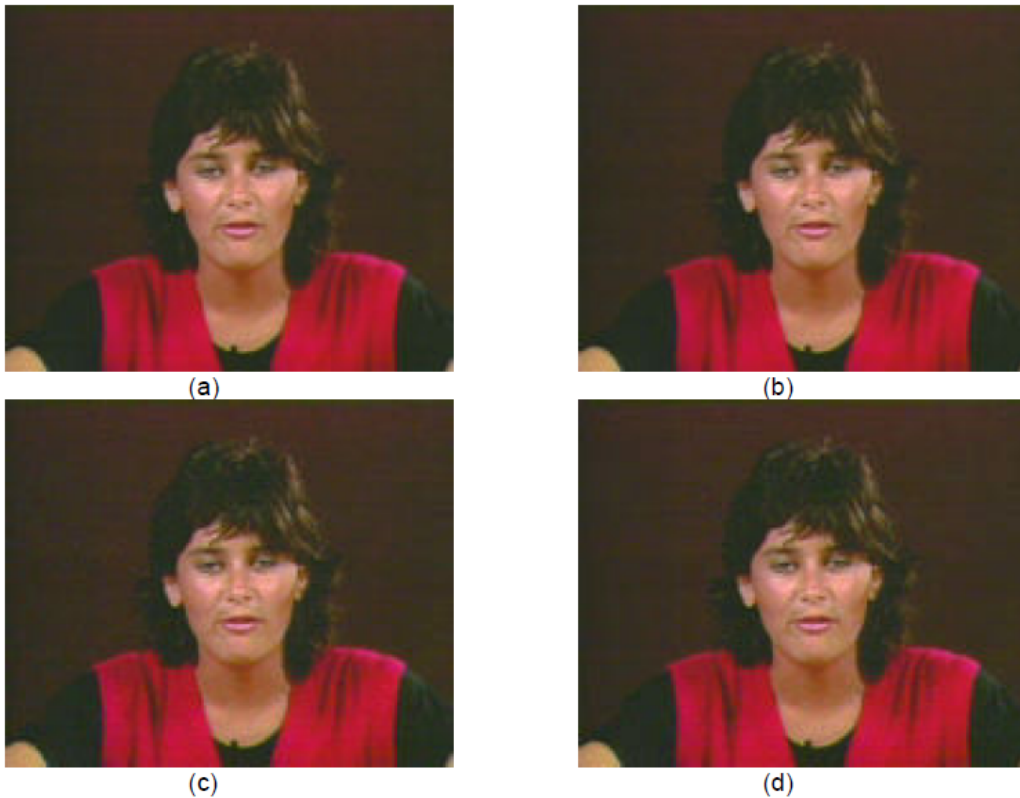


Figure 6.17 Miss America sequence (a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10

6.4.9.2.1.3 Test Sequence: Foreman, CIF Sequence, 10 frames

Table 6.8 Transcoder results for Foreman CIF sequence, cascade transcoder Vs proposed transcoder

QP	Open loop bitstream (kb)	Transcoded bitstream (kb)	% change in file size	Metrics type	Open loop	Transcoder	% Change
10	243503.00	236918.00	-2.70	Y MSE	3.57	3.33	-6.72
				U MSE	1.83	1.96	7.10
				V MSE	1.10	1.18	7.27
				Y PSNR	42.62	42.90	0.66
				U PSNR	45.52	45.24	-0.62
				V PSNR	47.72	47.48	-0.50
				Y SSIM	0.97	0.98	1.03
				U SSIM	0.98	0.98	0.00
				V SSIM	0.99	0.99	0.00
15	115937.00	112772.00	-2.73	Y MSE	7.76	7.21	-7.09
				U MSE	2.62	2.64	0.76
				V MSE	1.46	1.44	-1.37
				Y PSNR	39.24	39.55	0.79
				U PSNR	43.95	43.92	-0.07
				V PSNR	46.50	46.56	0.13
				Y SSIM	0.95	0.95	0.00
				U SSIM	0.97	0.97	0.00
				V SSIM	0.99	0.99	0.00
20	79991.00	75328.00	-5.83	Y MSE	11.83	11.22	-5.16
				U MSE	3.08	2.99	-2.92
				V MSE	1.94	1.94	0.00
				Y PSNR	37.41	37.64	0.61
				U PSNR	43.25	43.38	0.30
				V PSNR	45.25	45.27	0.04
				Y SSIM	0.94	0.94	0.00
				U SSIM	0.97	0.97	0.00
				V SSIM	0.98	0.98	0.00
28	39003.00	30616.00	-21.50	Y MSE	17.49	19.11	9.26
				U MSE	3.20	2.86	-10.63
				V MSE	3.04	2.74	-9.87
				Y PSNR	35.70	35.32	-1.06
				U PSNR	43.10	43.58	1.11
				V PSNR	43.32	43.77	1.04
				Y SSIM	0.94	0.92	-2.13
				U SSIM	0.97	0.98	1.03
				V SSIM	0.98	0.98	0.00
35	32784.00	14969.00	-54.34	Y MSE	23.96	31.02	29.47
				U MSE	4.32	3.50	-18.98
				V MSE	5.59	4.51	-19.32
				Y PSNR	34.36	33.22	-3.32
				U PSNR	41.82	42.70	2.10
				V PSNR	40.68	41.59	2.24
				Y SSIM	0.94	0.90	-4.26
				U SSIM	0.97	0.98	1.03
				V SSIM	0.97	0.98	1.03

Table 6.9 Transcoder results for Foreman CIF sequence, Time saving, cascade transcoder Vs proposed transcoder

QP	Open loop encoding time (s)	Transcoded encoding time (s)	Time saving %
10	59	5	92
15	58	5	91
20	56	4	93
28	58	4	93
36	63	3	95

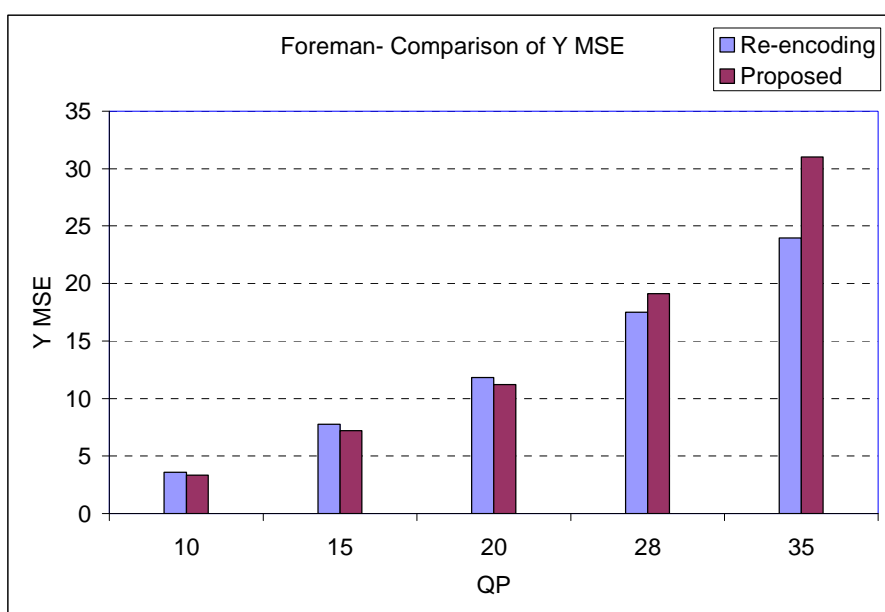


Figure 6.18 Transcoder Results – Foreman, CIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder

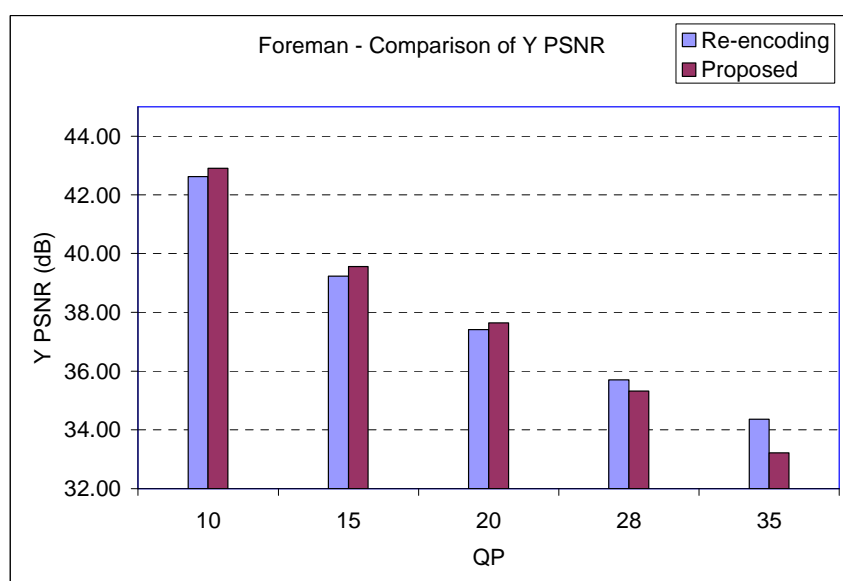


Figure 6.19 Transcoder Results – Foreman, CIF, Comparison of Y peak signal to noise ration, cascade transcoder Vs proposed transcoder

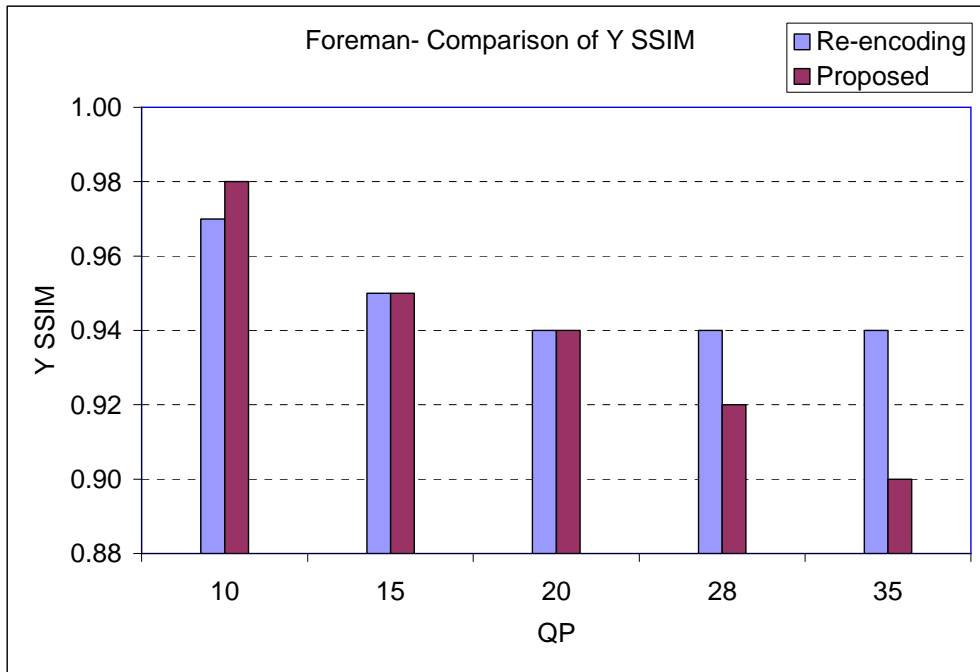


Figure 6.20 Transcoder Results – Foreman, CIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder

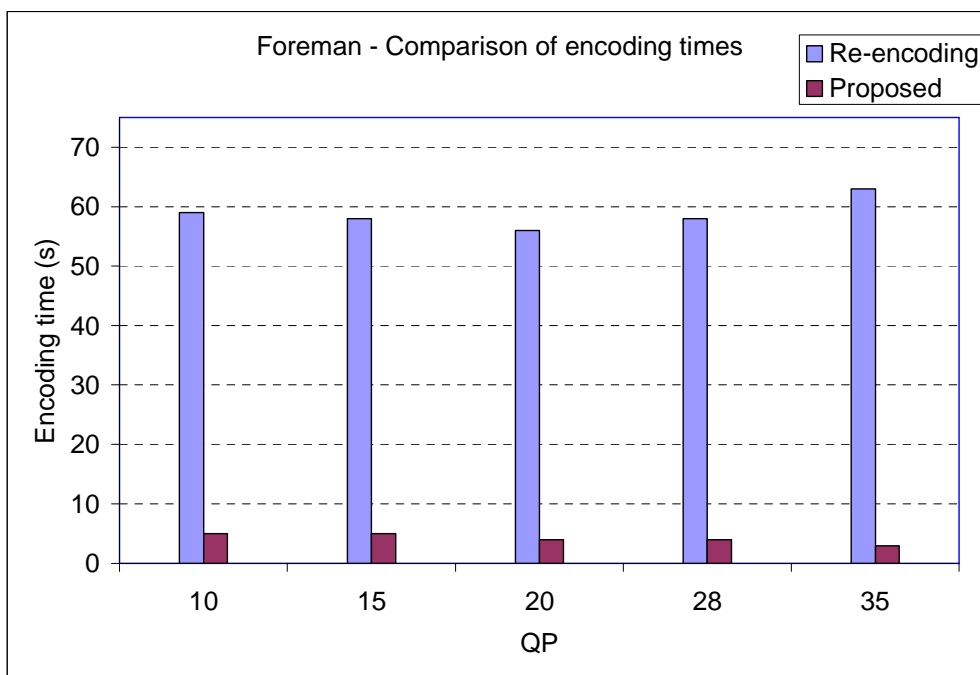


Figure 6.21 Transcoder Results – Foreman, CIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder

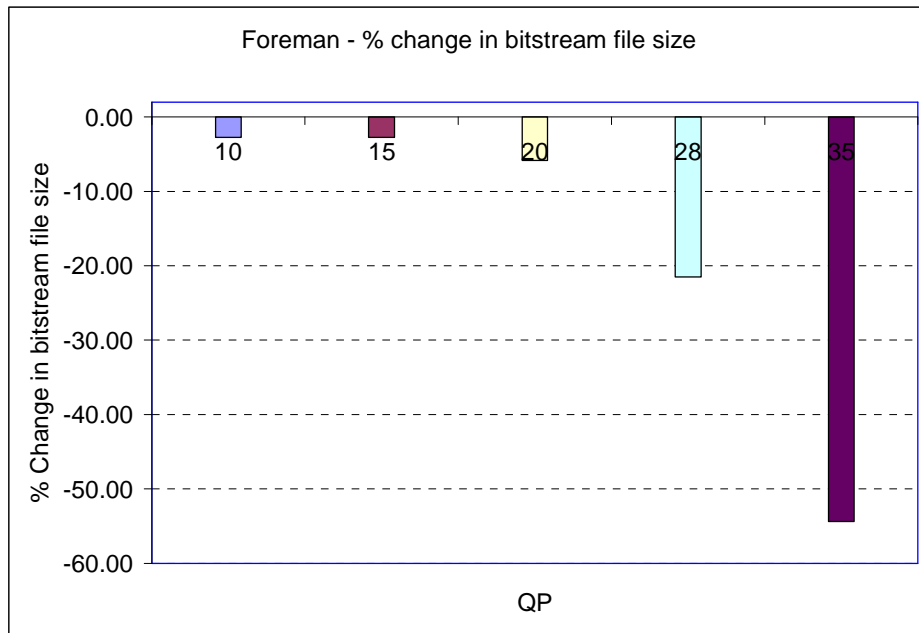


Figure 6.22 Transcoder Results – Foreman, CIF, Percentage change in Y mean bitstream file size, cascade transcoder Vs proposed transcoder



Figure 6.23 Foreman sequence (a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10

6.4.9.2.1.4 Test Sequence: Football, CIF Sequence, 10 frames

Table 6.10 Transcoder results for Football CIF sequence, cascade transcoder Vs proposed transcoder

QP	Open loop bitstream (kb)	Transcoded bitstream (kb)	% change in file size	Metrics type	Open loop	Transcoder	% Change
10	342460.00	339310.00	-0.92	Y MSE	3.90	3.74	-4.1
				U MSE	2.30	2.52	9.57
				V MSE	2.11	2.31	9.48
				Y PSNR	42.22	42.40	0.43
				U PSNR	44.53	44.16	-0.83
				V PSNR	44.90	44.53	-0.82
				Y SSIM	0.99	0.99	0
				U SSIM	0.98	0.98	0
15	198665.00	193011.00	-2.85	Y MSE	10.37	9.43	-9.06
				U MSE	4.24	4.32	1.89
				V MSE	3.41	3.49	2.35
				Y PSNR	37.98	38.39	1.08
				U PSNR	41.87	41.79	-0.19
				V PSNR	42.82	42.72	-0.23
				Y SSIM	0.96	0.97	1.04
				U SSIM	0.96	0.96	0
20	149883.00	143584.00	-4.2	Y MSE	19.40	17.20	-11.34
				U MSE	6.31	6.28	-0.48
				V MSE	4.48	4.41	-1.56
				Y PSNR	35.26	35.78	1.47
				U PSNR	40.15	40.16	0.02
				V PSNR	41.63	41.69	0.14
				Y SSIM	0.94	0.95	1.06
				U SSIM	0.95	0.95	0
28	79887.00	77521.00	-2.96	Y MSE	41.75	36.68	-12.14
				U MSE	10.91	10.49	-3.85
				V MSE	6.02	5.81	-3.49
				Y PSNR	31.94	32.49	1.72
				U PSNR	37.77	37.94	0.45
				V PSNR	40.35	40.50	0.37
				Y SSIM	0.88	0.89	1.14
				U SSIM	0.93	0.94	1.08
35	47257.00	40065.00	-15.22	Y MSE	68.63	64.08	-6.63
				U MSE	18.06	16.31	-9.69
				V MSE	8.66	7.48	-13.63
				Y PSNR	29.77	30.07	1.01
				U PSNR	35.59	36.02	1.21
				V PSNR	38.78	39.40	1.6
				Y SSIM	0.87	0.85	-2.3
				U SSIM	0.90	0.92	2.22
				V SSIM	0.94	0.95	1.06

Table 6.11 Transcoder results for Football CIF sequence, Time saving, cascade transcoder Vs proposed transcoder

QP	Open loop encoding time (s)	Transcoded encoding time (s)	Time saving %
10	65	11	83
15	64	9	86
20	68	6	91
28	71	4	94
36	72	4	94

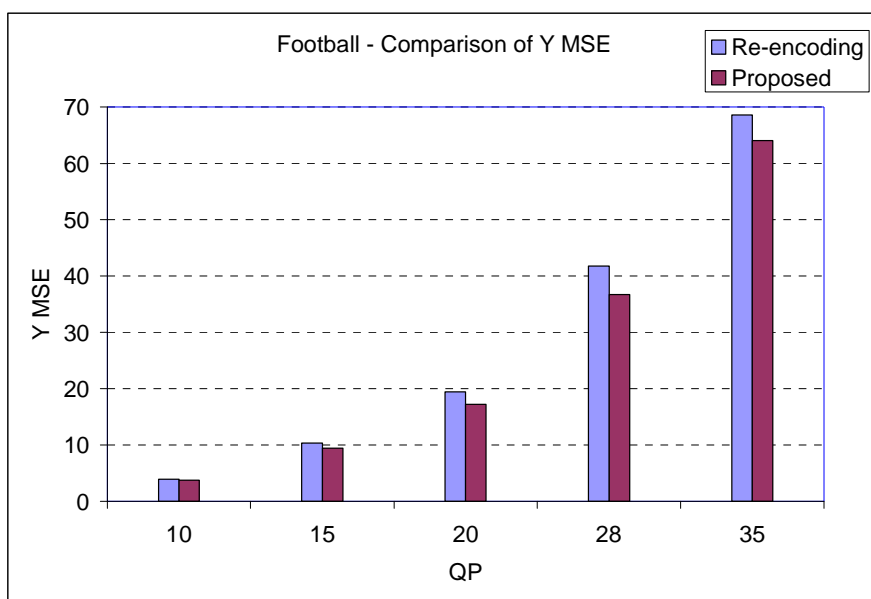


Figure 6.24 Transcoder Results – Football, CIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder

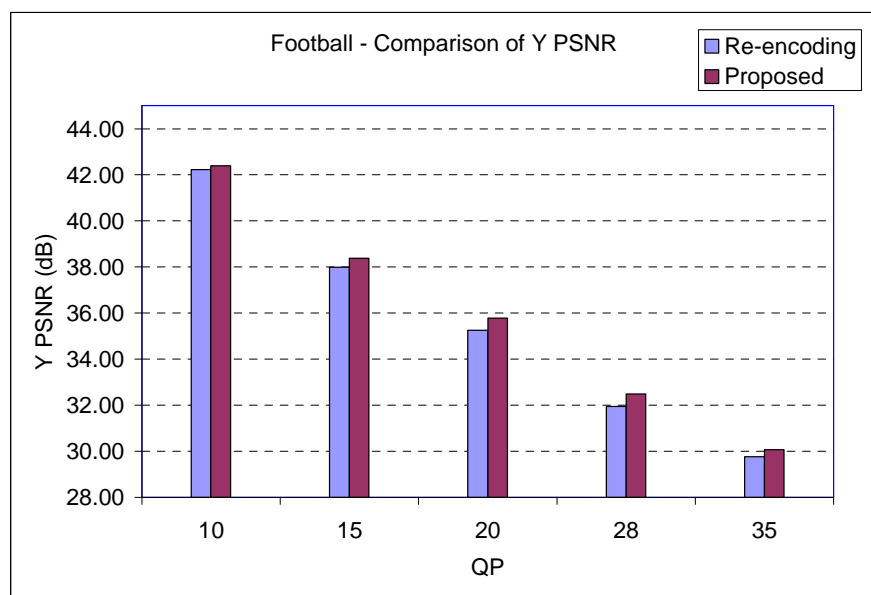


Figure 6.25 Transcoder Results – Football, CIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder

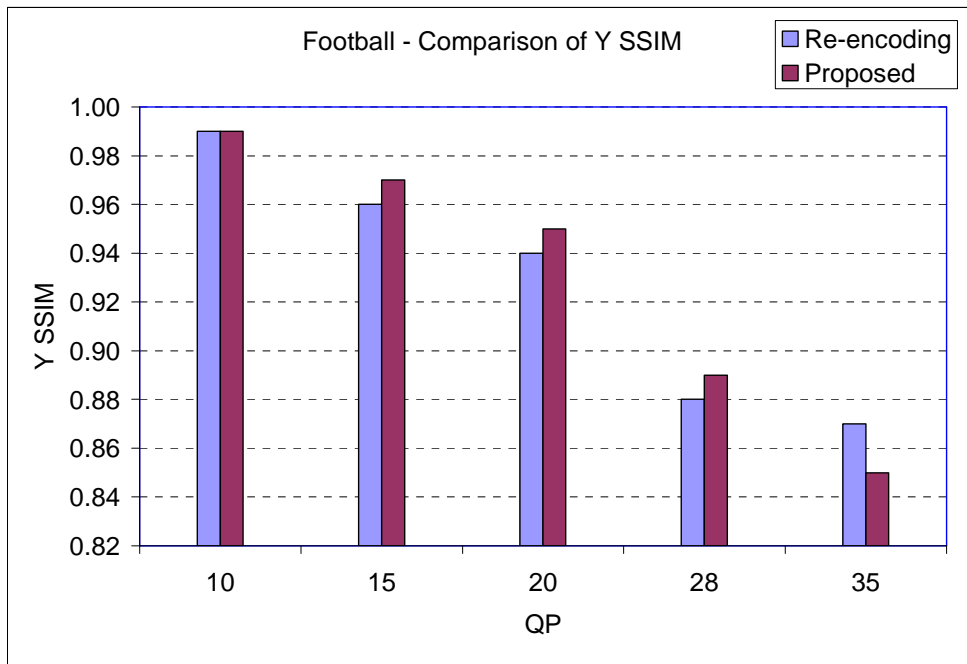


Figure 6.26 Transcoder Results – Football, CIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder

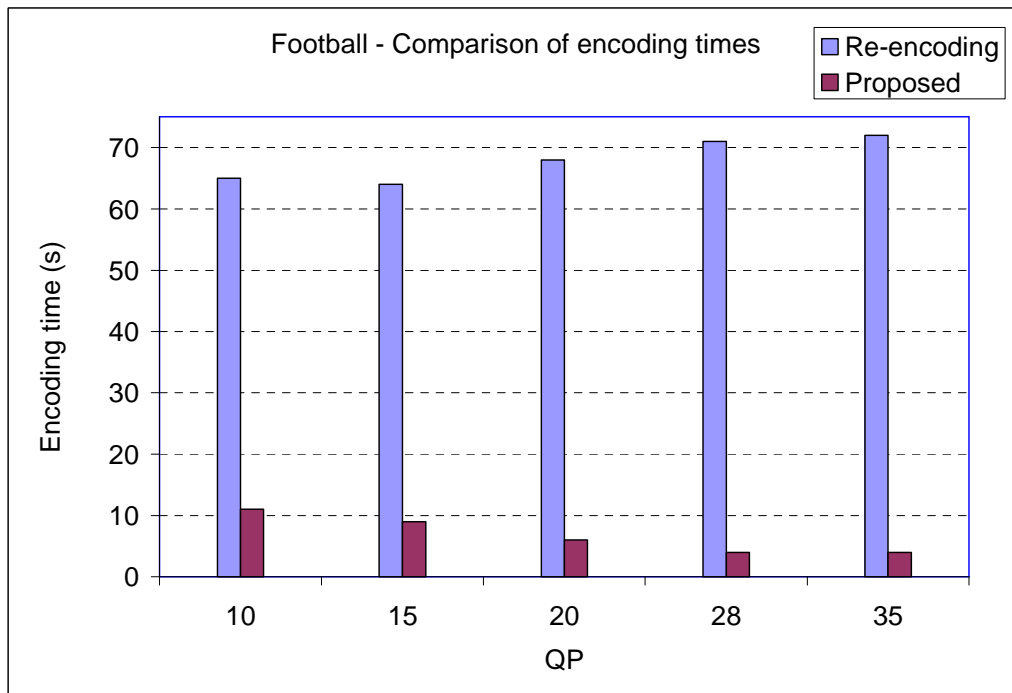


Figure 6.27 Transcoder Results – Football, CIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder

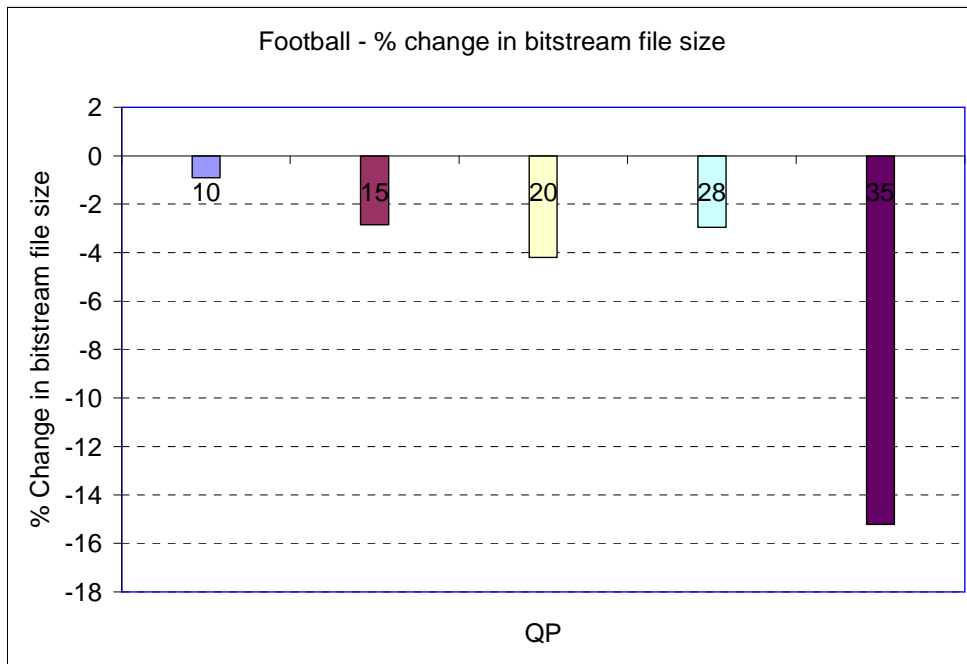


Figure 6.28 Transcoder Results – Football, CIF, Percentage change in bitstream file size, cascade transcoder Vs proposed transcoder

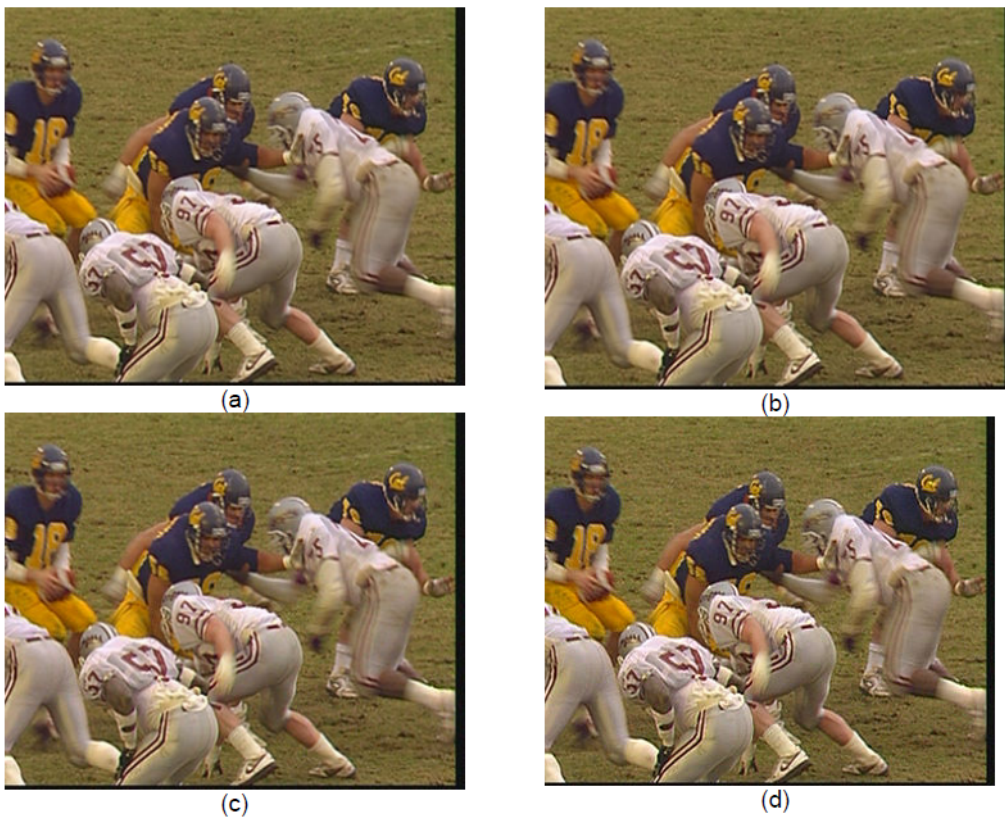


Figure 6.29 Football sequence (a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10

6.4.9.2.1.5 Test Sequence: Mobile, CIF Sequence, 10 frames

Table 6.12 Transcoder results for Mobile CIF sequence, cascade transcoder Vs proposed transcoder

QP	Open loop bitstream (kb)	Transcoded bitstream (kb)	% change in file size	Metrics type	Open loop	Transcoder	% Change
10	461468.00	467886.00	1.39	Y MSE	3.92	3.92	0
				U MSE	3.12	3.75	20.19
				V MSE	3.12	3.70	18.59
				Y PSNR	42.20	42.20	0
				U PSNR	43.19	42.44	-1.74
				V PSNR	43.20	42.49	-1.64
				Y SSIM	0.99	0.99	0
				U SSIM	0.98	0.98	0
				V SSIM	0.98	0.98	0
15	296914.00	298063.00	0.39	Y MSE	10.91	10.72	-1.74
				U MSE	6.62	6.92	4.53
				V MSE	6.83	7.11	4.1
				Y PSNR	37.75	37.83	0.21
				U PSNR	39.92	39.74	-0.45
				V PSNR	39.79	39.63	-0.4
				Y SSIM	0.98	0.98	0
				U SSIM	0.96	0.96	0
				V SSIM	0.96	0.96	0
20	234004.00	231593.00	-1.03	Y MSE	21.57	20.93	-2.97
				U MSE	9.95	10.11	1.61
				V MSE	10.65	10.90	2.35
				Y PSNR	34.79	34.92	0.37
				U PSNR	38.15	38.09	-0.16
				V PSNR	37.86	37.76	-0.26
				Y SSIM	0.96	0.96	0
				U SSIM	0.95	0.95	0
				V SSIM	0.95	0.95	0
28	146500.00	145306.00	-0.82	Y MSE	56.99	52.25	-8.32
				U MSE	15.81	15.83	0.13
				V MSE	18.64	18.63	-0.05
				Y PSNR	30.58	30.95	1.21
				U PSNR	36.15	36.15	0
				V PSNR	35.43	35.43	0
				Y SSIM	0.93	0.93	0
				U SSIM	0.93	0.93	0
				V SSIM	0.93	0.94	1.08
35	72849.00	67449.00	-7.41	Y MSE	123.48	111.44	-9.75
				U MSE	23.49	22.66	-3.53
				V MSE	30.99	29.50	-4.81
				Y PSNR	27.22	27.66	1.62
				U PSNR	34.44	34.60	0.46
				V PSNR	33.23	33.45	0.66
				Y SSIM	0.88	0.87	-1.14
				U SSIM	0.91	0.91	0
				V SSIM	0.91	0.92	1.1

Table 6.13 Transcoder results for Mobile CIF sequence, Time saving, cascade transcoder Vs proposed transcoder

QP	Open loop encoding time (s)	Transcoded encoding time (s)	Time saving %
10	64	6	91
15	68	6	91
20	66	5	92
28	71	5	93
36	69	6	91

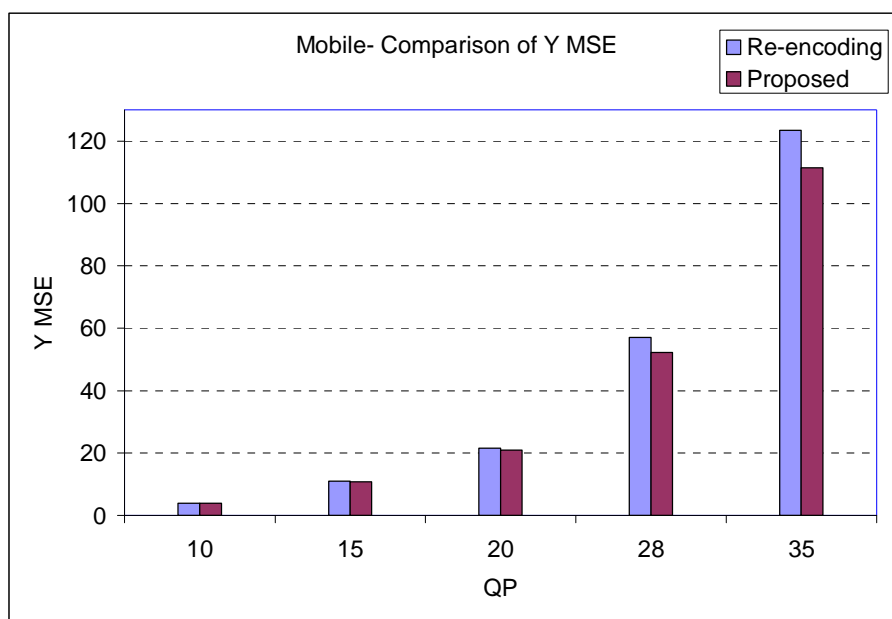


Figure 6.30 Transcoder Results – Mobile, CIF, Comparison of Y mean square error, cascade transcoder Vs proposed transcoder

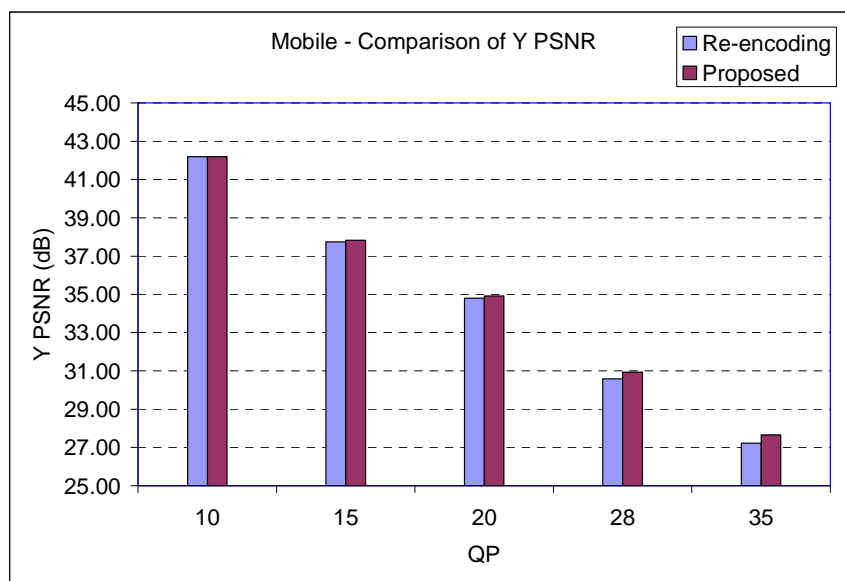


Figure 6.31 Transcoder Results – Mobile, CIF, Comparison of Y peak signal to noise ratio, cascade transcoder Vs proposed transcoder

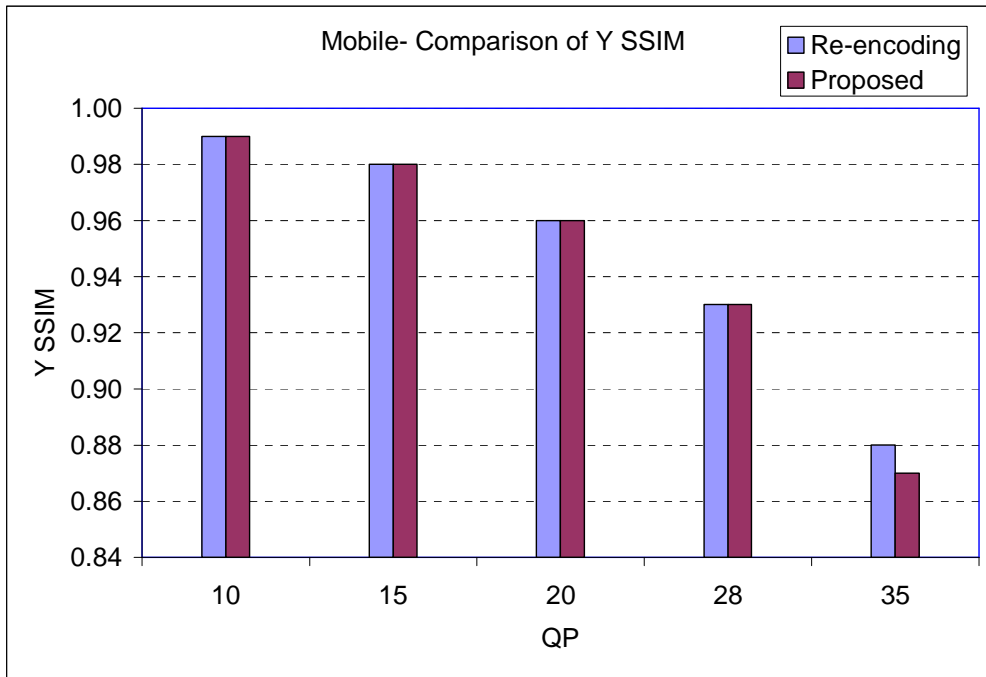


Figure 6.32 Transcoder Results – Mobile, CIF, Comparison of Y structural similarity index, cascade transcoder Vs proposed transcoder

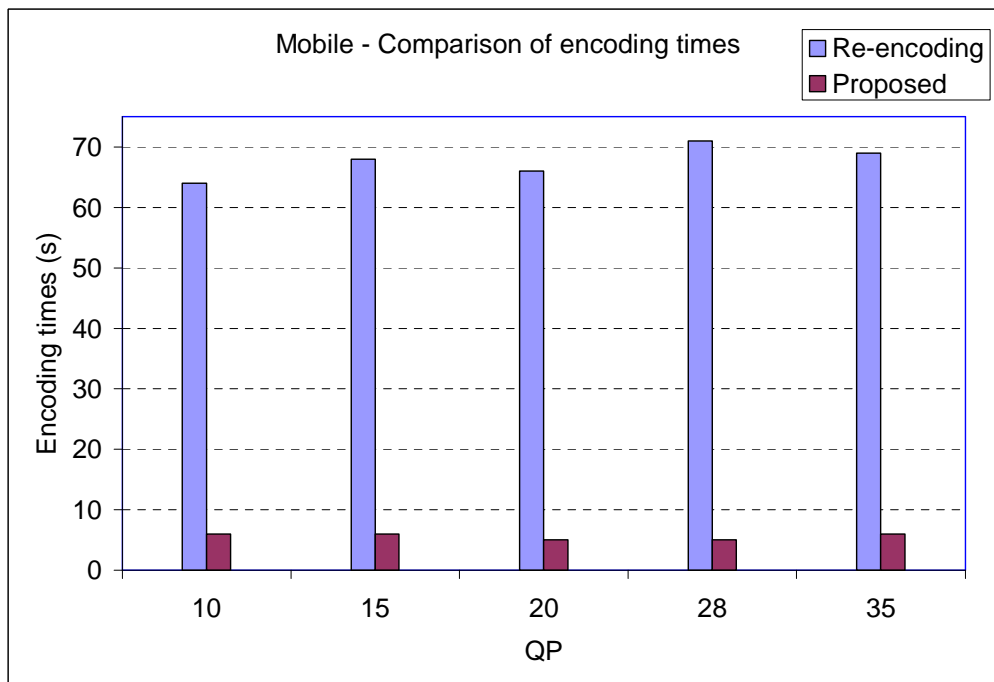


Figure 6.33 Transcoder Results – Mobile, CIF, Comparison of encoding times, cascade transcoder Vs proposed transcoder

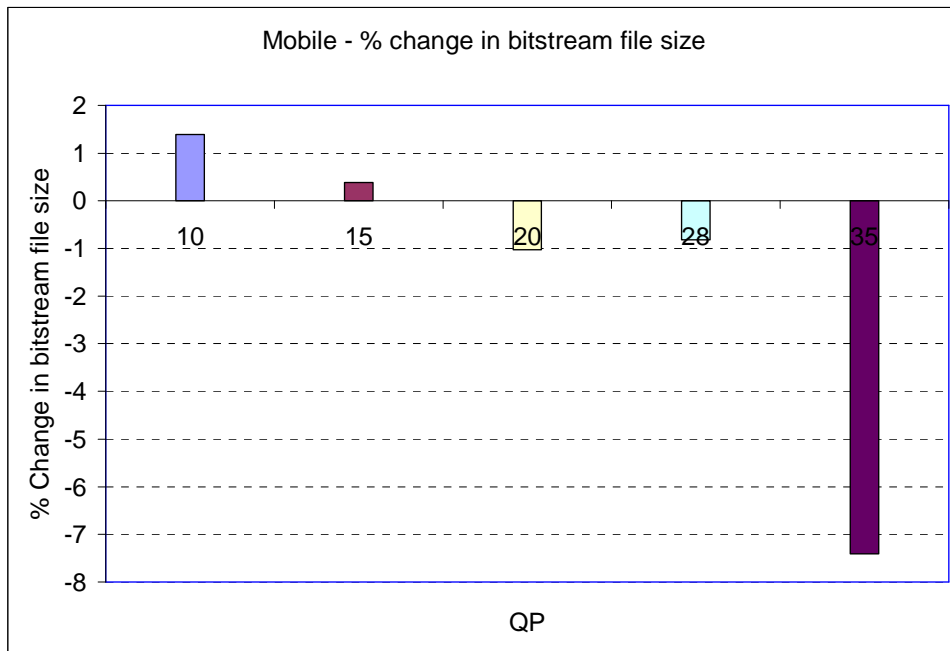


Figure 6.34 Transcoder Results – Mobile, CIF, Percentage change in bitstream file size, cascade transcoder Vs proposed transcoder



Figure 6.35 Football sequence (a) Original (b) H.264 decoded (c) Reference cascade decoded at QP 10 (d) Proposed transcoder decoded at QP 10

6.4.9.2.2 Comparison of proposed transcoder, cascaded transcoder and H.264 decoder with respect to original video

A comparison of the MSE, PSNR, SSIM, bit stream file size and encoding times between the proposed architecture, the cascaded re-encoding transcoder and the H.264 decoder are tabulated for various quantization parameters (QP) and illustrated in Sections 6.4.9.2.2.1 through 6.4.9.2.2.5. The reference video used for all the comparison purpose is the original video. This comparison allows us to understand the performance of VC-1 with respect to H.264 in terms of quality.

6.4.9.2.2.1 Test sequence: Akiyo, QCIF Resolution, 10 frames

Table 6.14 Transcoder results for Akiyo QCIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder

QP	Metrics type	H.264	Open loop	Transcoder
10	Y MSE	0.47	2.63	2.60
	U MSE	0.45	2.07	2.04
	V MSE	0.41	1.65	1.65
	Y PSNR	51.43	43.94	43.98
	U PSNR	51.59	44.97	45.03
	V PSNR	51.97	45.97	45.95
	Y SSIM	1	0.98	0.98
	U SSIM	1	0.98	0.98
	V SSIM	1	0.98	0.98
15	Y MSE	1.06	5.80	5.88
	U MSE	0.82	3.93	3.94
	V MSE	0.74	2.97	2.97
	Y PSNR	47.9	40.49	40.44
	U PSNR	48.98	42.19	42.18
	V PSNR	49.41	43.41	43.41
	Y SSIM	0.99	0.97	0.97
	U SSIM	0.99	0.97	0.97
	V SSIM	0.99	0.97	0.97
20	Y MSE	2.33	10.42	10.55
	U MSE	1.68	7.21	6.95
	V MSE	1.33	4.87	4.87
	Y PSNR	44.46	37.95	37.90
	U PSNR	45.89	39.55	39.71
	V PSNR	46.9	41.25	41.26
	Y SSIM	0.99	0.96	0.96
	U SSIM	0.99	0.96	0.96
	V SSIM	0.99	0.96	0.96
28	Y MSE	8.25	24.54	25.59
	U MSE	5.33	16.35	15.50
	V MSE	4.22	10.36	10.33
	Y PSNR	38.97	34.23	34.05
	U PSNR	40.87	36.00	36.23
	V PSNR	41.88	37.98	37.99
	Y SSIM	0.97	0.93	0.92
	U SSIM	0.97	0.94	0.94
	V SSIM	0.96	0.92	0.92
35	Y MSE	25.25	52.72	54.04
	U MSE	12.09	31.51	29.71
	V MSE	7.77	16.68	17.09
	Y PSNR	34.11	30.91	30.80
	U PSNR	37.31	33.15	33.40
	V PSNR	39.23	35.91	35.80
	Y SSIM	0.93	0.87	0.86
	U SSIM	0.96	0.91	0.91
	V SSIM	0.94	0.90	0.89

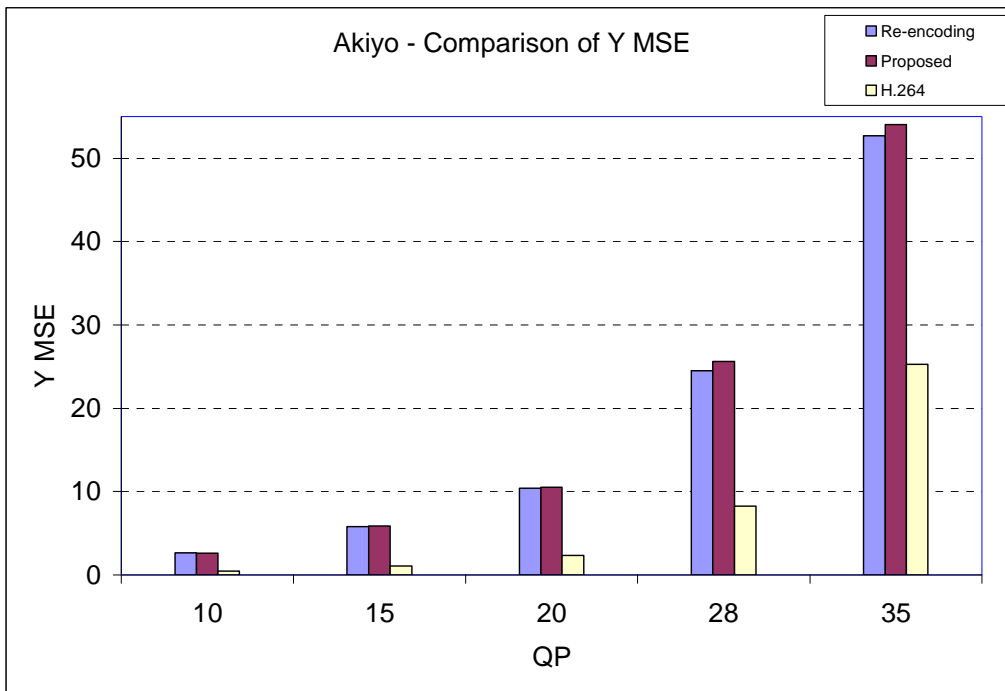


Figure 6.36 Transcoder Results - Akiyo, QCIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder

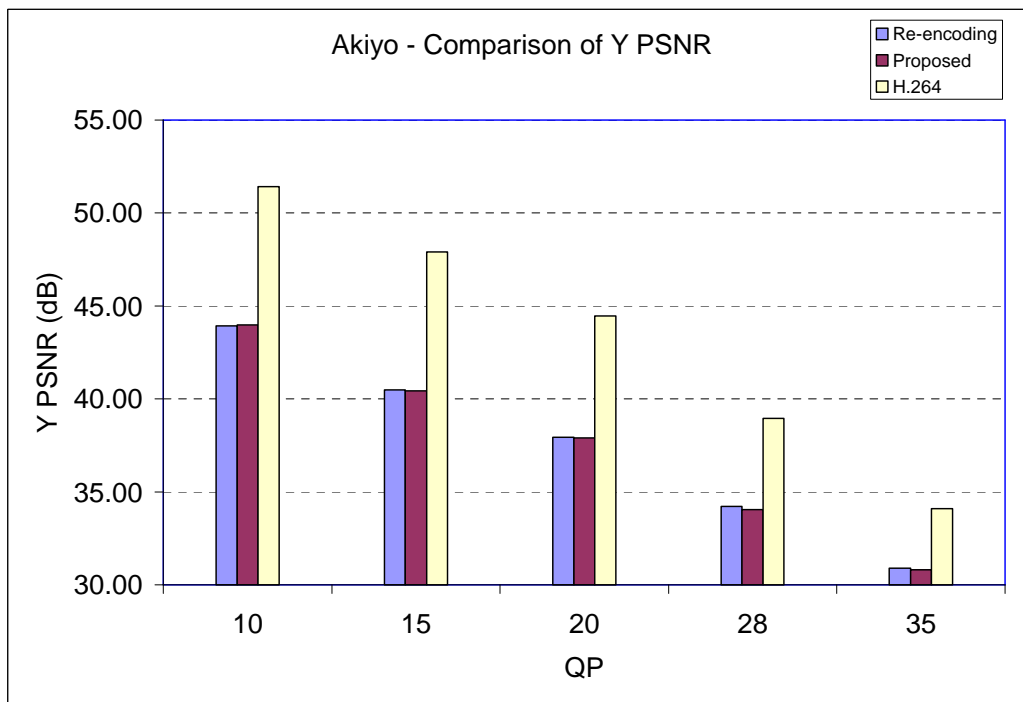


Figure 6.37 Transcoder Results - Akiyo, QCIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder

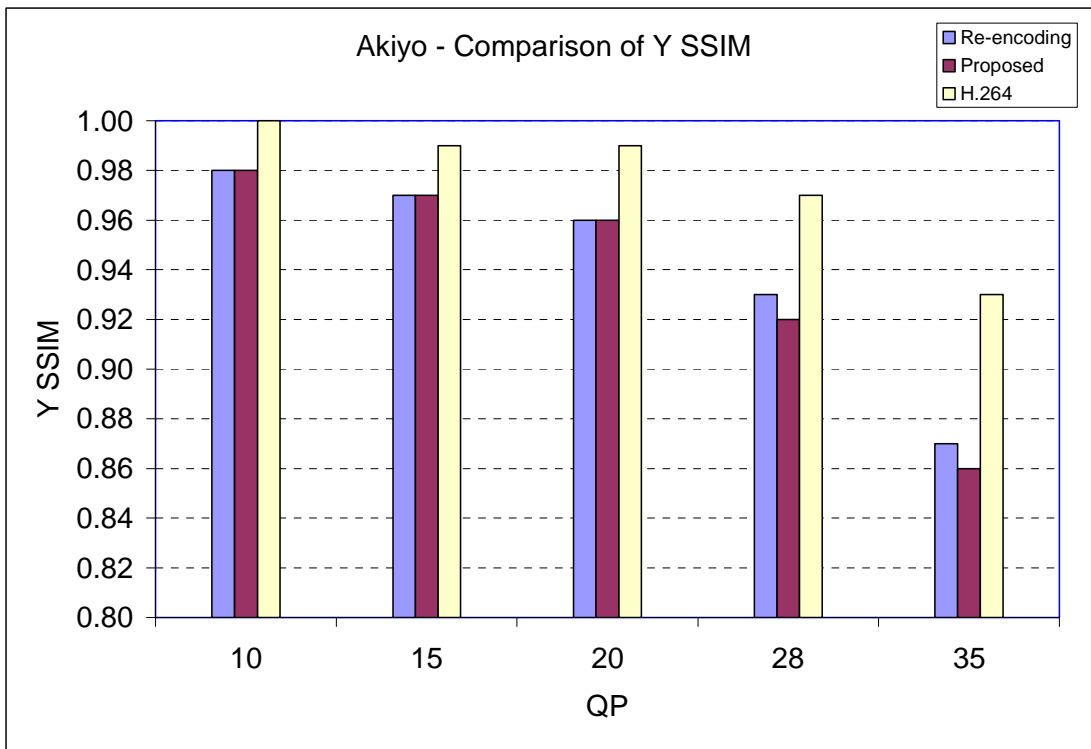


Figure 6.38 Transcoder Results - Akiyo, QCIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder

6.4.9.2.2.2 Test sequence: Miss America, QCIF Resolution, 10 frames

Table 6.15 Transcoder results for Miss America QCIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder

QP	Metrics type	H.264	Open loop	Transcoder
10	Y MSE	0.48	2.31	2.21
	U MSE	0.61	3.32	3.19
	V MSE	0.59	2.66	2.58
	Y PSNR	51.36	44.50	44.69
	U PSNR	50.29	42.93	43.10
	V PSNR	50.41	43.88	44.02
	Y SSIM	0.99	0.98	0.98
	U SSIM	0.99	0.96	0.96
	V SSIM	0.99	0.98	0.98
15	Y MSE	1.03	4.11	3.88
	U MSE	1.6	5.57	5.52
	V MSE	1.35	4.60	4.39
	Y PSNR	48.02	41.99	42.24
	U PSNR	46.11	40.68	40.72
	V PSNR	46.83	41.51	41.71
	Y SSIM	0.99	0.97	0.97
	U SSIM	0.98	0.94	0.94
	V SSIM	0.99	0.96	0.97
20	Y MSE	2	6.86	6.36
	U MSE	3.57	7.77	7.46
	V MSE	2.49	7.59	7.36
	Y PSNR	45.12	39.77	40.09
	U PSNR	42.61	39.23	39.41
	V PSNR	44.17	39.33	39.46
	Y SSIM	0.98	0.96	0.96
	U SSIM	0.96	0.92	0.92
	V SSIM	0.98	0.95	0.95
28	Y MSE	5.83	14.90	13.24
	U MSE	7.53	11.82	11.99
	V MSE	7.1	15.09	15.38
	Y PSNR	40.47	36.40	36.91
	U PSNR	39.36	37.41	37.34
	V PSNR	39.62	36.35	36.27
	Y SSIM	0.97	0.93	0.94
	U SSIM	0.92	0.89	0.89
	V SSIM	0.96	0.93	0.92
35	Y MSE	15.09	29.24	27.25
	U MSE	11.09	14.32	14.60
	V MSE	12.66	26.59	29.24
	Y PSNR	36.35	33.47	33.78
	U PSNR	37.68	36.57	36.49
	V PSNR	37.11	33.89	33.48
	Y SSIM	0.94	0.90	0.90
	U SSIM	0.9	0.88	0.88
	V SSIM	0.94	0.89	0.88

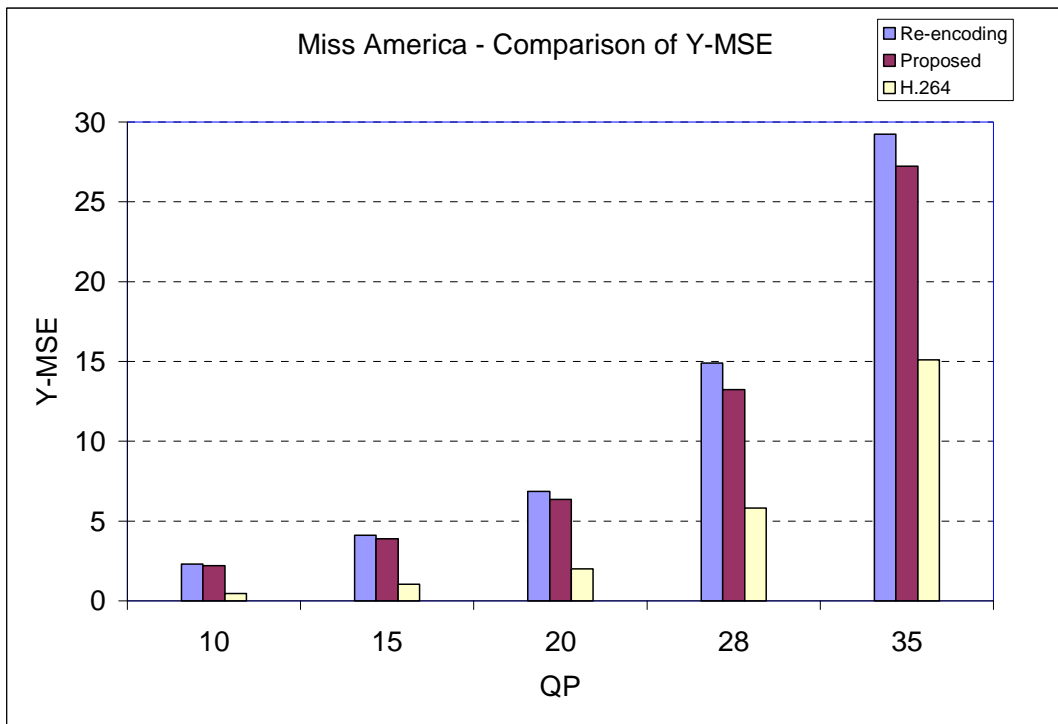


Figure 6.39 Transcoder Results – Miss America, QCIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder

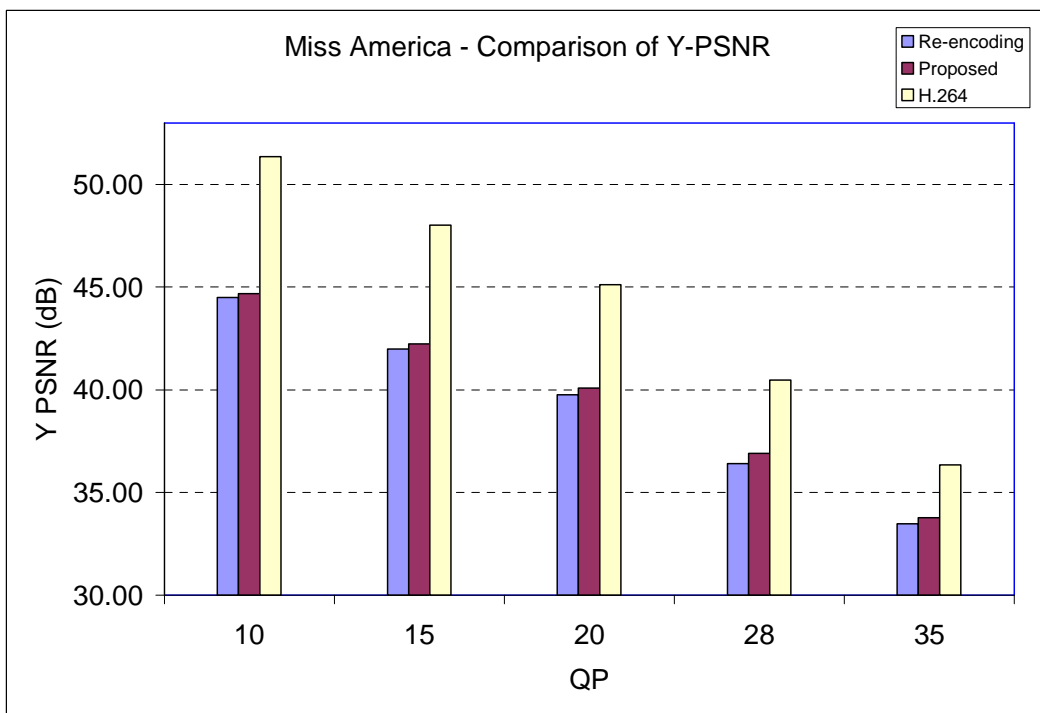


Figure 6.40 Transcoder Results - Miss America, QCIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder

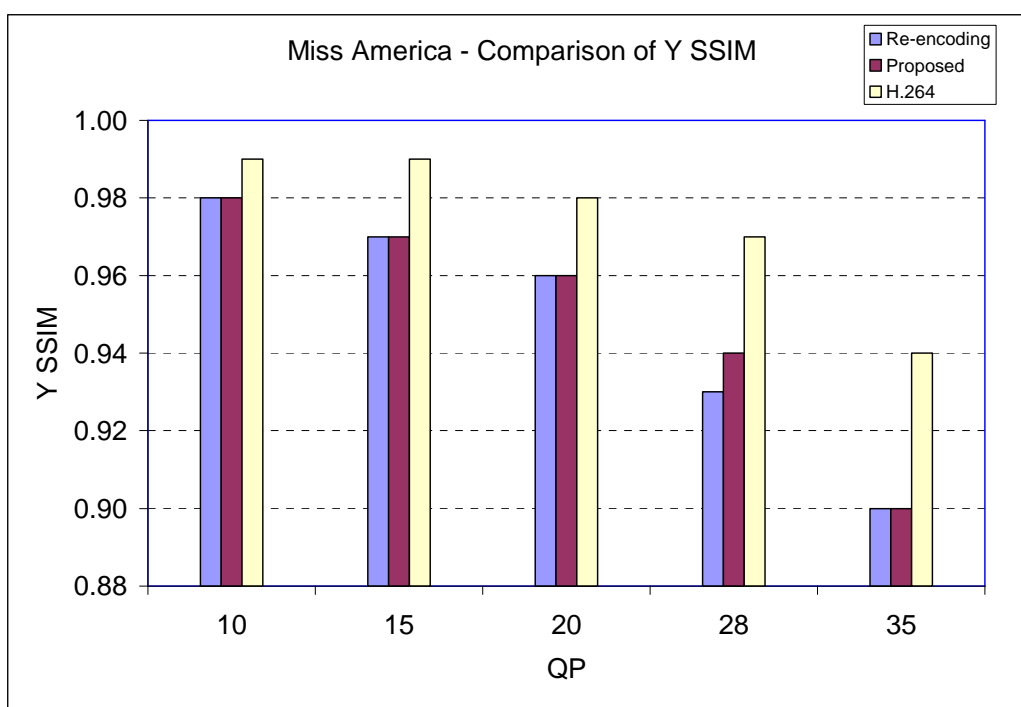


Figure 6.41 Transcoder Results - Miss America, QCIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder

6.4.9.2.2.3 Test sequence: Foreman, CIF Resolution, 10 frames

Table 6.16 Transcoder results for Foreman CIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder

QP	Metrics type	H.264	Open loop	Transcoder
10	Y MSE	0.43	3.98	3.71
	U MSE	0.52	2.16	2.28
	V MSE	0.48	1.40	1.46
	Y PSNR	51.8	42.14	42.45
	U PSNR	50.99	44.79	44.58
	V PSNR	51.29	46.67	46.53
	Y SSIM	1	0.97	0.97
	U SSIM	0.99	0.97	0.97
	V SSIM	0.99	0.98	0.98
15	Y MSE	1.22	8.61	8.03
	U MSE	1.11	3.52	3.56
	V MSE	0.78	2.02	2.01
	Y PSNR	47.27	38.79	39.09
	U PSNR	47.69	42.66	42.61
	V PSNR	49.2	45.09	45.11
	Y SSIM	0.99	0.95	0.95
	U SSIM	0.99	0.96	0.96
	V SSIM	0.99	0.98	0.98
20	Y MSE	3.32	14.50	13.83
	U MSE	2.1	5.10	5.05
	V MSE	1.13	2.81	2.82
	Y PSNR	42.93	36.52	36.73
	U PSNR	44.91	41.06	41.10
	V PSNR	47.61	43.64	43.63
	Y SSIM	0.98	0.92	0.92
	U SSIM	0.98	0.95	0.95
	V SSIM	0.99	0.97	0.98
28	Y MSE	11.61	28.08	30.15
	U MSE	5.09	8.18	7.97
	V MSE	2.58	5.33	5.18
	Y PSNR	37.49	33.65	33.34
	U PSNR	41.06	39.01	39.12
	V PSNR	44.02	40.87	41.00
	Y SSIM	0.94	0.89	0.87
	U SSIM	0.95	0.93	0.93
	V SSIM	0.98	0.97	0.97
35	Y MSE	29.24	51.68	60.94
	U MSE	7.53	11.18	10.87
	V MSE	4.72	9.74	8.76
	Y PSNR	33.47	31.00	30.28
	U PSNR	39.36	37.65	37.77
	V PSNR	41.39	38.25	38.71
	Y SSIM	0.89	0.85	0.82
	U SSIM	0.94	0.92	0.93
	V SSIM	0.97	0.95	0.96

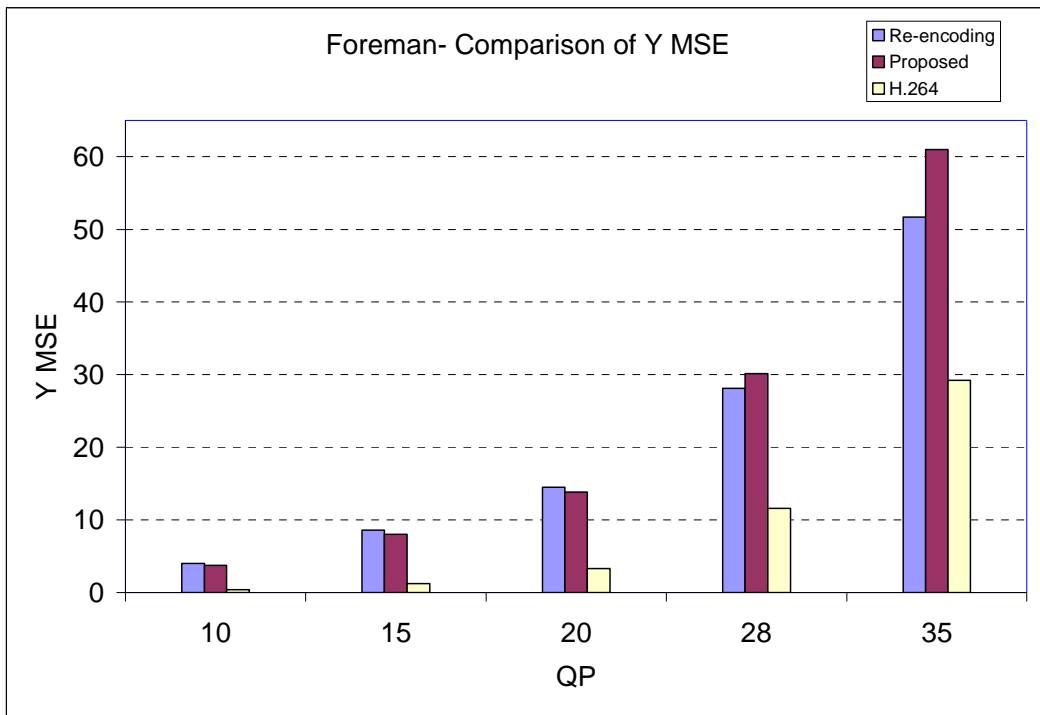


Figure 6.42 Transcoder Results – Foreman, CIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder

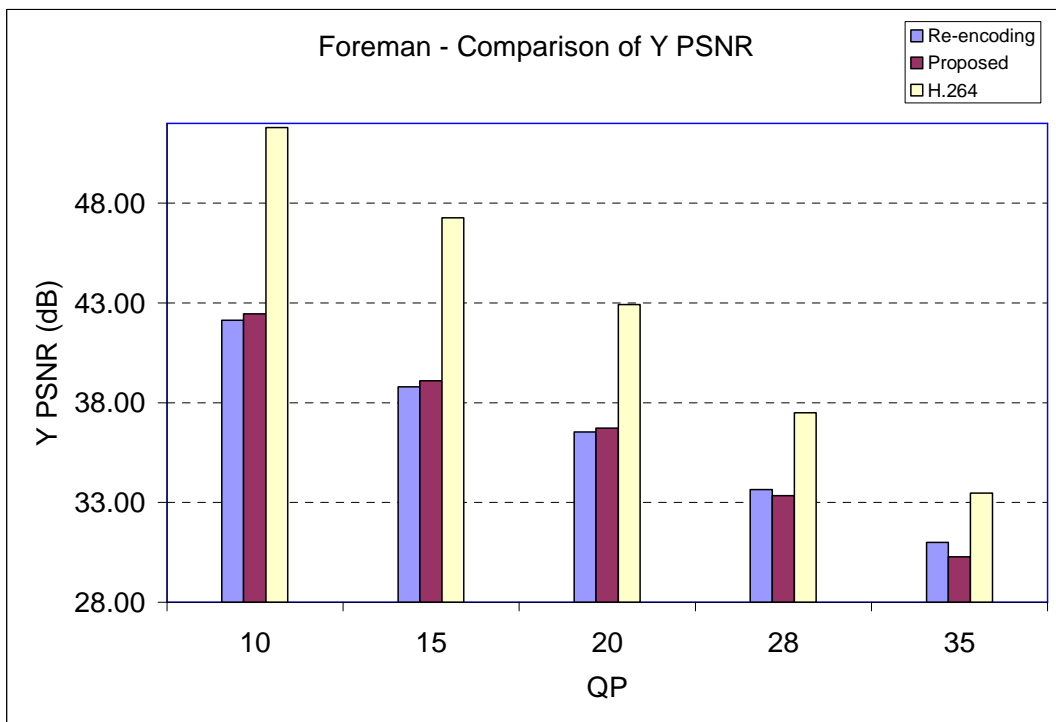


Figure 6.43 Transcoder Results - Foreman, CIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder

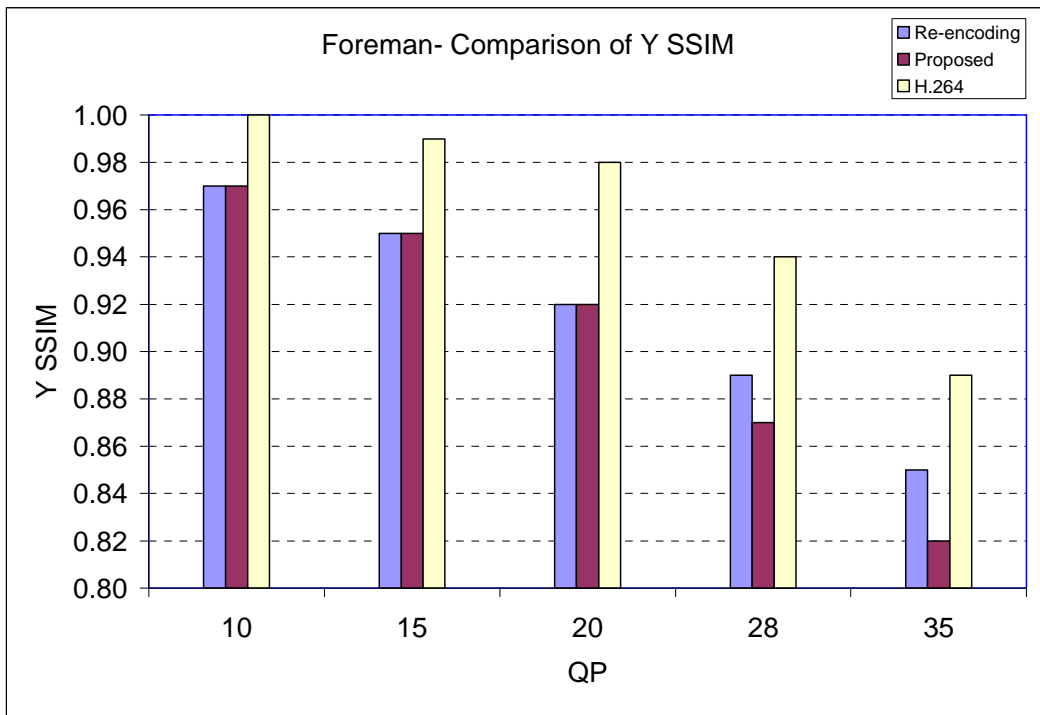


Figure 6.44 Transcoder Results - Foreman, CIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder

6.4.9.2.2.4 Test sequence: Football, CIF Resolution, 10 frames

Table 6.17 Transcoder results for Football CIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder

QP	Metrics type	H.264	Open loop	Transcoder
10	Y MSE	0.42	4.30	4.11
	U MSE	0.5	2.57	2.80
	V MSE	0.51	2.37	2.58
	Y PSNR	51.88	41.80	41.99
	U PSNR	51.16	44.04	43.69
	V PSNR	51.11	44.39	44.04
	Y SSIM	1	0.98	0.99
	U SSIM	0.99	0.98	0.97
	V SSIM	0.99	0.97	0.97
15	Y MSE	1.22	11.24	10.22
	U MSE	1.16	4.92	5.04
	V MSE	1.12	4.09	4.17
	Y PSNR	47.29	37.63	38.04
	U PSNR	47.5	41.21	41.11
	V PSNR	47.63	42.01	41.94
	Y SSIM	1	0.96	0.97
	U SSIM	0.99	0.96	0.96
	V SSIM	0.99	0.96	0.96
20	Y MSE	3.6	22.18	19.70
	U MSE	2.42	8.22	8.22
	V MSE	2.23	6.31	6.29
	Y PSNR	42.58	34.68	35.19
	U PSNR	44.3	38.99	38.99
	V PSNR	44.66	40.14	40.15
	Y SSIM	0.99	0.93	0.94
	U SSIM	0.98	0.94	0.94
	V SSIM	0.98	0.94	0.94
28	Y MSE	17.59	56.91	51.85
	U MSE	7.66	17.98	18.00
	V MSE	5.83	11.94	11.77
	Y PSNR	35.68	30.58	30.98
	U PSNR	39.3	35.59	35.58
	V PSNR	40.48	37.36	37.43
	Y SSIM	0.94	0.84	0.84
	U SSIM	0.95	0.89	0.89
	V SSIM	0.95	0.91	0.91
35	Y MSE	60.59	124.26	121.16
	U MSE	14.75	33.58	32.02
	V MSE	9.79	18.84	18.13
	Y PSNR	30.31	27.19	27.30
	U PSNR	36.45	32.88	33.09
	V PSNR	38.23	35.38	35.55
	Y SSIM	0.82	0.72	0.71
	U SSIM	0.92	0.84	0.85
	V SSIM	0.93	0.88	0.88

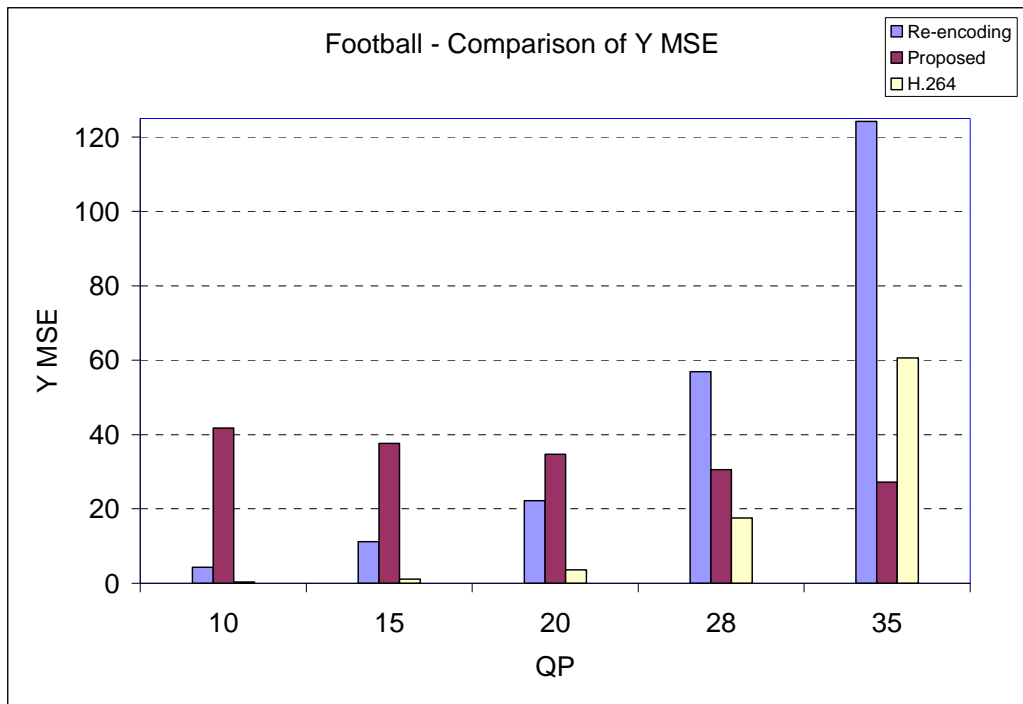


Figure 6.45 Transcoder Results – Football, CIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder

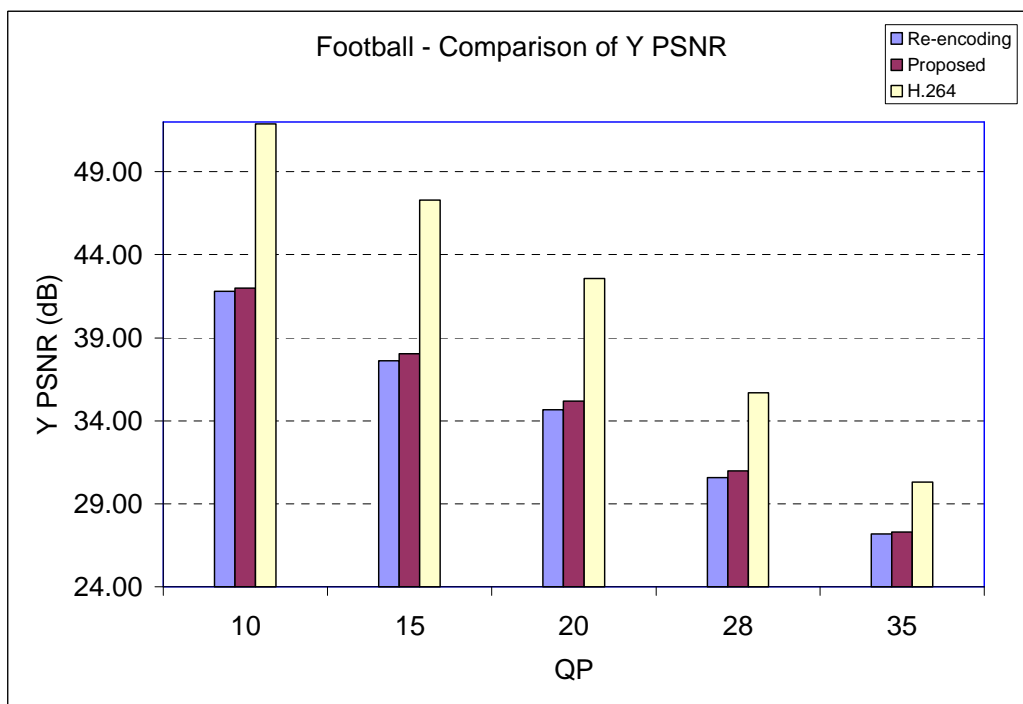


Figure 6.46 Transcoder Results - Football, CIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder

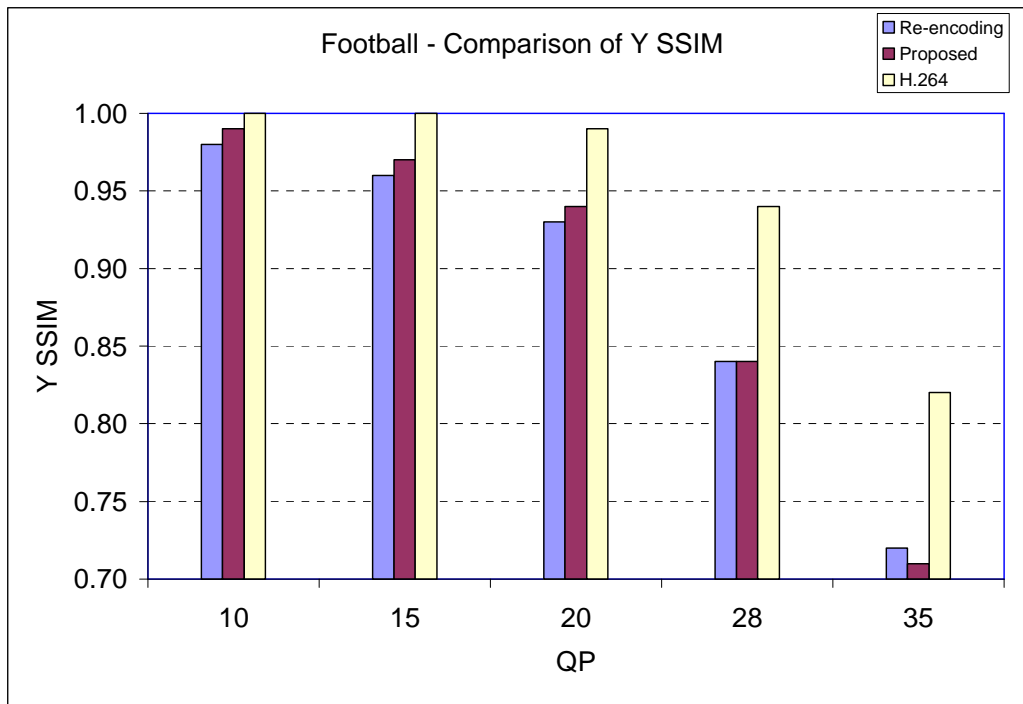


Figure 6.47 Transcoder Results - Football, CIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder

6.4.9.2.2.5 Test sequence: Mobile, CIF Resolution, 10 frames

Table 6.18 Transcoder results for Mobile CIF sequence, H.264 Vs cascade transcoder Vs proposed transcoder

QP	Metrics type	H.264	Open loop	Transcoder
10	Y MSE	0.41	4.32	4.31
	U MSE	0.48	3.53	4.14
	V MSE	0.47	3.51	4.08
	Y PSNR	52.04	41.77	41.79
	U PSNR	51.3	42.65	41.99
	V PSNR	51.39	42.68	42.06
	Y SSIM	1	0.99	0.99
	U SSIM	1	0.97	0.97
	V SSIM	1	0.98	0.98
15	Y MSE	1.13	11.84	11.62
	U MSE	1.34	7.88	8.18
	V MSE	1.31	8.05	8.31
	Y PSNR	47.63	37.40	37.48
	U PSNR	46.87	39.17	39.01
	V PSNR	46.98	39.08	38.94
	Y SSIM	1	0.97	0.97
	U SSIM	0.99	0.95	0.95
	V SSIM	0.99	0.96	0.96
20	Y MSE	3.32	24.42	23.75
	U MSE	3.71	13.77	13.98
	V MSE	3.59	14.37	14.52
	Y PSNR	42.94	34.26	34.37
	U PSNR	42.45	36.74	36.68
	V PSNR	42.58	36.56	36.51
	Y SSIM	0.99	0.96	0.95
	U SSIM	0.98	0.92	0.93
	V SSIM	0.98	0.93	0.93
28	Y MSE	19.15	73.16	69.32
	U MSE	13.88	32.31	32.17
	V MSE	14.43	35.46	35.33
	Y PSNR	35.32	29.50	29.73
	U PSNR	36.72	33.04	33.06
	V PSNR	36.55	32.64	32.65
	Y SSIM	0.97	0.91	0.91
	U SSIM	0.92	0.86	0.86
	V SSIM	0.93	0.87	0.87
35	Y MSE	83.28	202.69	194.80
	U MSE	26.96	55.12	53.70
	V MSE	29.54	65.49	63.83
	Y PSNR	28.93	25.07	25.24
	U PSNR	33.83	30.72	30.83
	V PSNR	33.44	29.97	30.09
	Y SSIM	0.92	0.82	0.81
	U SSIM	0.88	0.80	0.80
	V SSIM	0.89	0.81	0.81

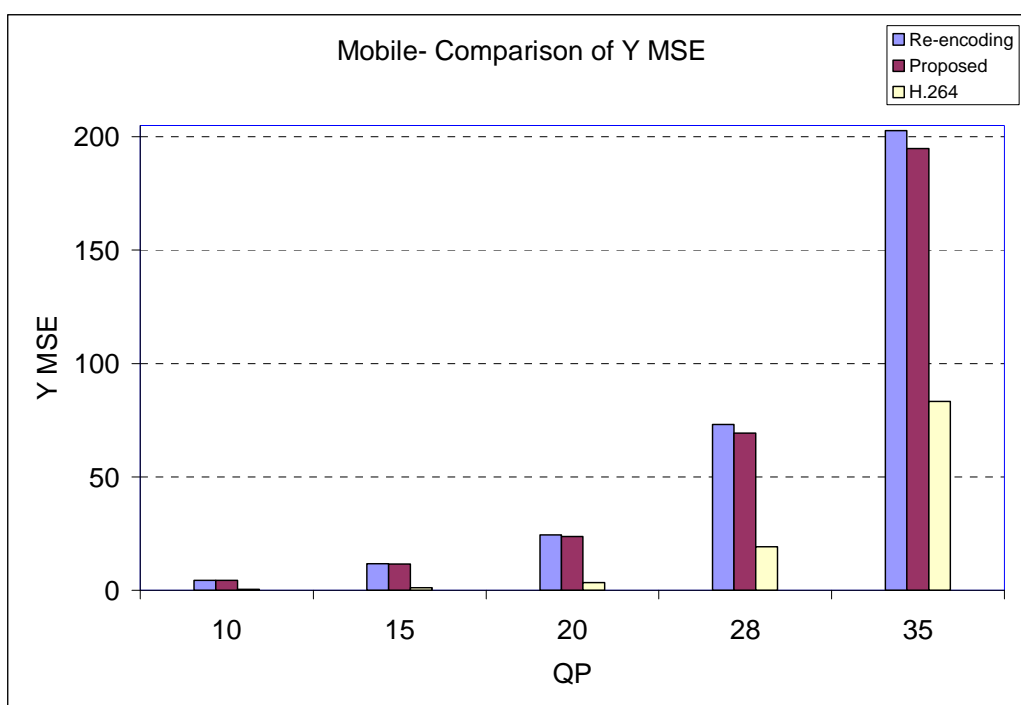


Figure 6.48 Transcoder Results – Mobile, CIF, Comparison of Y mean square error, H.264 Vs cascade transcoder Vs proposed transcoder

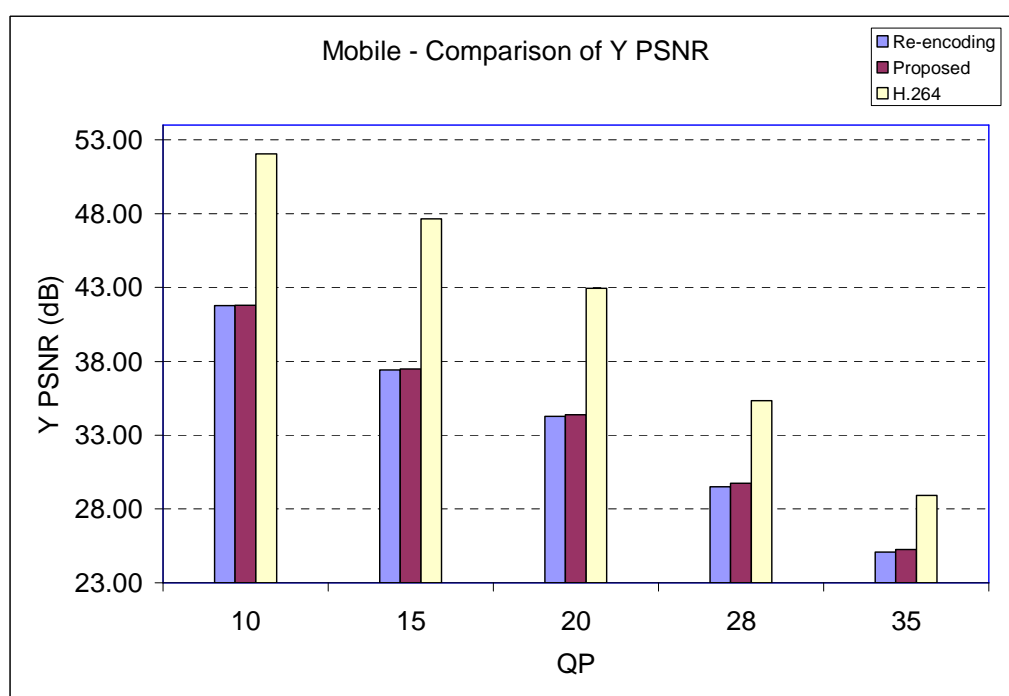


Figure 6.49 Transcoder Results - Mobile, CIF, Comparison of Y peak signal to noise ratio, H.264 Vs cascade transcoder Vs proposed transcoder

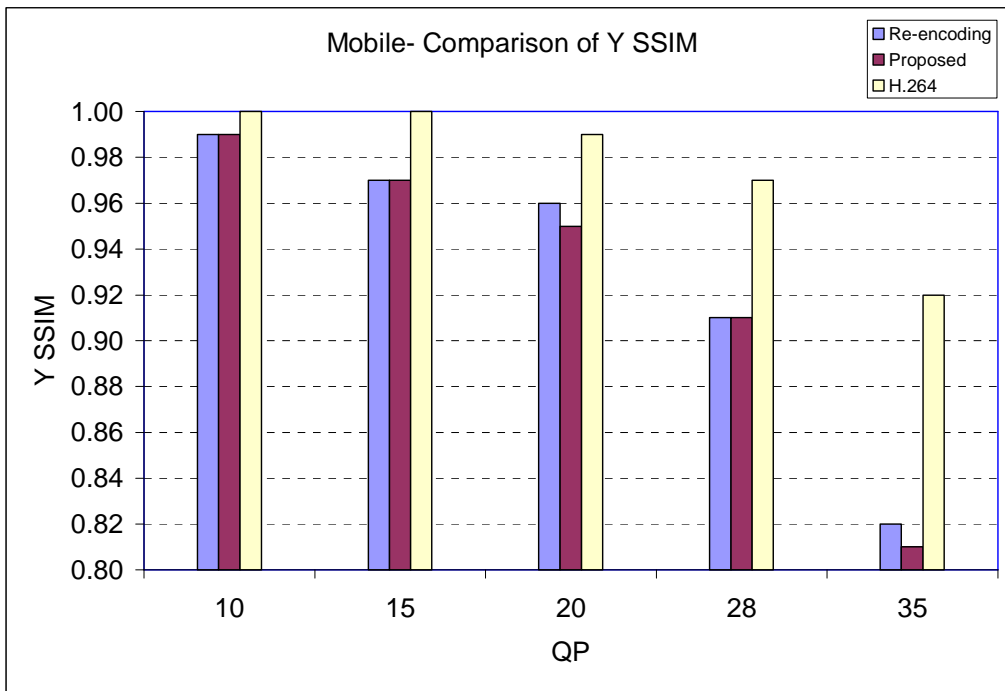


Figure 6.50 Transcoder Results - Mobile, CIF, Comparison of Y structural similarity index, H.264 Vs cascade transcoder Vs proposed transcoder

6.4.9.3 Observations

From the observed results in Section 6.4.9.2.1, it can be inferred that the proposed transcoder performs comparable to the cascaded transcoder in terms of subjective quality. In other words, the proposed low complexity transcoder is a good solution in real time transcoding applications.

Though the cascaded and proposed transcoder do not match up to the objective quality of H.264 compared to the original video (refer Section 6.4.9.2.2), in terms of subjective quality, the transcoder is comparable to H.264's performance at low quantization parameters. This observation strengthens the earlier claim that the low complexity VC-1 codec is comparable to H.264 codec in subjective quality, which is the primary motivation for this research. The transcoder, however, falls short of the H.264 performance at high quantization parameters.

6.5 Conclusions and future work

6.5.1 Conclusions

It can be observed that the mode decision and motion vector information from the incoming H.264 bitstream can be used in re-encoding the H.264 bitstream to VC-1 bitstream. The resulting loss of quality as a result, in comparison with the cascaded decoder and encoder model is very less. The quality reduction is mainly due to simplified motion estimation process of the proposed VC-1 encoder. However corresponding reduction in the encoding time is high. Hence the complexity in the re-encoding process is reduced significantly using the proposed technique. The motion estimation block is completely eliminated in the encoding process. This means significant reduction in hardware chip size, cost and less power consumption in power valuable mobile devices.

6.5.2 Future work

The current technique only proposes the re-use of motion vectors / median of motion vectors available from the H.264 bitstream. It does not involve any motion vector (MV) refinement. The MV refinement process is useful in getting more accurate motion vector values from the approximate motion vector values by making a simplified search in small windows around the approximate values. The authors in [61] and [63] describe different motion vector refinement techniques. The transcoded video quality can be further improved by using MV refinement. Finding an appropriate MV refinement technique to supplement the proposed MV reuse can form the basis for future research.

The current transcoder deals with the baseline profile of H.264 and simple profile of VC-1. The idea can be extended to other profiles like main and high profiles. Since main and high profiles of H.264 involve bi-directional prediction with multiple reference pictures and VC-1 allows only two reference pictures, algorithms to re-scale the motion vectors according to the appropriate reference pictures can be explored.

REFERENCES

- [1] I. E.G. Richardson, "H.264 and MPEG-4 video compression: video coding for next-generation multimedia", Wiley, 2003.
- [2] E. Lallana and M. Uy, "The Information Age," UNDP-APDIP, 2003.
- [3] Open source article, "Digital Revolution," Wikipedia Foundation, http://en.wikipedia.org/wiki/Digital_Revolution
- [4] J. Ribas-Corbera, "Windows Media 9 Series — a platform to deliver compressed audio and video for Internet and broadcast applications", EBU Technical Review No. 293, Jan. 2003
- [5] K. Sayood, "Introduction to Data Compression," 3rd Edition, Morgan Kaufmann Publisher Inc., 2006.
- [6] Open source article, "Rec 601," Wikipedia Foundation, http://en.wikipedia.org/wiki/Rec._601
- [7] R. Schafer and T. Sikora, "Digital video coding standards and their role in video communications," Proceedings of the IEEE, Vol. 83, pp. 907-923, Jan. 1995.
- [8] Information technology-generic coding of moving pictures and associated audio information: ISO/IEC 13818-2 (MPEG-2) Std.
- [9] Advanced Video Coding for Generic Audiovisual Services, ITU-T Rec. H.264 / ISO / IEC 14496-10, Nov. 2009.
- [10] VC-1 Compressed Video Bitstream Format and Decoding Process (SMPTE 421M-2006), SMPTE Standard, 2006 (<http://store.smppte.org/category-s/1.htm>).
- [11] A. Vetro, C. Christopoulos and H. Sun, "Video transcoding architectures and techniques: an overview," IEEE Signal Processing Magazine, pp 18-29, March 2003.
- [12] I. Ahmad et al, "Video transcoding: An overview of various techniques and

- research issues”, IEEE Trans. on Multimedia, vol. 7, pp. 793-804, Oct. 2005
- [13] C. Chen, P-H.Wu and H. Chen, “MPEG-2 to H.264 transcoding,” Picture Coding Symposium, pp. 15-17, Dec. 2004.
- [14] Open source article, “H.264/MPEG-4 AVC,” Wikipedia Foundation, http://en.wikipedia.org/wiki/H.264/MPEG-4_AVC
- [15] G.A Davidson et al, “ATSC video and audio coding”, Proc. IEEE, vol 94, pp. 60-76, Jan. 2006 (www.atsc.org).
- [16] G.F.-Escribano et al, “ An MPEG-2 to H.264 video transcoder in the baseline profile”, IEEE Trans. CSVT, vol. 20, pp. 763-768, May 2010
- [17] Jae-Beom Lee and H. Kalva, "An efficient algorithm for VC-1 to H.264 video transcoding in progressive compression," IEEE International Conference on Multimedia and Expo, pp. 53-56, July 2006.
- [18] Open Source Article, “VC-1 Technical Overview”, Microsoft Corporation, <http://www.microsoft.com/windows/windowsmedia/howto/articles/vc1techoverview.aspx>
- [19] DVD Forum, <http://www.dvdforum.org/forum.shtml>
- [20] DV Magazine, <http://www.dv.com/>
- [21] Tandberg Television, <http://www.tandberg.com/index2.jsp>
- [22] CT Magazine, <http://www.heise.de/ct/>
- [23] European Broadcasting Union, <http://www.ebu.ch/>
- [24] Blue-ray Disc Association - <http://www.blu-raydisc.com/index.htm>
- [25] Real Media, <http://www.real.com/>
- [26] Envivo MPEG-4, <http://www.envivio.com/>
- [27] Apple MPEG-4, <http://www.apple.com/quicktime/technologies/mpeg4/>
- [28] M. Jeffrey, “The SMPTE 421M "VC-1" Standardization Project”, ITU-T Workshop Video and Image Coding and Applications, July 2005
- [29] R. Pereira, K.R. Rao and A. Kruafak, “Efficient transcoding of an MPEG bitstream to an H.264 bit stream”, University Scientific Journal series Telecommunications and Electronics, vol.11, pp. 5-31, Poland, 2008

- [30] S. Sharma and K.R. Rao, "Transcoding of H.264 bitstream to MPEG-2 bitstream", IEEE APCC, pp. 391-396 Bangkok, Thailand, 18-20 Oct. 2007.
- [31] S. Moiron et al, "Video transcoding from H.264/AVC to MPEG-2 with reduced computational complexity", Signal Processing: Image Communication, vol 24, pp. 637-650, Sept. 2009
- [32] S. Kwon, A. Tamhankar and K.R. Rao, "Overview of H.264 / MPEG-4 Part 10", J. Visual Communication and Image Representation, vol. 17, pp.183-216, April 2006.
- [33] T. Wiegand and G. J. Sullivan, "The H.264 video coding standard", IEEE Signal Processing Magazine, vol. 24, pp. 148-153, March 2007.
- [34] A. Puri et al, "Video Coding using the H.264/ MPEG-4 AVC compression standard", Signal Processing: Image Communication, vol. 19, pp: 793 – 849, Oct. 2004.
- [35] D. Marpe, T. Wiegand and S. Gordon, "H.264/MPEG4-AVC Fidelity Range Extensions: Tools, Profiles, Performance, and Application Areas", Proc. IEEE International Conference on Image Processing 2005, vol. 1, pp. 593 - 596, Sept. 2005.
- [36] G. Sullivan, P. Topiwala and A. Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions", SPIE conference on Applications of Digital Image Processing XXVII, vol. 5558, pp. 53-74, Aug. 2004.
- [37] K. R. Rao and P. C. Yip, "The transform and data compression handbook", Boca Raton,FL: CRC press, 2001.
- [38] T. Wiegand et al, "Introduction to the Special Issue on Scalable Video Coding—Standardization and Beyond" IEEE Trans on Circuits and Systems for Video Technology, vol. 17, pp. 1034, Sept. 2007.
- [39] HHI presentation of the Scalable Extension of H.264/AVC,
http://ip.hhi.de/imagecom_G1/savce/index.htm
- [40] S. Srinivasan and S. L. Regunathan, "An overview of VC-1," Visual

- Communications and Image Processing (VCIP), *Proc. SPIE*, vol. 5960, pp. 720-728, July 2005.
- [41] Open Source Article, "VC-1", Wikipedia Foundation,
<http://en.wikipedia.org/wiki/VC-1>
- [42] S. Srinivasan et al, "Windows Media Video 9: overview and applications",
Signal Processing: Image Communication, vol. 19, Issue 9, pp. 851-875,
Oct. 2004
- [43] Y. Huh, K. Panusopone, K.R. Rao, "Variable block size coding of images
with hybrid quantization", *IEEE Trans. Circuits Systems Video Technol.* 6,
pp 679–685, Dec. 1996
- [44] J. Ribas-Corbera and D.L. Neuhoff, "Optimizing Block Size in Motion
Compensation", *Journal of Electronic Imaging*, vol. 7, pp.155-165, Jan.
1998
- [45] T.D.Tran, J.Liang and C. Tu, "Lapped Transform via Time-Domain Pre- and
Post-Filtering", *IEEE Trans on Signal Processing*, vol.51, no.6, pp. 1557-
1571, Jun. 2003.
- [46] W.B.Pennebaker and J.L.Mitchell, *JPEG Still Image Data Compression
Standard*, Van Nostrand Reinhold, 1993.
- [47] M. Wien, "Variable block size transforms for H.264/AVC", *IEEE Trans.
Circuits Systems Video Technology*, vol. 13, pp. 604–613, July 2003.
- [48] S. Gordon, "Adaptive Block Transform for Film Grain Reproduction in High
Definition Sequences", ", Joint Video Team (JVT) of ISO/IEC MPEG & ITU-
T VCEG, doc. JVT-H029, Geneva, Switzerland, 23-27 May, 2003 (available
via anonymous ftp from <ftp://ftp.imtc-files.org/jvt-experts/>)
- [49] Tektronics Picture Quality Analyzer PQA 300
(<http://www.tek.com/site/ps/0,,25-11735-INTRO EN,00.html>).
- [50] Microsoft, WMV-9—an Advanced Video Codec for 3GPP, 3GPP SA4
Meeting #18, document S4-030613. (available from
http://www.3gpp.org/ftp/tsg_sa/WG4_CODEC/TSGS4_28/Docs/)
- [51] Nokia, Proposal to support MPEG-4 AVC / H.264/ AVC in Rel-6, 3GPP SA4

- Meeting #27, document S4-030478. (available from
http://www.3gpp.org/ftp/tsg_sa/WG4_CODEC/TSGS4_27/Docs/)
- [52] Z. Volta, *Kompressionisten*, C'T Mag. 10 (2003) 146-159, (in German, summary at
<http://www.heise.de/ct/03/10/146/>).
- [53] Windows Media Web site for Consumer Electronic devices,
<http://www.microsoft.com/windows/windowsmedia/conselec.asp>.
- [54] M. Ravassi, M. Mattavelli and C. Clerc, "JVT/H.26L decoder complexity analysis",
Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, doc. JVT-D153,
Klagenfurt, Austria, 22–26 July, 2002 (available via anonymous ftp from
<ftp://ftp.imtc-files.org/jvt-experts/>).
- [55] H. Sun, X. Chen and T. Chiang, "Digital video transcoding for transmission and
storage," CRC Press, 2005.
- [56] J. Xin, C. Lin and M. Sun, "Digital Video Transcoding", *Proceedings of the IEEE*, Vol
93, pp 84-96, Jan. 2005
- [57] B. Girod, "Overview: Video coding standards," Stanford University coursework,
<http://www.stanford.edu/class/ee398b/handouts.htm>
- [58] G. Sullivan, "Overview of international video coding standards (preceding
H.264/AVC),"ITU-T VICA workshop, Geneva, July 2005.
- [59] M.-T. Sun, T.-D. Wu and J.-N. Hwang, "Dynamic bit allocation in video combining
for multipoint conferencing," *IEEE Trans. Circuit Syst. II*, vol. 45, no. 5, pp. 644-648,
May 1998.
- [60] O. Werner, "Re-quantization for transcoding of MPEG-2 intra frames," *IEEE Trans.
Image Processing*, vol. 8, no. 2, pp. 179-191, Feb. 1999.
- [61] T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to low spatial
temporal resolutions and different encoding formats," *IEEE Trans. Multimedia*, vol. 2,
no. 2, pp. 101-110, Jun. 2000.
- [62] K.-H. Tan and M. Ghanbari, "Layered image coding using the DCT pyramid," *IEEE
Trans. Image Processing*, vol. 4, no. 4, pp. 512-516, Apr. 1995.

- [63] J.-N.Hwang and T.-D. Wu, "Motion vector re-estimation and dynamic frame-skipping for video transcoding," Conf Rec. 32nd Asilomar Conf. Signals, Systems and Computer vol 2, pp 1606-1610, 1998.
- [64] J. Youn, M.-T. Sun and C.-W. Lin, "Motion vector refinement for high-performance transcoding," IEEE Trans. Multimedia, vol. 1, no. 1, pp. 30-40, Mar 1999.
- [65] M.-J. Chen, M.-C. Chu and C.-W. Pan, "Efficient motion estimation algorithm for reduced frame-rate video transcoder," IEEE Trans. Circuits Syst. Video Technol., vol. 12, pp. 269-275, Apr 2002.
- [66] N. Bjork and C. Christopoulos, "Transcoder architecture for video coding," IEEE Trans. Consumer Electronics., vol. 44, pp. 88–98, Feb. 1998.
- [67] R. Mohan, J. R. Smith, and C. Li, "Adapting multimedia internet content for universal 41access," IEEE Trans. Multimedia, vol. 1, no. 1, pp. 104–114, Mar. 1999
- [68] J.-N. Hwang and T.-D. Wu, "Motion vector re-estimation and dynamic frame-skipping for video transcoding," in Conf. Rec. 32nd Asilomar Conf. Signals, System & Computer, vol. 2, 1998, pp. 1606–1610.
- [69] I. E. Richardson, "The H.264 Advanced Video Compression Standard", Second Edition, Wiley, May 2010.
- [70] JM reference software <http://iphome.hhi.de/suehring/tml/>
- [71] VC-1 SMPTE software <http://store.smppte.org/category-s/30.htm>
- [72] A. Luthra, G. Sullivan and T. Wiegand, "Introduction to the special issue on the H.264/AVC video coding standard", IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, issue 7, pp. 557-559, July 2003.
- [73] J. Padia, "Complexity reduction for VP6 to H.264 transcoder using motion vector re-use", M.S. Thesis, EE Dept, UT Arlington, May 2010.
- [74] Open source article, "MPEG-2," Wikipedia Foundation, <http://en.wikipedia.org/wiki/MPEG-2>
- [75] Open source article, "MPEG-1," Wikipedia Foundation, <http://en.wikipedia.org/wiki/MPEG-1>

- [76] Open source article, "H.261," Wikipedia Foundation,
<http://en.wikipedia.org/wiki/H.261>
- [77] Open source article, "H.263," Wikipedia Foundation,
<http://en.wikipedia.org/wiki/H.263>
- [78] Open source article, "MPEG-4 - Part 2" Wikipedia Foundation,
http://en.wikipedia.org/wiki/MPEG-4_Part_2
- [79] H. Kalva and J.B. Lee, "The VC-1 and H.264 Video Compression Standards for Broadband Video Services", Springer, 2008.
- [80] H. Kalva and J.B. Lee, "The VC-1 Video Coding Standard," IEEE Multimedia, Vol. 14, No 4, pp. 88-91, Oct.-Dec. 2007.
- [81] G. Sullivan, "Recent developments towards standardization of next generation video codec", SPIE Photonics and Optics, SPIE, vol. 7798, paper 30, San Deigo, CA, Aug. 2010.

BIOGRAPHICAL INFORMATION

Vidhya Vijayakumar hails from Chennai (Madras), capital city of the southern state of Tamil Nadu in India. She received her bachelor's degree in Electronics and Communication from Sathyabama Deemed University, Chennai in May 2005. After her bachelors, she worked in the Digital Signal Processing and Multimedia group at Wipro technologies. To pursue her interest in academia, she later joined the graduate program at University of Texas at Arlington (UT Arlington). During her stay at UT Arlington, she was a member of the Multimedia Processing Lab, headed by Dr. K.R. Rao. She did her internship with Adobe Inc working in their Advanced Technology Lab. She worked as a Graduate Researcher for the Multimedia laboratory, headed by Dr. Ishfaq Ahmad. She was also awarded the Dean's Masters Scholarship at UT Arlington for the year 2008-10 for academic excellence. Her research interests are in the field of video compression, transcoding and wireless communications and she plans to join the video development group at Polycom after graduation.