Advanced video coding based on matrix decomposition

Gu, Zhouye

2012

Gu, Z. (2012). Advanced video coding based on matrix decomposition. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/55344>

<https://doi.org/10.32657/10356/55344>

Advanced Video Coding Based on Matrix Decomposition

A thesis submitted to
School of Computer Engineering
Nanyang Technological University

by

GU Zhouye

in Partial Fulfillment of the Requirement for the Degree of
Doctor of Philosophy (Ph.D)

2012

Acknowledgments

During my time as a Ph.D student, I received help, advice and support from many people around me. I would like to take this opportunity to thank them for the help that I received over the past years.

First of all, I would like to thank my supervisor Dr. Weisi Lin. During my enjoyable years as a Ph.D candidate in School of Computer Engineering, Weisi has provided me invaluable advice, unwavering support and patient guidance. Without his patience and constructive suggestions, it would not have been possible for the project to be an enriching experience for me.

I would like to thank Prof. Bu Sung Lee and Prof. Chiew Tong Lau for their helps and precious advice in my research. I also enjoyed working with the labmates at the CeMNet and CeMNet Annex Labs. I wish to thank them for their valuable suggestions in the discussions: Anmin Liu, Chenwei Deng, Huan Yang, Jinjian Wu, Lining Zhang, Lu Dong, Manoranjan Paul, Narwaria Manish, Yuming Fang and other students and staff in the lab. I have had my most memorable years with them in my Ph.D study.

I also acknowledge the funding received from the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF Tire 2) throughout my studies.

I would like to thank in advance the thesis examiners for accepting to be part of the committee, and for their comments and suggestions to improve this thesis.

Last but not the least, I would like to thank all my friends and relatives who help and support me these years during the whole candidature period.

Abstract

Video coding has been widely used in everyday work and life. Most of the video coding technologies are based on a set of principles that reduce the redundancy in digital video, including mainly temporal, spatial and statistical redundancies. In this study, we explore new methodology for further redundancy reduction. Combining the matrix decomposition algorithms, we have proposed three matrix decomposition based techniques that further enhance compression performance, when compared with the relevant state of the art technology.

Firstly, we propose a temporal redundancy reduction scheme based upon Two-Dimensional Singular Value Decomposition (2D-SVD) without resorting to motion estimation (ME). Its computational complexity is much lower than that of ME-based video coding scheme. By exploring the energy compaction property of 2D-SVD coefficient matrices, high coding efficiency is achieved compared with other non-ME based methods.

Secondly, for spatial redundancy reduction, the existing video codecs use Discrete Cosine Transform (DCT), which is originally designed based on the spatial characteristic of natural image pixels, for predicted residual pixels compression; Since the spatial characteristic of predicted residual pixels differs from that of natural image pixels, we develop a new orthogonal transform—Rotated Orthogonal Transform (ROT) that yields better compression on the prediction residual than the DCT.

Lastly, for statistical redundancy reduction, the entropy coders that originally

designed for lossy video coding are also used for lossless video coding in the current standard video coding scheme. Since the coefficients generated by lossy and lossless coding processes have different statistical distributions, we therefore propose a new methodology based on Mode Dependent Template (MD-Template) and scan order for better intra lossless video coding.

The algorithms proposed in this thesis are developed from three different matrix decomposition problems. They have been validated on a large number of standard video sequences. We have performed careful experimental analysis and demonstrated that the proposed techniques overall perform better than the existing relevant methods.

Contents

Acknowledgments	i
Abstract.....	ii
Contents	iv
List of Figures.....	vii
List of Tables	ix
List of Abbreviations	x
Chapter 1 Introduction.....	1
1.1 Motivation and Background	1
1.2 Objective and Scope of This Work.....	3
1.3 Major Technical Contributions	4
1.4 Organization of the Thesis	5
Chapter 2 Literature Survey.....	7
2.1 Video Coding Techniques.....	7
2.1.1 Video Coding Fundamentals.....	7
2.1.2 H.264/AVC Standard for Lossy and Lossless Video Coding	9
2.1.3 Coding Performance Evaluation	12
2.2 More Related Issues in Video Coding	14
2.2.1 Non-Motion Estimation Based Video Coding	14
2.2.2 Transforms for Prediction Residual Coding.....	16
2.2.3 Entropy Coding for H.264/AVC Lossless Coding	19
2.3 Matrix Decomposition	20
2.3.1 Vector Norms	21
2.3.2 Concept of Matrix Decomposition.....	25
2.3.3 Matrix Decomposition for Video Coding	26
Chapter 3 Two-dimensional Singular Value Decomposition Based Video Coding	29
3.1 Overview	30
3.2 2D-SVD for Video Decomposition	31

3.2.1	2D-SVD Fundamental.....	31
3.2.2	2D-SVD for Video Decomposition.....	35
3.3	Efficiency of 2D-SVD Coding.....	36
3.4	Framework of 2D-SVD Coding.....	40
3.4.1	2D-SVD Encoder.....	40
3.4.2	2D-SVD Decoder.....	45
3.5	Experimental Results.....	46
3.5.1	Objective Coding Results Comparison.....	48
3.5.2	Visual Quality Comparison.....	51
3.5.3	Computational Complexity.....	53
3.6	Concluding Remarks.....	55
Chapter 4 Rotated Orthogonal Transform for Motion-Compensation Residual		
Coding		57
4.1	Overview.....	58
4.2	Problem Formulation.....	60
4.3	The ROT Algorithm.....	65
4.3.1	Details of the ROT Algorithm.....	65
4.3.2	Analysis with Varying TR Sizes.....	67
4.3.3	TR Selection Rate Analysis.....	69
4.3.4	Convergence Performance Analysis.....	71
4.3.5	Energy Compaction Comparison with DCT.....	73
4.4	Experimental Results.....	76
4.4.1	Objective Coding Results Comparison.....	77
4.4.2	Visual Quality Comparison.....	80
4.4.3	Computational Overhead Analysis.....	81
4.5	Concluding Remarks.....	83
Chapter 5 Mode Dependent Templates and Scan Order for Intra Lossless Coding		
		85
5.1	Overview.....	86
5.2	Problem Formulation.....	90
5.2.1	MD-Template Learning.....	90
5.2.2	Scan Order for Each MD-Template.....	97
5.3	Algorithms and Analysis.....	99
5.3.1	Proposed Algorithms.....	99

5.3.2	Statistics Before and After Coding with the Proposed Algorithms.....	100
5.4	Experimental Results	102
5.4.1	Number of MD-Templates to Use.....	103
5.4.2	Coding Performance Comparison	104
5.4.3	Computational Complexity	106
5.5	Concluding Remarks.....	107
Chapter 6	Summary and Future Work	109
6.1	Summary	109
6.1.1	2D-SVD for Non-Motion Estimation Based Video Coding	109
6.1.2	Rotated Orthogonal Transform for MC-residual Coding.....	111
6.1.3	MD-Templates and Scan Orders for H.264 Intra Lossless coding.....	112
6.2	Future work.....	112
References		116
Publications		131
	Journal Papers	131
	Conference Papers	132

List of Figures

Figure 1.1 Block diagram of video encoder.....	3
Figure 2.1 Correspondence between issues in video coding and our solutions based on matrix decomposition techniques.....	28
Figure 3.1 SVD decomposition.....	32
Figure 3.2 DCT decomposition.....	32
Figure 3.3 2D-SVD decomposition.....	32
Figure 3.4 Coefficient distribution comparison for DCT and 2D-SVD.....	37
Figure 3.5 The proposed 2D-SVD encoder.....	42
Figure 3.6 2D-SVD coefficients after 2D-SVD with different GOPs.....	43
Figure 3.7 2D-SVD coding result comparison at GOP 8, 16, 24 and 48	45
Figure 3.8 The 2D-SVD decoder	46
Figure 3.9 R-D curves of 4 low complexity codecs.....	48
Figure 3.10 Subjective coding result comparison of the 90th frame of QCIF <i>Akiyo</i>	51
Figure 4.1 DCT result of an original image and its MC-residual image.....	59
Figure 4.2 Transform Regions (areas in boxes) for CIF <i>Mobile</i> MC-residual frame	62
Figure 4.3 The gradient-based searching algorithm.....	66
Figure 4.4 The ROT algorithm	67
Figure 4.5 Comparison of compression efficiency with varying TR sizes	68
Figure 4.6 TR selection results (Solid rectangles: TRs choose to use DCT, Dash rectangles: TRs choose to use ROT)	69
Figure 4.7 Convergence performance with the proposed algorithm for the top-left TR.....	71
Figure 4.8 Averaged iterations need for each sequence.....	72
Figure 4.9 Comparison of base functions of DCT and the base functions derived by ROT.....	74
Figure 4.10 Comparison transformed coefficients of DCT and ROT	74
Figure 4.11 Comparison of energy retained by the number of coefficients of DCT and ROT	75
Figure 4.12 R-D performance comparison.....	78
Figure 4.13 Subjective coding performance comparison of the frame of CIF <i>Mobile</i>	81
Figure 4.14 Average coding time increase for 10 testing sequences	82

Figure 5.1 Nine intra prediction modes[91].....	88
Figure 5.2 Comparison of H.264 intra lossy and lossless coding process.	88
Figure 5.3 A demonstration of X_i classification process	91
Figure 5.4 Probability distribution of absolute value of first 20 non-zero coefficients in residual blocks.....	92
Figure 5.5 The top 1st to 16th learned MD-Templates from training set A for Mode 0.....	95
Figure 5.6 The top 1st to 16th learned MD-Templates from training set B for Mode 0.....	95
Figure 5.7 The 16 MD-Templates used for Mode 0	96
Figure 5.8 The 16 MD-Templates used for Mode 1	96
Figure 5.9 The 16 MD-Templates used for Mode 3	96
Figure 5.10 The 16 MD-Templates used for Mode 4	97
Figure 5.11 The 16 MD-Templates used for Mode 2	97
Figure 5.12 Examples for scan orders.....	97
Figure 5.13 A demonstration of H.264 intra lossless coding with and without MD-Templates...	98
Figure 5.14 The pseudo-code of Algorithm 1	99
Figure 5.15 The pseudo-code of Algorithm 2	100
Figure 5.16 Probability distribution of non-zero coefficients vs. scan position.....	102
Figure 5.17 Percentage of coding time increase of Algorithm2 as compared to H.264-LS.....	107

List of Tables

Table 3.1 Number of coefficients for 1D-SVD and 2D-SVD	37
Table 3.2 Average numbers of zero DCT and 2D-SVD coefficients at same quantization steps (the figures in bold represent the better cases for Zeros Per Block and MSE)	38
Table 3.3 Coding Gain comparison of the DCT and the proposed 2D-SVD	39
Table 3.4 Bits (bpp--bits per pixel) required for eigenvector Y_l^i, Y_r^j s for a GOB	42
Table 3.5 Subjective coding efficiency comparison	50
Table 3.6 Subjective coding efficiency comparison with Mean Opinion Score	52
Table 3.7 Estimated encoding complexity comparison for 4CIF ‘Harbor’ at 34dB	54
Table 3.8 Estimated encoding complexity comparison for 4CIF ‘Ice’ at 34dB	54
Table 4.1 Average PSNR(dB) comparison for CIF sequences (200 frames) with varying TR sizes (The bold number indicates the best result among four TR sizes)	68
Table 4.2 TR Selection Rate (SR) in Percentage	69
Table 4.3 Result of Modified Temporal Information (Modified _TI)	70
Table 4.4 Coding Gain (dB) comparison of DCT and ROT	75
Table 4.5 BD-PSNR and BD-Bitrate comparison among DDCT, adaptive orthogonal transform, 1D transform and proposed ROT	79
Table 4.6 Subjective coding efficiency comparison	81
Table 5.1 Training sequences set A	91
Table 5.2 Probability distribution of selected Intra prediction modes	92
Table 5.3 Codeword table for coefficients a_i	94
Table 5.4 Training sequences set B	95
Table 5.5 Selection rate comparison	101
Table 5.6 Number of zero coefficients comparison	102
Table 5.7 Compression efficiency comparison with different MD-Templates numbers	104
Table 5.8 Performance comparison of Heo’s[35] method and the proposed method	105
Table 5.9 Performance comparison of Wei’s[37] method and the proposed method	105
Table 5.10 Probabilities of times of CAVLC used in Algorithm 2	107

List of Abbreviations

2D/3D	Two/Three Dimensional
3G/4G	Third/Fourth Generation mobile telecommunications
bpp	bits per pixel
CABAC	Content Adaptive Binary Arithmetical Coding
CAVLC	Content Adaptive Variable Length Coding
DCT	Discrete Cosine Transform
DPCM	Differential Pulse Code Modulation
EPZS	Enhanced Predictive Zonal Search
FPS	Frames Per Second
GOB	Group of Blocks
GOP	Group of Pictures
HEVC	High Efficiency Video Coding
ITU-T	International Telecommunication Union-Telecommunication Standardization
J2K	JPEG 2000
JPEG	Joint Photographic Experts Group
JPEG-LS	Lossless JPEG
KLT	Karhunen-Loeve Transform
MB	Macro-block
MC	Motion Compensation
MD-Template	Mode Dependent Template
MDT	Mode Dependent Transform
ME	Motion Estimation
MPEG	Motion Picture Expert Group
MSE	Mean Squared Error
PSNR	Peak Signal Noise Ratio
QCIF	Quarter Common Intermediate Format
QP	Quantization Parameter
QS	Quantization Step
RD	Rate-Distortion
RGB	Red-Green-Blue
ROT	Rotated Orthogonal Transform
SAD	Sum of Absolute Difference
SLRAM	Simultaneous Low Rank Approximation of Matrices
SSD	Sum of Squared Difference
SVD	Singular Value Decomposition
TR	Transform Regions
TSS	Three Step Search
VLC	Variable Length Coding
WDR	Wavelet Difference Reduction

Chapter 1

Introduction

1.1 Motivation and Background

Digital video is essentially a sequence of pictures displayed over time. Each picture of a digital video sequence is a 2D projection of the 3D world. Digital video is thus captured as a series of digital pictures or sampled in space and time from an analog video signal. A frame of digital video or a picture can be seen as a matrix of pixels. Each pixel value represents the color and intensity values of a specific spatial location at a specific time. The Red-Green-Blue (RGB) color space is typically used to capture and display digital pictures. Each pixel is thus represented by R, G, and B components. The matrix of pixels that constitutes a picture is actually three matrices with one matrix for each of the RGB components. A resolution of 8 bits per component is usually sufficient for typical consumer applications.

Consider a digital video sequence at a standard definition TV picture with resolution of 720×480 and frame rate of 30 Frames Per Second (FPS). If a picture is represented using the RGB color space with 8 bits per component or 3 bytes per pixel, size of each frame is $720 \times 480 \times 3$ bytes. The disk space required to store one second of video is

Chapter 1. Introduction

$720 \times 480 \times 3 \times 30 = 31.1$ Mega Bytes. A one hour video would thus require 112 Giga Bytes. To deliver video over wired and/or wireless networks, bandwidth required is $31.1 \times 8 = 249$ Mbps. In addition to these extremely high storage and bandwidth requirements, using uncompressed video will add significant cost to the hardware and systems, if it is not totally impossible. Digital video compression is thus necessary even with exponentially increasing bandwidth and storage capacities.

Fortunately, digital video has significant redundancies and eliminating or reducing those redundancies results in compression. Video compression can be lossy or lossless. A number of video coding standards, i.e., MPEG-2, MPEG-4, H.261, H.263, and H.264 [1-5], achieve compression mainly by reducing three types of redundancies: temporal, spatial and statistical redundancies. Figure 1.1 shows a basic block diagram of video encoder. By Motion Estimation (ME) and Motion Compensation (MC) [6-7], temporal redundancy is considerably removed, and very little energy is left in the residual frame; by transform, spatial redundancy is largely removed since most of the energy is concentrated in few transform coefficients; by entropy coding, the statistical redundancy is further reduced. Although these techniques have made it practical to store, transmit and manipulate digital image and video information under currently storage systems and data networks, rooms for performance improvement are still worth to explored, in order to make the related products and services more cost effective, as well as enabling more new functionalities.

Chapter 1. Introduction

Figure 1.1 Block diagram of video encoder

1.2 Objective and Scope of This Work

The objective of this study is to explore new methodologies to improve video coding performance, in removal of temporal, spatial and statistical redundancies, by introducing matrix decomposition to video coding task.

In particular, we try to address the following three problems of current video codecs.

Firstly, although motion estimation (ME) is efficient for temporal redundancy removal, it brings high computation which results in long computational time and high energy consumption. For scenarios which could not afford the intensive computation of ME, how to explore temporal redundancy efficiently without resorting to ME?

Secondly, the Motion Compensation residual (MC-residual) is transformed with the Discrete Cosine Transform (DCT) [8] in the current standard coding technology. Since

Chapter 1. Introduction

the spatial characteristics of the MC-residual pixels are different from that of original pixels, DCT is less efficient for residual data compression. To better explore spatial redundancy in MC-residual pixels, how to design a new transform that better fits the characteristic of MC-residual?

Thirdly, for lossless video coding, the same entropy coder, originally designed for lossy video coding, is also used for lossless video coding. Since the coefficients generated by lossy and lossless coding have different statistical distributions, how to develop new technology to better explore statistical redundancy for lossless video coding?

1.3 Major Technical Contributions

This thesis has presented new coding methods which enhance video coding performance inspired by the development in matrix decomposition.

Firstly, we propose a new temporal redundancy removal scheme based upon Two-Dimensional Singular Value Decomposition (2D-SVD) without resorting to ME. By exploring the energy compaction property of 2D-SVD coefficient matrices, high coding efficiency is achieved with low computation.

Secondly, we develop a new orthogonal transform—Rotated Orthogonal Transform (ROT) that can perform better on the MC-residual than the DCT for coding purpose. We derive the proposed ROT based on orthogonal-constrained L1-Norm minimization problem for its sparse property. An iterative algorithm is proposed to solve the ROT problem.

Lastly, we propose a Mode Dependent Template (MD-Template) and scan order based method for intra lossless coding. By exploring the statistical redundancy of the

Chapter 1. Introduction

prediction residual in the H.264/AVC intra prediction modes, the scanned coefficient sequence fits the H.264/AVC entropy coders better. A fast implementation algorithm is also designed.

1.4 Organization of the Thesis

The whole thesis is divided into six chapters.

Chapter 1 (this chapter) gives an introduction about the thesis, including the motivation and background, objective and scope, technical contribution and thesis organization.

Chapter 2 generally reviews the major related existing work, encompassing the basics of video signal coding techniques, literature related to our research topics and fundamental of matrix decomposition.

Chapter 3 proposes a video coding scheme based upon two-dimensional Singular Value Decomposition (2D-SVD), which exploits basic temporal correlation in visual signal without resorting to motion estimation. The proposed scheme is for better compromise of computational complexity and temporal redundancy reduction, compared with the ME based techniques.

Chapter 4 discusses the issue with transformation module and introduces a new orthogonal transform—Rotated Orthogonal Transform (ROT) that can perform better on the MC-residual than the DCT for coding purpose. With small computation overhead, the ROT provides higher compression efficiency for the MC-residual than DCT, especially for high/complex motion videos.

Chapter 5 addresses the problem with the entropy coding module and proposes a Mode Dependent Template (MD-Template) and scan order based method for better intra

Chapter 1. Introduction

lossless coding. A fast implementation algorithm is also designed. With little computation increase the proposed fast algorithm achieves better compression compared with the current H.264/AVC standard.

Chapter 6 closes the thesis with a summary of the main research work performed and several directions for further studies.

Chapter 2

Literature Survey

In this chapter, we give a brief overview of the major existing work relevant to visual signal coding and matrix decomposition, since these topics are the closest to our research in this thesis and our research work is grounded on them.

2.1 Video Coding Techniques

2.1.1 Video Coding Fundamentals

Video is a collection of natural images (frames) arranged in certain order. Temporal redundancy exists between these images besides the spatial redundancy within each natural image. Video coding thus achieves compression by mainly removing three kinds of redundancies: spatial redundancy, temporal redundancy and statistic redundancy.

Firstly, neighboring pixels within an image tend to be highly correlated. As such, it is desired to use a transform to concentrate energy into fewer, de-correlated coefficients.

To measure efficiency of a transform, the transform coding gain, G_{TC} , is often used [9–12]. It is computed as the ratio of the arithmetic to the geometric means of the transform coefficient variances, and expressed in dB as (2.1) below, where N is the total number of coefficients and σ_i^2 is the variance of the i -th coefficient.

Chapter 2. Literature Survey

$$G_{TC} = 10 \log_{10} \left(\frac{\frac{1}{N} \sum_{i=1}^N \sigma_i^2}{\left(\prod_{i=1}^N \sigma_i^2 \right)^{1/N}} \right) \quad (2.1)$$

In general, the coding gain is understood as the SNR (signal-to-noise ratio) improvement of compression result of a coder under test over the straightforward PCM quantization at the same bitrate [9–12]. The Discrete Cosine Transform (DCT) [8] is an orthogonal transform that has been shown to be near optimal for a large class of natural images in energy concentration and de-correlating. Thus it is adopted in most of video coding standards for spatial redundancy removal, i.e. MPEG-2, MPEG-4, H.261, H.263, and H.264[1-5] . In many of standards [1-5], DCT is used for both natural image pixels de-correlation and residual image pixels de-correlation.

Secondly, consecutive video frames will be similar except for changes induced by objects motion. Therefore, pixels in adjacent frames are also highly correlated and this leads to temporal redundancy. The most commonly used technique for temporal redundancy removal is Block-based Motion Estimation (ME) [6-7]. In ME, each frame is divided into small blocks, i.e. 8x8 blocks [4], 16×16 macro-blocks (MBs) [5], and each block in the current frame is predicted from a block of equal size in the reference frame. The offset between the two blocks is known as a motion vector [6-7]. The error between the current block and the similar block in the reference frame is encoded and transmitted along with the motion vector for the block. To exploit the redundancy between neighboring block vectors (e.g., for a single moving object occupying multiple blocks), it is common to encode only the difference between the current and previous motion vectors into the bit stream.

In some cases, ME is impossible due to its high complexity (the computational complexity of ME varies from 50% to 90% for a typical video coding system [13],

Chapter 2. Literature Survey

[14]). Therefore, researches aiming at exploring temporal redundancy without ME is conducted [9]-[14]. Coding without ME results in reduction of compression capability, yet low complexity [6] and robustness to error due to transmission [7].

Thirdly, after quantization, which is often controlled by quantization parameter (QP) [2-5], the transformed coefficients need to be saved for storage and transmission. It is observed that the distribution of the value of these coefficients is highly uneven and this leads to statistic redundancy [20][21]. Entropy coding is a lossless compression scheme that maps quantized transform coefficient values to Variable Length Coding (VLC) codes [20]. The basic principle is to assign long codes to infrequent values and short codes to frequent values. Based on the statistics of quantized coefficients by various video coding schemes, several entropy coding methods (Run Length Coding, Huffman coding, Content Adaptive Variable Length Coding, Content Adaptive Binary Arithmetical Coding, etc.) have been proposed and the video compression efficiency is further improved.

2.1.2 H.264/AVC Standard for Lossy and Lossless Video Coding

A video coding standard is the language with which a video encoder and a decoder communicate. The development of international video coding standards has evolved through ITU-T H.261, H.262/MPEG-2, H.263/MPEG-4(part 2), and H.264/MPEG-4(part 10) which are mainly designed for lossy video coding; ITU-T and MPEG are the International Telecommunication Union-Telecommunication Standardization and the Motion Picture Expert Group, respectively. As we know, H.264 video compression has employed more complicated techniques to achieve higher coding efficiency than its

Chapter 2. Literature Survey

predecessors did. Nevertheless, the fundamental technology behind these standards remains similar: motion compensated prediction to remove inter-frame redundancy, e.g., via ME, followed by transform coding for energy compaction that allows effective entropy coding and has been proven to be exceptionally effective to compress video data.

H.264 includes many profiles. Each profile is designed for specific applications and there is no need to support all applications in one profile. For example, Baseline Profile is primarily for low-cost applications, and this profile is used in video conferencing and mobile applications; High Profile is primary for broadcast and disc storage applications such as Blu-ray Disc storage format. The standard also contains four additional all-intra profiles, which are defined as simple subsets of other corresponding profiles. These are mostly for camcorders, editing, and professional applications.

One reason for the improved coding efficiency of H.264 is that a more extensive search/optimization for the best coding mode is used. The optimization process is usually referred as Rate-Distortion (R-D) optimization [22-23], which compares different coding modes in terms of coding efficiency (measured by the R-D cost). To make a R-D optimized decision during encoding a block of video data, the block has to be encoded a number of times before the encoder can arrive at the best mode decision.

The other reason for the improved coding efficiency of H.264 is the adoption of quarter-pixel motion estimation [5]. Quarter-pixel accuracy allows the calculations used to detect motion to be more accurate than MPEG-2, which only uses Half-Pixel accuracy. Combining the multiple reference frames and variable block sizes, quarter-pixel motion estimation significantly enhances the compression performance. Recently, the latest High Efficient Video Coding (HEVC) standard has been released [116]. Compared with H.264, HEVC achieves 30%-50% bitrate reduction by introducing

Chapter 2. Literature Survey

larger prediction and transformation block size, more flexible partition strategy, more intra prediction directions, new in-loop filter, more flexible inter prediction scheme and many other new tools.

On the other side, lossy video coding is not applicable for applications where no loss of pixel values is tolerable since it discards some of the original image information that cannot be later recovered. In some applications, lossless video coding is required where only the statistically-redundant spatial and temporal information is allowed to be removed, and the process is reversible to guarantee that the reconstructed signal is mathematically the same as the original one.

Examples of lossless coding applications are medical imaging (i.e., alteration of the original data are not allowed in order to make sure that physicians analyze pristine diagnostic images)[25], satellite remote sensing (since every piece of information is acquired with high cost and therefore it is better to keep all acquired visual signals), and film archiving and studio applications (where the genuineness of the original images should be preserved)[24].

Lured by the growing lossless coding market, H.264/AVC standard decided to provide improved functionality for lossless video coding. Firstly, Pulse-Code Modulation (PCM) macroblock coding mode has been added in H.264/AVC standard. In this mode, the values of the original image pixels are sent directly without prediction, transform, and quantization. Although the PCM mode achieves lossless coding, its compression ratio is limited [26].

Later on, an extension to the original H.264/AVC, known as the fidelity range extensions (FRExt) [27], [28], has been developed by Joint Video Team (JVT). In FRExt, a transform-bypass [29] lossless mode was included. Compared to the PCM mode, this mode employs prediction and entropy coding for lossless coding. To further

Chapter 2. Literature Survey

enhance the coding performance for lossless coding, however, more efficient coding techniques for prediction are still required.

In the meantime, a new intra prediction method called sample-wise Differential Pulse-Code Modulation (DPCM) [26], [30], [31] was introduced for lossless intra prediction. Its main consideration is that the sample next to the one to be predicted is often a better predictor than a sample in a neighboring block several samples farther away. As a result, sample-wise DPCM has been shown to provide better compression performance without incurring a major increase in the computational complexity, and was subsequently adopted as a part of the new draft amendment for the H.264/AVC standard [32].

In lossless coding profiles, since the prediction residues are entropy encoded directly rather than undergoing transform and quantization first, the coefficients generated by lossy and lossless coding have different statistical distributions [34][35]. Nevertheless, in H.264 standard, the entropy coder for lossless coding is the same as lossy coding one, which is originally designed for quantized coefficients [26][33]. This makes the current H.264 standard less effective for lossless video coding.

2.1.3 Coding Performance Evaluation

Most of video compression applications are lossy for its high compression ratio. Videos after lossy compression are similar but not identical to the original video, since the quantization step during compression is not reversible.

The most traditional ways of evaluating quality or distortion of digital video processing system are calculation of the Peak Signal-to-Noise Ratio (PSNR) between the original video signal and compressed video signal. It is defined as follows:

Chapter 2. Literature Survey

$$PSNR = 10\log_{10}\left(\frac{Max_I^2}{MSE}\right) \quad (2.2)$$

where Max_I is the maximum possible value of the image. When the pixels are represented using 8 bits per sample, Max_I is 255. MSE is the Mean Squared Error between the original image and reconstructed image. The PSNR of the video is the average PSNR of each frame. Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB; a higher PSNR indicates better coding quality. However, to evaluate the coding performance of a codec we should also take the coded bitrate into consideration, since with the increase of bits for compression the PSNR also increases. By plotting the bitrate-PSNR pair as a dot in a two-dimensional coordinate system, we can draw a Rate-Distortion (R-D) curve to analyze how much compression can be achieved using lossy compression methods. The conventional way to compare the performance of two video codecs is to calculate the average PSNR difference over the same range of bitrate. Usually, 4 PSNR-bitrate pairs derived with different quantization parameters (QP) are used to fit a third order polynomial curve as:

$$PSNR = a + b \times bitrate + c \times bitrate^2 + d \times bitrate^3 \quad (2.3)$$

when we get the polynomial coefficients a, b, c and d , we can derive the PSNR difference by calculating the difference between the two integrals of two R-D curves and divide it by the integration interval.

Recently, researchers argue that conventional averaged PSNR calculation gives more weight on the high bitrate part of the R-D curve [22] since value of PSNR is in logarithmic scale while the value of bitrate is in linear scale. By converting the bitrate into logarithmic scale with $\log(bitrate)$ during R-D performance comparison, they propose a new performance comparison scheme, namely Bjontegaard-Delta(BD)-PSNR or Bjontegaard-Delta(BD)-Bitrate [36], that treats the high bitrate part and low bitrate

Chapter 2. Literature Survey

part more equally. This method has been widely adopted in coding performance comparison.

For lossless coding, since no distortion is introduced, the performance is often evaluated by Compression Ratio or Bit Saving [37] which is defined in (2.4) and (2.5). Larger Compression Ratio or Bit Saving indicates better compression performance.

$$\text{Compression Ratio} = \frac{\text{Original Video Size}}{\text{Compressed Video Size}} \quad (2.4)$$

$$\text{Bit Saving}(\%) = \frac{\text{Original Video Size} - \text{Compressed Video Size}}{\text{Original Video Size}} \times 100 \quad (2.5)$$

2.2 More Related Issues in Video Coding

2.2.1 Non-Motion Estimation Based Video Coding

As we introduced in Section 2.1, although motion estimation removes temporal redundancy efficiently, it also brings high computation. The computation load is especially intensive when quarter-pixel motion estimation is applied. It is estimated that for H.264 codec, the computational complexity of inter-frame coding is 5 -10 times higher than that of intra-frame (i.e., without ME) coding [13],[14]. The consequences of high complexity are long computational time, high energy consumption, short battery-life [15]-[17] and high cost of hardware implementation [18],[19]. Besides, the hybrid coding system also causes inter-frame dependency in decompression, and this is a serious problem for an error-prone channel such as wireless channel where error occurrence is frequent and error loss is substantial. In addition, with applications in which frames need to be edited and changed, all the frames affected by the changes have to be encoded again in an ME-based coder, because of the inter-frame dependency

Chapter 2. Literature Survey

[16]; therefore, hybrid coding is not popular for situations requiring video editing and easy access to an individual frame.

On the other hand, intra-frame video codecs, e.g. H.264/AVC Intra [5] and Motion-JPEG2000 (Motion-J2K)[38],[39], do not suffer from the aforementioned problems since each frame is coded independently. Intra-frame codec enables random access to the frames, and thus is useful for video editing applications. Moreover, with the transmission through an error-prone channel, the error that happens in one frame would not affect the subsequent frames since they are decoded independently. However, although H.264/AVC Intra is reported to yield higher coding efficiency compared with Motion-J2K[40],[41] for most testing sequences, its coding efficiency is still low compared with hybrid codecs at the same bit rate, since it does not exploit the temporal redundancy of successive frames.

Three-dimensional (3D) transform coding [118–123] is also an alternative video coding approach without motion estimation. In video coding, the application of the Discrete Cosine Transform (DCT) or Discrete Wavelet Transform (DWT) along the temporal axis is advantageous over motion compensation prediction schemes because the structure can be non-recursive, which avoids error propagation. Several techniques have been proposed to compress video using the 3D-DCT. Bauer and Sayood [118] propose the use of dynamic code selection to code 3D-DCT coefficients. Servais and de Jager [119] reduce the entropy of the 3D-DCT coefficients by exploiting the relationship between them and the 2D-DCT coefficients of the first frame in the group. Several 3D-DWT approaches were also proposed in the last decades. Kim and Pearlman extended the Set Partitioning in Hierarchical Trees (SPIHT) algorithm from 2D images to 3D video and developed a 3D-SPIHT video coder [121], where segments of 16 consecutive frames are grouped together for temporal domain wavelet filtering. Other 3D-DWT

Chapter 2. Literature Survey

based approaches include lifting-based 3D wavelet transform with deformable mesh motion compensation proposed by the group of David Taubman [123], 3D Embedded Subband Coding with Optimal Truncation (3D ESCOT) [122] and so on. Compared with motion estimation based video coding scheme, 3D transform coding scheme achieves comparable performance for certain kinds of video sequences with slow motion [118], [119]. The price to be paid is a longer encoding/decoding delay and the requirement for a large memory size.

Recently, several alternative non-ME based video coding schemes have been proposed for applications where low complexity compression is required [15]-[19]. Chiu, et al. proposed a low complexity video codec using pixel-wise conditional differential replenishment [19]; Kamath, et al. proposed a Motion-JPEG with differential pulse code modulation (MJPEG-DPCM) video codec where differential encoding between coefficients of adjacent frames is carried out [17]. Law, et al. proposed a fast video codec using Wavelet Difference Reduction (WDR) algorithm without resorting to ME [42]. In [42][17][19], the improvement of coding efficiency compared with either Motion-JPEG or Motion-JPEG2000 is reported; however, they are based on coding the difference between consecutive frames, and lose the frame independence of video and therefore intra-frame codecs' error resilience property. In [15], Amir Asif proposed a noncausal predictive video coder for low bitrate multimedia applications. Although no ME is used in this codec, the iterative algorithm in their codec is also very complex and time consuming.

2.2.2 Transforms for Prediction Residual Coding

The transform used is the key component for spatial redundancy removal. A good

Chapter 2. Literature Survey

transform should concentrate energy to as few coefficients as possible. Typically, the same transform is used to transform both image intensities in image coding and the Motion-Compensation residual (MC-residual) in inter-coding for video. For example, the Discrete Cosine Transform (DCT) is used to compress a natural image in the JPEG standard as well as MC-residual in many video coding standards, e.g. H.261/3, MPEG-1/2/4 and H.264/AVC. However, MC-residual has different spatial characteristic from natural image. In a natural image, pixels are highly correlated; while in a MC-residual image, pixels have low correlation and their spatial characteristic changes from area to area in the same MC-residual frame [48,49].

Recently, new transforms have been developed that take the advantage of local spatial features in MC-residual with wavelets [50-54]. Candes and Donoho have proposed the curvelets [50], which yields an optimal approximation of smooth images because of their high degree of directionality. Originally designed for the continuous case, curvelets cannot be easily used in the discrete case. To overcome this problem, Do and Vetterli have proposed the contourlets [51] which use the same principle as the curvelets but are directly defined in the discrete case.

Bandelets[52] and the second generation bandelets [53] have been introduced by Mallat recently. The second generation bandelets apply a geometric orthogonal transform to orthogonal wavelet coefficients. Each geometric direction is assigned with a different transform. As a result, edge detection and warping are required to apply the proper transform.

Introduced by Velisavljevic and Vetterli et al., directionlets [54] have recently received much attention for research efforts. Working at critical sampling by applying separable filtering along horizontal, vertical directions and cosets of numerical co-lines, directionlets provide an efficient tool for nonlinear approximation of images.

Chapter 2. Literature Survey

However, none of these methods uses block-based transforms, which is an important part in most video coding standards and methods [2-5] since they can be efficiently combined with block-based motion compensation.

Karhunen-Loeve Transform (KLT) is also good orthogonal transform for energy compaction. Moreover, KLT has been adopted in HEVC [55]. In HEVC, for each intra prediction mode, a pair of KLT matrices is learned. Within a certain intra prediction mode, intra predicted residual blocks have strong correlation. Therefore, a pair of de-correlation matrices, better than DCT, could be derived with KLT. However, as introduced in Subsection 2.1.1, for MC-residuals, the residual blocks are predicted with various motion vectors and reference blocks such that it is difficult to classify them into several modes/classes as intra predicted residual blocks for further de-correlation. Consequently, KLT is not used for MC-residual blocks transformation [55].

Another important type of such techniques, called directional (orientation-adaptive) transform, which combines directional prediction and DCT/DWT, has been developed in the literature [56-64][131]. The key idea in the directional transform is to explore the direction information in the residual/image block by adaptive scan in both spatial and transformed domains (in either 1D or 2D manner) to derive a more sparse transformed matrix for compression. In [64], the Rotational Transform is proposed as a 2nd transform that could be applied after the DCT operation. Compression ratio improvement is observed. However, the Rotational Transform is only applied for intra prediction residuals [64]. In [58-59], Directional DCT [58] and 1D-Transform[59] show that a significant performance gain can be achieved with directional transforms for image/residual with strong texture patterns.

Chapter 2. Literature Survey

2.2.3 Entropy Coding for H.264/AVC Lossless Coding

After transform, quantized coefficients will be further coded with entropy coder for statistical redundancy removal. After transform, most coefficients have low magnitude, since most energy is concentrated into only few high magnitude coefficients. Entropy coders explore the uneven distribution of these coefficients to achieve compression. In H.264/AVC, the entropy coders designed for lossy coding are also used for lossless coding. However, in lossless coding, the data sent to entropy coders are residual errors, which are the differential pixel values between the original and the predicted pixel values without transform and quantization. Residual data differ from quantized transform coefficients in at least two aspects: 1) it has fewer zero entries; 2) all the positions in the prediction residual block have almost the same probability of having non-zero entries. Thus, the zigzag scan and the conventional entropy coders in H.264/AVC cannot provide the best coding performance for lossless video coding.

Several studies [65-71] have improved lossless coding based on H.264/AVC. Since H.264 lossless coding adopts a transform-bypass scheme, studies in [65]–[67] have explored possible improvements using transforms due to the energy compaction benefit. Wei et al. [65] proposed an adaptive truncation algorithm for Hadamard transformed H.264/AVC Lossless Video Coding. Ding et al. [66] proposed lossless coding algorithms which use the discrete wavelet transform to substitute the integer discrete cosine transform in H.264/AVC. Ding et al. [67] also suggested two-layer coding algorithms to achieve both lossy and lossless coding using H.264/AVC. Moreover, the authors in [66][67] use CABAC and CAVLC adaptively based on the image content variation.

Besides transforms, some other techniques [34][35][68][69] have been investigated

Chapter 2. Literature Survey

for improving the lossless coding of H.264/AVC. Lee et al. [26] presented an efficient intra lossless coding method based on sample by sample (SbS) differential pulse code modulation (DPCM) for five modes of H.264/AVC lossless coding and this technique was adopted in a new draft amendment for the H.264/AVC standard [32]. Nam et al. [34] proposed an additional pixel-wise prediction mode to cooperate with the original intra modes in H.264/AVC. Heo et al. [35] modified three main coding processes of the conventional CAVLC in the level and zero coding parts and achieved considerable performance improvement. Kim et al. proposed an improved entropy coding method [71] as well as an efficient Two-layered Residual Coding (TRC) scheme [70] that boost the compression efficiency of H.264 intra lossless coding significantly. In [68], Zhang et al. observed that the prediction residuals are not white. The redundancies can be reduced by further predictions since the residuals are in the form of granular noise. They thus proposed a compression scheme based on residual image prediction and coding. In [69], an adaptive scan order for CAVLC was proposed. For each intra mode, the authors first analyze the average power of each position in a residual block and derive the new scan order in the descending order of average power of each position. However, its improvement is very limited, around 1% bit saving compared with [32].

2.3 Matrix Decomposition

Matrix Decomposition is a very powerful tool and widely used in both academic and industrial research. As introduced in Section 1.1, a video is a group of frames and each frame could be understood as a matrix of pixels. Therefore, matrix decomposition is intensively applied for digital video processing. In this section, we only introduce the concepts that are closely related to the methodologies used in this thesis for video

Chapter 2. Literature Survey

coding.

2.3.1 Vector Norms

A vector norm is a total size or length of the vector in a vector space. For simplicity, we can say that the higher the norm is, the bigger the (value in) vector is. Generally, the norm is denoted by: $\|\mathbf{x}\|$, where $\mathbf{x}$ is a vector.

Formally, the Ln-Norm of $\mathbf{x}$ is defined as:

$$\|\mathbf{x}\|_n = \sqrt[n]{\sum_i |x_i|^n} \text{ where } n \in \mathbb{R}$$

A n^{th} root of a summation of all elements' absolute values to the n^{th} power is what we call a Ln-Norm.

The mathematical properties of each norm are very different and thus their applications are dramatically different too. Several commonly used norms are L2-Norm, L1-Norm and L0-Norm. Hereby we are going to look into these three norms in details.

2.3.1.1 Property of L2-Norm based Optimization Problems

The most popular of all norms is the L2-Norm. It is used in almost every field of engineering and science as a whole. Following the basic definition, L2-Norm is defined as

$$\|\mathbf{x}\|_2 = \sqrt{\sum_i |x_i|^2}$$

The L2-Norm is well known as a Euclidean-Norm, which is used as a standard quantity for measuring a vector difference. If the L2-Norm is computed for a vector difference, as shown in (2.6), it is known as a Euclidean distance, where x_1^i and x_2^i are the i^{th} element in vector $\mathbf{x}_1$ and $\mathbf{x}_2$.

$$\|\mathbf{x}_1 - \mathbf{x}_2\|_2 = \sqrt{\sum_i |x_1^i - x_2^i|^2} \quad (2.6)$$

Chapter 2. Literature Survey

In the squared form, it is also known as a Sum of Squared Difference (SSD) among video coding or computer vision scientists:

$$SSD = ||\mathbf{x}_1 - \mathbf{x}_2||_2^2 = \sum_i |x_1^i - x_2^i|^2 \quad (2.7)$$

The Mean Squared Error (MSE) used in (2.1) for PSNR calculation is the mean of SSD. To show the property of L2-Norm optimization, we use one of the simplest L2-Norm minimization problems in (2.8) as an example.

$$\begin{aligned} f(\mathbf{x}) &= \min ||\mathbf{x}||_2 \\ \text{s.t. } \mathbf{Ax} &= \mathbf{b} \end{aligned} \quad (2.8)$$

Assume that the constraint matrix $\mathbf{A}$ has full rank.

By using Lagrange Multipliers[72], we can eliminate the constraint and transform of problem in (2.8) into

$$L(\mathbf{x}) = ||\mathbf{x}||_2^2 + \boldsymbol{\lambda}^T (\mathbf{Ax} - \mathbf{b}) \quad (2.9)$$

where $\boldsymbol{\lambda}$ is the introduced Lagrange multipliers.

Since $||\mathbf{x}||_2^2$ is differentiable according to the definition of the L2-Norm, we can take derivative of this equation equal to zero to find an optimal solution and get

$$\mathbf{x}_{\text{opt}} = -\frac{1}{2} \mathbf{A}^T \boldsymbol{\lambda} \quad (2.10)$$

We plug this solution into the constraint to get

$$-\frac{1}{2} \mathbf{AA}^T \boldsymbol{\lambda} = \mathbf{b} \quad (2.11)$$

$$-\boldsymbol{\lambda} = -2(\mathbf{AA}^T)^{-1} \mathbf{b} \quad (2.12)$$

and finally

$$\mathbf{x}_{\text{opt}} = \mathbf{A}^T (\mathbf{AA}^T)^{-1} \mathbf{b} \quad (2.13)$$

By using this equation, we can now instantly compute an optimal solution of the L2-optimisation problem. This equation is well known as the Moore-Penrose Pseudoinverse and the problem itself (2.8) is usually known as Least Square problem, or Least Square

Chapter 2. Literature Survey

optimization.

One of the keys for solving the L2-Norm optimization problem is that the L2-Norm is differentiable. If the constraints are also differentiable, we can get the solution directly by calculating their derivatives as shown in example (2.8).

2.3.1.2 Property of L1-Norm based Optimization Problems

Following the definition of a vector norm, the L1-Norm of $\mathbf{x}$ is defined as

$$\|\mathbf{x}\|_1 = \sum_i |x_i| \quad (2.14)$$

It has many forms among various fields, e.g. the Manhattan norm[117]. If the L1-Norm is computed for a difference between two vectors or matrices, that is,

$$\text{SAD} = \|\mathbf{x}_1 - \mathbf{x}_2\|_1 = \sum_i |x_1^i - x_2^i| \quad (2.15)$$

It is called Sum of Absolute Difference (SAD) among video coding and computer vision scientists.

The simplest L1-Norm optimization problem could be formulated as

$$\begin{aligned} f(\mathbf{x}) &= \min \|\mathbf{x}\|_1 \\ \text{s.t. } \mathbf{Ax} &= \mathbf{b} \end{aligned} \quad (2.16)$$

Because the L1-Norm doesn't have a smooth function and the Lagrange multipliers based method is much hard to apply, since the object function is no long differentiable. Since it is very difficult to find the solution for this problem explicitly, the usefulness of the L1-Norm optimization was very limited for decades. Until recently, by using many helpful algorithms, namely the Convex Optimization algorithm [72] such as linear programming, or non-linear programming, etc. it's now possible to find the best solution to some L1-norm optimization cases depending on the uniqueness and convexity of the problem. Many applications that rely on the L1-Norm optimization, including the Compressive Sensing [98], are now possible and are under intensive research.

Chapter 2. Literature Survey

2.3.1.3 Property of L0-Norm based Optimization Problems

By definition, the L0-Norm of $\mathbf{x}$ is

$$\|\mathbf{x}\|_0 = \sqrt[0]{\sum_i |x_i|^0} \quad (2.17)$$

Since the definition of 0^{th} root is not clear, engineers often treat the L0-Norm as follows [117]:

$$\|\mathbf{x}\|_0 = \#\{x_i \mid x_i \neq 0\} \quad (2.18)$$

which is a total number of non-zero elements in a vector.

Many applications, including Compressive Sensing, try to minimize the L0-Norm of a vector corresponding to some constraints, hence called “L0-Norm minimization”. One of the simplest L0-Norm minimization problems could be formulated as:

$$\begin{aligned} f(\mathbf{x}) &= \min \|\mathbf{x}\|_0 \\ \text{s.t. } \mathbf{Ax} &= \mathbf{b} \end{aligned} \quad (2.19)$$

However, this is not an easy task. Because the lack of L0-Norm’s mathematical representation, L0-Norm minimization is regarded by computer scientist as an NP-hard problem, which means the problem requires non-polynomial time to be solved and would take thus a lot of time.

In many cases, researchers try to relax the L0-Norm minimization problem to be higher-order norm problem such as L1-Norm minimization and L2-Norm minimization for a possible solution.

In this subsection, we have briefly introduced three members of norm family and their properties relevant to the topics in this thesis. In the next part, we’re going to look at some applications of these norms in matrix decomposition.

Chapter 2. Literature Survey

2.3.2 Concept of Matrix Decomposition

Matrix decomposition is a factorization of a matrix into a set of sub-matrices. There are two major types of matrix decompositions: multiplicative matrix decomposition and additive matrix decomposition.

Multiplicative matrix decomposition is a factorization of a matrix, X , into a product of matrices, Y_i , as:

$$X = Y_1 \cdot Y_2 \cdot Y_3 \dots Y_k \quad (2.20)$$

Additive matrix decomposition is a decomposition of a matrix, X , into a summation of their corresponding matrices, Y_i , as:

$$X = Y_1 + Y_2 + Y_3 \dots Y_k \quad (2.21)$$

Multiplicative and additive matrix decompositions are often combined to model complex problems [73-76]. Given an input matrix, $X \in \mathbb{R}^{m \times m}$, matrix decomposition is often achieved by the following optimization problem

$$f(Y_1, Y_2 \dots Y_k) = c(X, Y_1, Y_2 \dots Y_k) \quad (2.22)$$

where c is the cost function with various definitions, and usually, $f(Y_1, Y_2 \dots Y_k)$ should be small if $Y_1, Y_2 \dots Y_k$ is “good”, which means they yield a small cost. In most situations, there will be some constraints on $Y_1, Y_2 \dots Y_k$.

The following are two famous matrix factorization problems for illustration. Both problems try to decompose X into the form of

$$X = Y_1 \cdot Y_2 + Y_3 \quad (2.23)$$

If we define

$$f(Y_1, Y_2) = \min \|Vec(X - Y_1 \cdot Y_2)\|_1 \quad (2.24)$$

$$s.t. Y_1 \cdot Y_1^T = I_r, r \leq \text{Rank of } X \quad (2.25)$$

Chapter 2. Literature Survey

This is a typical L1-PCA problem [77]. Here, $Vec()$ is a vectorization operator, which turns all the matrix elements to a vector form in the row by row order. Suppose A is any $m \times m$ matrix and after vectorization, we have the vector in (2.26), where each $a_{r,c}$ is an element of A at the position (r,c) .

$$Vec(A)=[a_{1,1},a_{1,2}...a_{1,m},a_{2,1},a_{2,2},...a_{2,m},...a_{m,1},a_{m,2}...a_{m,m}] \quad (2.26)$$

$\|\cdot\|_1$ in (2.24) is the L1-Norm as we defined in (2.14). $\|vec(X - Y_1 \cdot Y_2)\|_1$ is the cost function we want to minimize. $Y_1 \cdot Y_1^T = I_r$ is an orthogonal constraint which means Y_1 is an orthogonal matrix that has lower rank than that of X .

Similarly, if we define the problem as (2.27), this will be a classical non-negative matrix factorization problem [70]:

$$f(Y_1, Y_2) = \min \|vec(X - Y_1 \cdot Y_2)\|_2 \quad (2.27)$$

$$s.t. Y_1 \geq 0, Y_2 \geq 0$$

Here $\|vec(X - Y_1 \cdot Y_2)\|_2$ is the cost function and $Y_1 \geq 0$ and $Y_2 \geq 0$ mean Y_1 and Y_2 only have non negative components, and its solution [78] explores the differentiable property of L2-Norm as we discussed in Subsection 2.3.1.1.

There are many other matrix factorization methods, e.g. QR decomposition, LU decomposition and etc. [73-76] which are not mentioned here and they have made great contributions in areas such as computer vision, pattern recognition and signal processing.

2.3.3 Matrix Decomposition for Video Coding

In the rest of this thesis, we will show how matrix decomposition techniques help to meet the challenges we mentioned in Chapter 1. More specifically, as shown in Figure 2.1, we deal with each issue with one specific matrix decomposition technique. In

Chapter 2. Literature Survey

Chapter 3, we propose a non-motion estimation based temporal redundancy removal methodology with the 2D-SVD technique, which is formulated as a matrix decomposition problem based on the L2-Norm optimization based technique. In Chapter 4, we propose a new transform, namely Rotated Orthogonal Transform (ROT) instead of the DCT for Motion Compensation residual (MC-residual) coding. It is formulated as a matrix decomposition problem based on the L1-Norm optimization. In Chapter 5, we propose a new template and scan order based method, namely Mode Dependent Template (MD-Template) and scan order for the issue of entropy coding in a lossless coding scenario. It is formulated as a matrix decomposition problem based on the L0-Norm optimization technique. The details of these techniques and performance will be discussed in the rest of this thesis.

Chapter 2. Literature Survey

Figure 2.1 Correspondence between issues in video coding and our solutions based on matrix decomposition techniques

Chapter 3

Two-dimensional Singular Value

Decomposition Based Video Coding

As introduced in Subsection 2.2.1, for temporal redundancy reduction, Motion Estimation (ME) is a very important component. However, ME also brings high computation which results in long computational time, high energy consumption and short battery-life. In this section, we propose a low complexity video coding scheme based upon two-dimensional Singular Value Decomposition (2D-SVD), which exploits basic temporal correlation in visual signal without resorting to motion estimation. The proposed scheme provides a better compromise between computational complexity and temporal redundancy reduction, compared with the existing video coding methods. In addition, the problems caused by frame decoding dependency in hybrid video coding, such as unavailability of random access are avoided. The comparison of the proposed 2D-SVD coding scheme with the existing relevant non-ME based low complexity codecs shows its advantages and potential in applications.

3.1 Overview

Hybrid video coding combines the inter-frame prediction and the transformation coding of the prediction residues. H.261/3, MPEG-1/2/4 [1-4] and H.264/AVC [5] video coding standards adopt this coding framework. Temporal redundancy is exploited by the inter-frame prediction techniques such as block-based motion estimation (ME). Consequently, coding efficiency is significantly improved.

However, as discussed in Subsection 2.2.1, the price for the high compression efficiency is the high computational complexity due to the ME process (even when fast ME techniques are used).

It is a challenge to exploit temporal redundancy for coding gain without significant increase in computational complexity. In this chapter, we propose a new video coding scheme based on two-dimensional Singular Value Decomposition (2D-SVD). The 2D-SVD belongs to the category of matrix decomposition [79-83] and has been studied intensively for computer vision during the recent years. In the perspective of its use in video coding, the proposed 2D-SVD coding combines the advantages of Singular Value Decomposition (i.e., one-dimensional SVD, or 1D-SVD for short) and Discrete Cosine Transformation (DCT) [8]. For block based 1D-SVD coding [84][85], although the coefficient matrix is diagonalized after 1D-SVD, this requires the transmission of two eigenvector matrices for each block. These two matrices incur very high overhead which often leads to low coding efficiency, and therefore block based 1D-SVD video coding is not commonly used. The proposed 2D-SVD scheme needs to transmit only two eigenvector matrices to reconstruct a group of frames/blocks, instead of two transformation matrices for every frame/block; compared with the commonly used DCT

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

(non-ME) based coding, the energy compaction of the coefficient matrices in 2D-SVD is better, as demonstrated in this work. Compared with ME based hybrid codecs, the computational complexity of our coding scheme is significantly reduced since no ME is adopted.

The rest of the chapter is organized as follows: Section 3.2 gives the theoretical foundation of the 2D-SVD, the comparison with DCT and block based 1D-SVD, and the associated procedures for 2D-SVD based video decomposition that will be used in this work; Section 3.3 introduces the proposed 2D-SVD coding principles with comparison with block based 1D-SVD and DCT; in Section 3.4, the whole codec framework is discussed; Section 3.5 presents the experimental results and comparison with the existing relevant codecs in terms of both coding performance and computational complexity, while Section 3.6 concludes this chapter.

3.2 2D-SVD for Video Decomposition

3.2.1 2D-SVD Fundamental

An image or a video frame can be divided into $m \times m$ non-overlapping blocks. The methodology to be discussed in this section can be applied to a frame or a block (we discuss frame-based transform in this subsection, but the results can be used for blocks).

If the i^{th} frame is denoted as X_i , it can be transformed as:

$$M_i = Y_l^T X_i Y_r \quad (3.1)$$

where M_i is the transformed coefficient matrix, and Y_l and Y_r are the left and right real-valued unitary transform matrices, respectively, whose exact forms depend on the transformation used. Such a transformation is reversible because the recovered frame can

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

be obtained as follows:

$$X_i = Y_l M_i Y_r^T \quad (3.2)$$

In the case of SVD (i.e., 1D-SVD), a frame is decomposed into two eigenvector matrices, Y_l and Y_r , and a diagonal matrix M_i . The decomposition is illustrated in Figure 3.1.

The main problem for 1D-SVD coding is that, although its coefficient matrix contains the fewest non-zero coefficients compared with other transforms, its transformation matrices Y_l and Y_r are not fixed and hence need to be sent to the decoder for each frame. As a result, the overall coding efficiency for the 1D-SVD coding is not very promising [84][85].

Figure 3.1 SVD decomposition

Figure 3.2 DCT decomposition

Figure 3.3 2D-SVD decomposition

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

For DCT [8], one fixed matrix is used, and thus $Y_l = Y_r$. The DCT decomposition is shown in Figure 3.2. Since the transformation matrix is fixed, we can simply represent X_i with M_i (i.e., we only need to code M_i for transmission).

The 2D-SVD decomposition is shown in Figure 3.3. In the case of 2D-SVD [81], two transformation matrices Y_l and Y_r are extracted from a group of frames $\{X_1, \dots, X_n\}$ along the time axis and based on optimization function in (3.3), the resultant transform coefficient group $\{M_1, \dots, M_n\}$ yields optimal energy compaction for any given group of frames in terms of Mean Squared Error (MSE).

Compared with most 3D wavelet and 3D-DCT based codecs, 2D-SVD based approach has less potential for temporal redundancy reduction. This is because after we get the 2D-SVD transform matrices, the transform is limited in spatial domain, while for 3D wavelet and 3D-DCT based approaches, transforms are applied in both spatial and temporal domain. On the other hand, 2D-SVD based coding provides better frame independency at the decoder side since transform is limited in spatial domain. Compared with block based 1D-SVD, we only need to transmit two transformation matrices for one group of frames for reconstruction, whereas 1D-SVD has to transmit two transformation matrices for each frame in the group. Compared with DCT, the energy compaction of the group of coefficient matrices is optimized for 2D-SVD in the least square sense. This is because the transformation matrices derived by 2D-SVD is adaptive to the input group of frames while DCT uses the predefined transformation matrix. More comparison in this regard will be further demonstrated in Section 3.3.

The 2D-SVD process is formulated as the following Simultaneous Low Rank Approximation of Matrices (SLRAM) problem [83]:

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

$$\begin{aligned} f(Y_l, M_i, Y_r) &= \min_{Y_l, Y_r, M_i} \sum_{i=1}^n \|X_i - Y_l M_i Y_r^T\|_F^2 \\ s.t \quad & Y_l^T \cdot Y_l = I_{l1}, \quad Y_r^T \cdot Y_r = I_{l2} \end{aligned} \quad (3.3)$$

where $X_i \in \mathbb{R}^{H \times W}$ being the i^{th} video frame and we aim to compute two matrices $Y_l \in \mathbb{R}^{H \times l1}$, $Y_r \in \mathbb{R}^{W \times l2}$, $I_{l1} \in \mathbb{R}^{l1 \times l1}$, $I_{l2} \in \mathbb{R}^{l2 \times l2}$, $M_i \in \mathbb{R}^{l1 \times l2}$, $l1 \leq H$ and $l2 \leq W$, such that for $i = 1, 2, \dots, n$, $\sum_{i=1}^n \|X_i - Y_l M_i Y_r^T\|_F^2$ is minimized, where $\|\cdot\|_F$ denotes the Frobenius norm (F-Norm) of a matrix. According to the definition of F-Norm,

$$\sum_{i=1}^n \|X_i - Y_l M_i Y_r^T\|_F^2 = \sum_{i=1}^n \|Vec(X_i - Y_l M_i Y_r^T)\|_2^2 \quad (3.4)$$

Here, $Vec()$ is the vectorization operator as we defined in (2.26).

Then, the objective function in (3.3) turns to be the L2-Norm optimization problem we discussed in Subsection 2.3.1.1 and it is differentiable.

Therefore, taking derivative of $f(Y_l, M_i, Y_r)$ with respect to M_i and let it equate to 0, we obtain: $\partial f / \partial M_i = M_i^T - Y_r^T X_i Y_l = 0$, and thus

$$M_i = Y_l^T X_i Y_r \quad (3.5)$$

Substituting (3.5) into (3.4) we have (please refer to [83] for the detail of formula derivation)

$$\sum_{i=1}^n \|X_i - Y_l M_i Y_r^T\|_F^2 = \sum_{i=1}^n \|X_i\|_F^2 - \sum_{i=1}^n \|Y_l^T X_i Y_r\|_F^2 \quad (3.6)$$

Thus, the minimization in (3.3) is equivalent to:

$$\begin{aligned} f(Y_l, Y_r) &= \max_{Y_l, Y_r} \sum_{i=1}^n \|Y_l^T X_i Y_r\|_F^2 \\ s.t \quad & Y_l^T \cdot Y_l = I_{l1}, \quad Y_r^T \cdot Y_r = I_{l2} \end{aligned} \quad (3.7)$$

The optimal solution, Y_l^{opt} and Y_r^{opt} , of (3.7) is also the solution of (3.3), and $M_i =$

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

$Y_l^{\text{opt}T} X_i Y_r^{\text{opt}}$. More details of the associated analysis are available in [83].

3.2.2 2D-SVD for Video Decomposition

However, there is no explicit solution for (3.7) [81]. Therefore, a near-optimal solution for such a low rank approximation problem is given in [81]. Based upon the concepts presented in Subsection 3.2.1 and the solution in [81], we formulate the 2D-SVD process for each Group of Blocks (GOB) as follows.

- 1) Given a Group of Pictures (GOP) with n frames, we denote the i^{th} frame as X_i .

The mean frame of the n frames, X_{mean} , is subtracted from each frame and consequently $X_i' = X_i - X_{\text{mean}}$.

- 2) For any Group of Blocks (GOB), B , in the mean-subtracted frames $X_1' \dots X_n'$, we denote $b_1, \dots, b_n$ as each block in B , and the row-row and column-column covariance matrices are defined as

$$F = \sum_{i=1}^n b_i b_i^T$$

$$G = \sum_{i=1}^n b_i^T b_i.$$

Then Y_l and Y_r are made up by k principal eigenvectors of F and s principal eigenvectors of G , respectively, i.e.,

$$F = \sum_{p=1}^r \lambda_p u_p u_p^T \quad Y_l \equiv (u_1, \dots, u_k)$$

$$G = \sum_{q=1}^c \zeta_q v_q v_q^T \quad Y_r \equiv (v_1, \dots, v_s)$$

where $k \leq r$ is the number of eigenvectors in Y_l and $s \leq c$ is the number of eigenvectors in Y_r . Here r and c represent the size of the block.

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

- 3) $M_i = Y_l^T b_i Y_r$ and $b_i' = Y_l M_i Y_r^T$. We calculate the near optimal approximation of each block b_i as $\bar{b}_i = b_i' + b_{mean}$, where b_{mean} is the corresponding mean block in the mean frame X_{mean} .

According to the experiment in [83], the minimum squared error is achieved when $s/k \approx 1$; thus, in the following analysis we set $s = k$.

The details of the proposed 2D-SVD coding scheme are described in Section 3.4. After 2D-SVD, the common features of the video block along the time axis are represented by Y_l and Y_r , which need to be transmitted only once for a GOB; the changing features for every block with time are captured by M_i which contains few high magnitude coefficients.

3.3 Efficiency of 2D-SVD Coding

In the block based 1D-SVD coding, each frame of a video is divided into several blocks and each block is coded with 1D-SVD [84][85]. Suppose that we want to compress a video with a GOP of n frames with the frame resolution of $W \times H$, and the block size is $m \times m$. For 1D-SVD, we have $(W \times H)/(m \times m)$ blocks in each frame. After 1D-SVD, each block is factorized into two transform matrices (contain two $m \times m$ coefficients) and one diagonal coefficient matrix (contains m non-zero coefficients). Therefore, for reconstruction, we need to transmit $2m^2 + m$ coefficients for each block, and for a GOP of n frames with the frame resolution of $W \times H$, the number of coefficients we need to transmit with 1D-SVD is $n \times ((W \times H)/(m \times m)) \times (2m^2 + m) = nWH(2m^2 + m)/m^2$. Similarly, for 2D-SVD, we have $(W \times H)/(m \times m)$ GOBs to compress, and each GOB is factorized into two transform matrices (contain two $m \times m$ coefficients) and n coefficient matrices (each contains $m \times m$ non-zero coefficients). Therefore, the total number of

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

coefficients for a GOB to transmit is $(2m^2 + nm^2)$ and for a GOP of n frames with the frame resolution of $W \times H$, the number of coefficients we need to transmit with 2D-SVD based approach is $((W \times H)/(m \times m)) \times (2m^2 + nm^2) = (2+n)WH$. As shown in Table 3.1, with the GOP size being 24 (details about GOP size selection will be discussed in Section 3.4), 2D-SVD needs only about 50% of coefficients compared with 1D-SVD at various resolutions and block sizes.

Table 3.1 Number of coefficients for 1D-SVD and 2D-SVD

Block Size	Method	Video Resolution			
		QCIF	CIF	2CIF	4CIF
8×8	1D-SVD	1.34×10^6	5.38×10^6	1.07×10^7	2.15×10^7
	2D-SVD	6.84×10^5	2.73×10^6	5.47×10^6	1.09×10^7
16×16	1D-SVD	1.30×10^6	5.23×10^6	1.04×10^7	2.09×10^7
	2D-SVD	6.84×10^5	2.73×10^6	5.47×10^6	1.09×10^7
Coefficient ratio of 2D-SVD over 1D-SVD				8×8	48%
				16×16	49%

(a) DCT Coefficients Distribution

(b) 2D-SVD Coefficients Distribution (with GOP 24)

Figure 3.4 Coefficient distribution comparison for DCT and 2D-SVD

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Table 3.2 Average numbers of zero DCT and 2D-SVD coefficients at same quantization steps (the figures in bold represent the better cases for Zeros Per Block and MSE)

	Sequences	$QS = 0.1$				$QS = 0.5$				$QS = 1.0$			
		Zeros Per Block		MSE		Zeros Per Block		MSE		Zeros Per Block		MSE	
		DCT	2D-SVD	DCT	2D-SVD	DCT	2D-SVD	DCT	2D-SVD	DCT	2D-SVD	DCT	2D-SVD
CIF	Container	33.9	39.9	1.9	1.4	49.3	52.6	16.9	7.3	53.2	56.8	36.6	13.7
	Coastguard	27.7	29.4	2.5	2.3	45.4	46.5	23.0	18.4	51.5	52.2	48.9	34.9
	Waterfall	29.4	31.9	2.1	1.9	45.3	47.5	17.1	13.9	50.7	52.5	33.6	26.0
	Silent	31.7	42.5	2.6	1.5	48.8	54.1	17.3	7.1	53.1	57.1	29.1	11.6
	Hall	37.8	41.8	1.5	1.3	51.3	55.1	9.9	4.6	54.0	58.4	20.0	7.9
	Paris	28.3	39.7	2.5	1.6	44.1	52.8	27.2	10.3	49.3	55.5	60.3	20.7
QCIF	Claire	45.5	48.9	0.8	0.7	54.7	56.8	8.7	3.4	56.0	58.6	14.6	6.2
	Salesman	28.3	43.9	2.5	1.3	45.1	53.7	21.4	7.6	51.1	56.6	41.2	14.5
	News	34.1	46.8	1.9	0.9	47.6	55.7	18.5	5.2	51.3	57.4	37.9	10.2
	Akiyo	40.8	47.1	1.2	0.7	51.4	56.3	9.1	3.8	54.3	58.2	18.2	7.3
	Hall	34.9	44.5	1.9	1.1	48.7	55.6	17.1	4.2	52.5	58.1	37.8	7.9
	Grandma	34.6	46.7	2.1	1.0	50.0	57.2	14.6	4.3	54.1	58.5	25.6	7.3

In comparison with DCT, the energy compaction of the group of coefficient matrices is optimized for 2D-SVD in the least square sense. In Figure 3.4, we apply both 2D-SVD and DCT on the *carphone* video sequence, and show the resultant coefficient matrices of the 150th frame (with GOP size of 24 for 2D-SVD) after the same quantization.

As demonstrated in Figure 3.4, the coefficient matrix of 2D-SVD has fewer high magnitude coefficients (darker spots in the figure) compared with that of DCT.

We further compare the numbers of zero DCT and 2D-SVD coefficients at same quantization steps in Table 3.2. We use the quantization matrix used in JPEG standard as (3.8) [115], which is optimized for the DCT coefficient distribution. We quantize DCT and 2D-SVD coefficients with the quantization table derived by $QS \times Q_{Table}$ for comparison, where QS is the Quantization Step set as 0.1, 0.5 and 1.0. The average number of zero coefficients of an 8×8 block is calculated based on the transformed blocks of 120 frames for each sequence (with GOP size of 24 for 2D-SVD). We also

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

calculate the reconstruction error in terms of Mean Squared Error (MSE) after quantization for the coefficient matrices of these two transforms.

$$Q_{Table} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \quad (3.8)$$

In Table 3.2, we observe that with different quantization steps, 2D-SVD coefficients have more zero coefficients with less reconstruction error. For example, for QCIF Salesman sequence, with $QS=0.1$, 2D-SVD coefficients contains 15 more zero coefficients in each block on average, while the MSE is only half of that of DCT coefficients. Although more zero coefficients do not guarantee higher coding efficiency (this is the reason for our subsequent analysis) since the distribution of the zero coefficients also impacts the efficiency, more zero coefficients and less MSE still represent a remarkable potential for higher coding efficiency.

Table 3.3 Coding Gain comparison of the DCT and the proposed 2D-SVD

	Sequences	Coding Gain (dB)		Coding Gain Improvement (dB)
		DCT	2D-SVD	
CIF	Container	16.88	22.52	5.64
	Coastguard	19.24	23.71	4.47
	Waterfall	16.96	19.56	2.60
	Silent	18.68	21.76	3.08
	Hall	13.92	23.32	9.40
	Paris	11.28	17.98	6.70
QCIF	Claire	19.76	24.74	4.98
	Salesman	11.73	18.41	6.68
	News	12.30	19.44	7.14
	Akiyo	17.11	23.72	6.61
	Hall	13.82	22.19	8.37
	Gramdma	13.41	22.54	8.13
<i>Average Coding Gain Improvement (dB)</i>				<i>6.15</i>

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

We also show the advantage of 2D-SVD in terms of coding gain [9-12] (as defined in (2.1) of Subsection 2.1.1) against DCT, since most of the mainstream video codecs are DCT based. For coding gain comparison, we computer the DCT and 2D-SVD based on the 8×8 blocks with GOP size of 24 frames. Table 3.3 shows the comparison results. As observed, 2D-SVD gives significant coding gain improvement (6.15 dB on average) compared with the DCT due to its optimized energy compaction.

3.4 Framework of 2D-SVD Coding

3.4.1 2D-SVD Encoder

Following the diagram of the proposed 2D-SVD encoder depicted in Figure 3.5 and the procedures in Subsection 3.2.2, we discuss each part of the encoder in more details below.

First we extract the mean frame from the GOP with n frames, and compress it with JPEG standard with a high quality factor (quality factor of 90 has been used in the current work). Then, we divide the mean-subtracted GOP into 8×8 GOBs. For the j^{th} GOB B_j , we conduct 2D-SVD to obtain the corresponding eigenvector matrices Y_l^j and Y_r^j and the group of 8×8 coefficient matrices $M_1^j \dots M_n^j$. Here, each of Y_l^j and Y_r^j is an eigenvector matrix, which contains eight eigenvectors as $Y_l^j = [u_1^j, u_2^j, \dots, u_8^j]$ and, $Y_r^j = [v_1^j, v_2^j, \dots, v_8^j]$. However, the importance of these eigenvectors varies. Let $m_i^j(x, y)$ denote the coefficient value at position (x, y) of M_i^j , we can reconstruct the block b_i^j by

$$b_i^j = \sum_{x=1}^n \sum_{y=1}^n u_x^j m_i^j(x, y) v_y^{jT} \quad (3.10)$$

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

The eigenvectors with small sub-indices, e.g. $u_1^j, u_2^j, v_1^j, v_2^j$, are more important since they correspond to the top left coefficients of coefficient matrices $M_1^j \dots M_n^j$. The top left coefficients contain more energy, and thus the same distortion in u_1^j, u_8^j will result in a different ultimate reconstruction quality: if $m_i^j(1,1)$ is thousands of times of $m_i^j(8,8)$, the distortion in u_1^j is amplified by the same factor compared with the same distortion in u_8^j . Vector Quantization (VQ) is often used in coding the eigenvectors [84],[85]. Here we adopt the VQ strategy in [85]. For the 8 eigenvectors, $[u_1^j, u_2^j, \dots, u_8^j]$, we code them with codebooks of length 256, 256, 128, 128, 128, 64, 64 and 32 respectively due to their different importance. These codebooks are also used for $[v_1^j, v_2^j, \dots, v_8^j]$. We learn these codebooks using LBG algorithm [86] from the training sets of eigenvectors derived by computing the 8×8 2D-SVD of 6 video sequences [84]. The training sequences are QCIF *Carphone*, *Grandma*, *Silent*, and *Suzie*, each with 192 frames and CIF *Hall* and *News*, each with 192 frames. They are stored in both the encoder and the decoder. In this way, we are able to transmit the eigenvectors with very few bits. Table 3.4 lists the bits required for the transmission of the eigenvectors.

From Table 3.4, we observe that the number of additional bits required is very small, and with GOP 24, we only need at most 0.07 bpp to transmit the eigenvectors.

We quantize the group of coefficient matrices, $M_1^j \dots M_n^j$, with the 8×8 JPEG quantization table in (3.8), accounting for the fact that the coefficients with most energy are located around the top left corner, while those around the right bottom part is much smaller (i.e., with much less energy), which is similar to the distribution of DCT coefficients.

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Figure 3.5 The proposed 2D-SVD encoder

Table 3.4 Bits (bpp--bits per pixel) required for eigenvectors Y_l^j, Y_r^j for a GOB

		Number of Eigenvectors to transmit							
		N = 1	N = 2	N = 3	N = 4	N = 5	N = 6	N = 7	N = 8
GOP	24	0.010	0.021	0.030	0.039	0.048	0.056	0.064	0.070
	48	0.005	0.010	0.015	0.019	0.024	0.028	0.032	0.035

After quantization, we use 6 bits to indicate how many eigenvectors to transmit for each GOB. Suppose now $m_i^j(x, y)$ represents the non-zero coefficients after quantization and (x, y) is its position. Then let $x_{max}(i) = \max(x)$, $y_{max}(i) = \max(y)$, which denotes the maximum position of non-zero coefficients in quantized coefficient matrix M_i^j . For the GOB with size n , the maximum position is $R_{max} = \max(x_{max}(i), i = 1, 2, \dots, n)$ and $C_{max} = \max(y_{max}(i), i = 1, 2, \dots, n)$. Then for each GOB we represent R_{max} and C_{max} with 3 bits respectively, since the block size is 8×8 . After we have R_{max} and C_{max} , we only need to transmit $[u_1^j, u_2^j, \dots, u_{R_{max}}^j]$ and $[v_1^j, v_2^j, \dots, v_{C_{max}}^j]$ for each GOB B_j . For most cases, R_{max} and C_{max} are less than 4.

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Figure 3.6 2D-SVD coefficients after 2D-SVD with different GOPs

After compressing the mean frame and eigenvectors, we transmit them as Group Information (GI) for the whole GOP. For most cases, the GI occupies less than 5% of the overall GOP bitstream with GOP size being 24.

Finally, by adopting the Zig-Zag Scan [1][2], Run Length Coding and Variable Length Coding in JPEG, the coefficient matrix of each frame is further compressed.

Within a stable scene, we assume the size of GI is fixed since GI only contains a compressed mean frame and a group of eigenvectors regardless the GOP size. Under this assumption, with longer GOP, the GI will occupy smaller percentage of the bitstream, since every GOP only contains one GI. However, we cannot form a GOP as long as we

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

want if we consider the transmission delay and memory requirement. With the increase of GOP size, there will be more high-magnitude 2D-SVD coefficients, which negatively impact the coding efficiency, as observed in Figure 3.6, where we compare the 2D-SVD coefficients of 150th frame of *carphone* sequence after 2D-SVD with GOP size of 8, 24 and 48. As we can see, when GOP size is 8, there are a few high magnitude coefficients located close to the diagonal, while with GOP 48, there are many more high magnitude coefficients. Although we show coefficients comparison in a frame level, this is also the case in block level. Based on the analysis above, we need to choose a rational GOP size, such that we balance the portion of GI in each GOP and the number of high magnitude coefficients. In Figure 3.7 we show the coding result at GOP 8, 16, 24 and 48 for *coastguard* sequence. From GOP 8 to GOP 24, the coding efficiency keeps increasing, and when GOP further increases to 48, the coding efficiency turns to decrease. For the other tested sequences, the optimal GOP size also ranges from 21 to 31.

The 2D-SVD decomposition process is memory-consuming compared with existing frame-by-frame techniques. Moreover, there is an initial delay (GOP-1=23 frames here) in our scheme. Therefore, our scheme may not be suitable for very low delay applications. However, the existing video coding standards and many schemes also have such delay (e.g., as analyzed in [87], decoding can be only completed after transmitting a GOP). The GI is crucial for GOP reconstruction. If the GI is destroyed, the whole GOP is useless. However, this is also the case for GOP hierarchy of IPPP..P structure that most algorithms use: if the I-frame is lost, the GOP is also lost. Therefore, error resilience techniques (e.g. Forward Error Correction (FEC), data partitioning, retransmission) for the existing key frames (e.g. I-frame in IPPP or IBBP structure) could also be used for our scheme to achieve error resilience.

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Figure 3.7 2D-SVD coding result comparison at GOP 8, 16, 24 and 48

Ideally, we should design an adaptive GOP size such that the optimal coding efficiency is achieved for each GOP; however, due to low-complexity consideration, we fix the GOP size to 24 in this work.

3.4.2 2D-SVD Decoder

The decoder part, as illustrated in Figure 3.8, basically carries out the reverse steps of the encoder. First, the GI is decoded; the mean frame and eigenvector matrices are extracted. Then we get the reconstructed frames without the mean frame by inverse-SVD as $\overline{X}_t' = \bigcup_{j=1}^N Y_t^j M_t^j Y_r^{jT}$, where N is the number of 8×8 blocks in each frame. Finally, by adding the reconstructed mean frame, we obtain the reconstructed frame $\overline{X}_t$.

In this way, once the GI, whose size is small, is successfully received, we are able to decode each frame independently just like an intra frame video codec. This inter-frame independency is a very desirable property since the problems such as error propagation and difficulties in random frame access are all due to the inter-frame dependency. A lot of work in the literature has been done to deal with these problems by introducing complex codec structure or transmission protocol [88-90].

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Figure 3.8 The 2D-SVD decoder

3.5 Experimental Results

This part evaluates the performance of the proposed 2D-SVD video codec in comparison with other low complexity codecs which do not adopt ME either. Result from one motion estimation based method, H.264 Inter, is also added to show the gap between motion estimation based approach and the proposed 2D-SVD approach. 14 testing sequences are selected (four QCIF (144x176) sequences, four CIF (288x352) sequences, four 4CIF (576x704) sequences and two 1080P (1080x1920) sequences, as listed in the first column of Table 3.5), with frame rate at 30 FPS are used for test. Each sequence contains 180 frames.

For the proposed 2D-SVD based approach, the mean frame is compressed by JPEG with quality factor of 90, the GOP size and GOB size are both 24 and the block size is 8x8. The following video codecs have been used as benchmarks to evaluate the performance of 2D-SVD approach.

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

1) The H.264/AVC Intra — Coding with JM16.0 [95] in main profile without exploiting temporal redundancy (i.e., Intra mode is selected); it is considered as one of the most efficient Intra coding approach.

2) H.264/AVC No-motion [97] — This profile is often used as a benchmark in comparison for non-ME based low complexity video codecs [97]. We use JM16.0 [95] main profile to code the GOP of 24 frames with the first frame coded as I-frame and all other frames coded as predictive frames. The full Rate-Distortion Optimization (RDO) is enabled. The loop filter is enabled. Entropy coding is performed in CABAC mode and the ME search range is set to 0.

3) MJPEG-DPCM Video Codec [17] — In this codec, differential encoding between DCT coefficients of adjacent frames is carried out. The encoder and decoder are both based on JPEG codec.

4) H.264/AVC Inter — Coding using JM16.0 [95] in main profile, with quarter-pixel motion compensation, Enhanced Predictive Zonal Search (EPZS)[124] based fast motion estimation, full Rate-Distortion Optimization (RDO), 24 frames intra period, 7 B-frames inserted between I-frame and P-frame and the entropy coding is performed in CABAC mode. The ME search range is 7 for QCIF and CIF sequences and 16 for 4CIF and 1080P sequences.

The first three codecs represent three low complexity non-ME based codecs mentioned in the Section 3.1. H.264/AVC Inter coding result is also recorded in the comparison to show the gap between motion-estimation based approach and the proposed 2D-SVD approach.

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

3.5.1 Objective Coding Results Comparison

Figure 3.9 R-D curves of 4 low complexity codecs

For the four coding approaches without motion estimation, the R-D curves of four video sequences are plotted in Figure 3.9. In Figure 3.9, H.264/AVC Intra gives the lowest coding efficiency, while 2D-SVD is the most efficient low complexity codec. This is mainly due to the fact that H.264/AVC Intra does not explore any temporal redundancy, while the other three low complexity codecs explore the temporal redundancy without resorting to ME. Among all these four coding methods, the H.264 Intra, MJPEG-DPCM and H.264 No-Motion are DCT based.

A closer observation on Figure 3.9 indicates that compared with H.264 No-motion, the best existing DCT based non-ME codec in our comparison, the proposed 2D-SVD coding method gives higher PSNR improvement at low bitrates compared with high bitrates. This result is consistent with our previous analysis in Table 3.2. Take the data of

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Claire in Table 3.2 for instance. Usually, a similar number of non-zero coefficients leads to similar bitrate (without considering the distribution of coefficients). When $QS = 0.1$ (corresponding to a high bitrate), 2D-SVD has 15.1 (= 64-48.9) non-zero coefficients per block on average, while DCT has 18.5 (= 64-45.5) non-zero coefficients on average; we can expect their coding to have similar bitrates. In this situation, their reconstruction error (MSE) is very close, 0.7 for 2D-SVD and 0.8 for DCT (i.e., not much PSNR gain for 2D-SVD). Still for *Claire* in Table 3.2, when $QS = 1.0$ (corresponding to a low bitrate), both 2D-SVD and DCT still have similar numbers of non-zero coefficients: 5.4 (= 64-58.6) for 2D-SVD and 8 (= 64-56) for DCT. However, 2D-SVD has MSE of 6.2, less than a half compared with that of DCT (=15.6). Therefore, with less non-zero coefficients (e.g. $QS = 1.0$, or low bitrate), the 2D-SVD coefficients contain more energy than those of DCT, and thus the proposed scheme gives higher coding efficiency improvement at low bitrates.

We further calculate the average PSNR of the R-D performance for these testing sequences. By taking the R-D curve of other reference approaches as the anchor, we also calculate the Bjontegaard-Delta (BD)-Bitrate [36] as introduced in Subsection 2.1.3. It is worth noting that in our comparison, the negative number of BD-Bitrate means the BD-Bitrate saving (positive number of BD-Bitrate means BD-Bitrate loss) of the proposed scheme compared with other approaches. The average performance comparison on these 14 sequences is summarized in Table 3.5.

As we can observe in Table 3.5, compared with the other three non-ME based approaches, H.264 No-Motion, MJPEG-DPCM and H.264 Intra, the proposed 2D-SVD approach achieves the best performance in terms of BD-Bitrate saving, although the saving compared with H.264 No-Motion is minor. On the other hand, compared with the

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

result of H.264 Inter with optimized configuration, the bitrate loss of proposed 2D-SVD approach is huge, roughly 130%, since motion estimation is not used.

Table 3.5 Objective coding efficiency comparison

Sequences		Coding Methods				
		2D-SVD	H.264 No Motion	MJPEG DPCM[17]	H.264 Intra	H.264 Inter
Akiyo (QCIF)	Ave. PSNR(dB)	43.43	42.02	37.06	35.49	47.31
	BD-Bitrate(%)	-	-10.5	-37.8	-70.8	73.0
Claire (QCIF)	Ave. PSNR(dB)	42.73	41.29	39.36	37.71	47.53
	BD-Bitrate(%)	-	-21.8	-33.0	-44.2	83.3
Coastguard (QCIF)	Ave. PSNR(dB)	32.93	32.62	28.61	31.33	39.23
	BD-Bitrate(%)	-	-2.4	-38.5	-7.2	137.3
Container (QCIF)	Ave. PSNR(dB)	43.93	42.56	41.80	34.77	47.98
	BD-Bitrate(%)	-	-8.5	-18.5	-57.5	91.5
Mobile (CIF)	Ave. PSNR(dB)	28.09	27.33	25.44	26.19	34.33
	BD-Bitrate(%)	-	-10.0	-32.4	-19.1	169.4
Paris (CIF)	Ave. PSNR(dB)	36.39	36.54	31.62	28.89	40.72
	BD-Bitrate(%)	-	-2.4	-46.1	-62.5	80.6
Silent (CIF)	Ave. PSNR(dB)	35.09	33.73	31.31	28.99	38.71
	BD-Bitrate(%)	-	-7.7	-42.5	-62.6	64.0
Waterfall (CIF)	Ave. PSNR(dB)	37.93	36.29	32.80	29.01	42.51
	BD-Bitrate(%)	-	-6.9	-47.3	-72.5	125.3
City (4CIF)	Ave. PSNR(dB)	32.96	33.23	28.03	31.35	39.08
	BD-Bitrate(%)	-	4.7	-32.8	-18.2	163.9
Crew (4CIF)	Ave. PSNR(dB)	35.38	36.03	29.67	34.79	43.32
	BD-Bitrate(%)	-	7.3	-39.3	-4.1	204.0
Harbor (4CIF)	Ave. PSNR(dB)	35.36	34.12	31.65	30.85	39.27
	BD-Bitrate(%)	-	-13.4	-45.1	-43.5	80.5
Ice (4CIF)	Ave. PSNR(dB)	40.27	39.21	37.31	38.14	45.63
	BD-Bitrate(%)	-	-11.7	-35.5	-27.6	107.7
SunFlower (1080P)	Ave. PSNR(dB)	32.43	32.91	30.55	28.40	38.71
	BD-Bitrate(%)		9.6	-31.0	-44.8	196.3
ParkScene (1080P)	Ave. PSNR(dB)	31.72	32.08	29.24	27.62	39.90
	BD-Bitrate(%)		7.0	-27.7	-32.5	228.4
Average BD-Bitrate(%)			-4.8	-36.5	-42.6	129.3

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

3.5.2 Visual Quality Comparison

In Figure 3.10, we show the coding visual result for subjective viewing comparison. For *Akiyo* at 30 FPS and around 125kbps, the 2D-SVD and H.264/AVC No-Motion coding results are very at a glance. Close observation shows that the 2D-SVD coding result preserves the detailed texture well while the H.264/AVC No-Motion result loses the details as shown in the amplified parts in Figure 3.10 (a), (b) and (c). This is because that for H.264/AVC No-motion coding, in-loop filters are used for de-blocking. However, at low bitrates, the de-blocking process also blurs the texture and other visual details, while for the result of the proposed 2D-SVD method, the decoded frame keeps the details well without obvious distortion (the PSNR is also the highest in Figure 3.10 under similar bitrates).

Figure 3.10 Subjective coding result comparison of the 90th frame of QCIF *Akiyo*

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

Table 3.6 Subjective coding efficiency comparison with Mean Opinion Score (MOS)

Resolution	Sequences	Bitrate	MOS				
			Proposed (2DSVD)	H.264 No Motion	MJPEG DPCM[17]	H.264 Intra	H.264 Inter
QCIF	Akiyo	100kbps	8.5	8.4	7.2	6.5	9.2
	Claire	100kbps	7.9	7.9	7.0	6.3	9.5
CIF	Container	250kbps	7.9	7.7	6.8	6.2	8.5
	Silent	250kbps	7.2	7.3	6.4	6.0	8.6
4CIF	Harbor	1500kbps	5.3	5.6	4.5	3.4	7.9
	Ice	1500kbps	6.0	6.2	5.1	3.8	8.1
1080P	SunFlower	3000kbps	4.9	5.5	4.3	3.6	7.2
	ParkScene	3000kbps	4.7	5.2	4.1	3.4	7.1
<i>Average MOS</i>			6.5	6.7	5.7	4.9	8.3

To give a comprehensive evaluation on the subjective perceptual quality, we worked with 14 unpaid voluntary test persons. The tests were performed according to [128]. Eight sequences are compressed at similar bitrates as indicated in Table 3.6. The structure of the Basic Test Cell consists of two consecutive presentations of the sequence under test. First the uncompressed version of the video sequence is displayed, followed by the decoded sequence. Then a message is shown for 5 seconds asking the viewers to vote. The presentation of the video clips is preceded by a mid-level gray screen for a duration of one second. For each sequence, we go through the voting of all the coding approaches before we start to vote the next sequence. Test subjects were asked to evaluate the overall quality with Mean Opinion Score (MOS) range from 0 to 10 (10 denotes a quality of reproduction that is perfectly faithful to the original, 9 denotes a “Excellent” quality, 7 denotes a “Good” quality, 5 denotes a “Fair” quality, 3 denotes a “Poor” quality, 1 denotes a “Bad” quality while 0 denotes a quality of reproduction that has no similarity to the original). We show the MOS value of eight test sequences in Table 3.6. As we can observe, the MOS of H.264 Inter is always much higher than other non-ME based methods. Moreover, for high resolution video at “fair” reconstruction

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

quality, e.g. the two 1080P sequences, it is also noticed that the blockiness artifact generated by 2D-SVD approach is more annoying compared with the coding results generated by H.264 No-Motion, which are filtered by de-blocking filter.

3.5.3 Computational Complexity

In our experiment, 2D-SVD is implemented in Matlab, while H.264 No-Motion, H.264 Inter and H.264 Intra are implemented with C/C++. Therefore, it is difficult to compare their computational complexities in the same platform directly. However, we can conduct indirect comparison as follows.

Based on the codec framework shown in Section 3.4, the framework of 2D-SVD based approach is very similar to that of Motion-JPEG. This is because after we derive the coefficients transformed by 2D-SVD, we just treat them as DCT coefficients in Motion-JPEG. Therefore, we use Motion-JPEG as a bridge (or reference) for complexity comparison between two platforms.

Under Matlab platform, for 2D-SVD based approach, we record the time required before coding the 2D-SVD coefficients. This includes the time of mean frame extraction, 2D-SVD decomposition, mean frame compression and eigenvectors compression. This time is subtracted from the overall 2D-SVD encoding time to calculate the time required for 2D-SVD coefficient compression. After we derive the 2D-SVD coefficients, the computation complexity of rest steps is similar to that of Motion-JPEG as confirmed by Row 2 and 3 data for coding time in Tables 3.7 and 3.8. Using the compression time of Motion-JPEG, we estimate the complexity of other coding methods in terms of the complexity of Motion-JPEG. The video coding software used for time calculation of Motion-JPEG, H.264 Intra, H.264 No-Motion and H.264 Inter is *ffmpeg version_1.2*

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

[127] with all parallelization and multi-threading tools disabled.

The test conditions we used are: Intel X5560 CPU @ 2.80GHz, 16GB RAM and 64-bit Operation System, 240 frames of 4CIF 'Harbor' and 'Ice' sequences, with reconstruction quality at around 34dB. The encoding time under Matlab is recorded with function *toc()* and *tic()*, while the encoding time using *ffmpeg* is accessible in the log file. The motion estimation search method used in H.264 Inter is EPZS motion estimation with quarter-pixel motion compensation; full R-D optimization is enabled, the intra period is 24 and 7 B-frames are inserted between I-frame and P-frame. All the configurations for H.264 Intra, H.264 Inter and H.264 No-Motion are the same as that used in Subsection 3.5.1 for R-D performance comparison.

Table 3.7 Estimated encoding complexity comparison for 4CIF 'Harbor' at 34dB

Implementation platform	Coding method	Coding time(s)	Approximated Complexity Ratio (In terms of the complexity of MJPEG)
Matlab	2D-SVD	1852.5	1.5
	2D-SVD coefficient compression only	1200.9	1
	MJPEG-DPCM	1291.1	1
C/C++ (ffmpeg v_1.2)	H.264 Intra	3.5	1.3
	H.264 No Motion	4.2	1.5
	H.264 Inter	8.8	3.3
	Motion-JPEG	2.7	1

Table 3.8 Estimated encoding complexity comparison for 4CIF 'Ice' at 34dB

Implementation platform	Coding method	Coding time(s)	Approximated Complexity Ratio (In terms of the complexity of MJPEG)
Matlab	2D-SVD	1839.3	1.5
	2D-SVD coefficient compression only	1142.0	1
	MJPEG-DPCM	1184.6	1
C/C++ (ffmpeg v_1.2)	H.264 Intra	2.7	1.1
	H.264 No Motion	4.1	1.7
	H.264 Inter	10.2	4.2
	Motion-JPEG	2.4	1

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

As indicated in Table 3.7 and Table 3.8, all these non-ME based approaches, H.264 Intra, H.264 No-Motion, MJPEG-DPCM and 2D-SVD, take much lower computations compared with H.264 Inter, though EPZS fast motion estimation is used. Depending on the video content, the complexity of H.264 Inter with EPZS motion estimation is roughly more than doubled than that of the 2D-SVD approach. H.264 Intra, H.264 No-Motion, MJPEG-DPCM and 2D-SVD have comparatively lower complexities and they can be adopted by devices which cannot afford the intensive computation such as ME.

3.6 Concluding Remarks

In this chapter, we have proposed a new scheme for temporal redundancy removal based upon 2D-SVD. By exploring the energy compaction property of 2D-SVD coefficient matrices, we have designed the proposed low-complexity video codec without ME. Compared with the ME-based hybrid codec, our codec is more efficient in terms of computation. In essence, the most common (stable) content of a group of blocks along the time axis is represented by two 2D-SVD transformation matrices, while the varying content for each individual block is captured by a highly energy compact coefficient matrix. In other words, with 2D-SVD, we use the SVD-like transformation matrices to represent (and to be transmitted once) the most common visual content in video, as a way to exploit temporal redundancy without resorting to ME.

In comparison with the state of the art intra frame codec, H.264/AVC Intra, which also provides frame independency for decoding, our codec outperforms significantly in terms of BD-Bitrate. Compared with other non-ME based low complexity codecs, our approach gives the higher coding efficiency in terms of BD-Bitrate, especially at low bitrates. Compared with the result of H.264 Inter with optimized configuration, the proposed 2D-

Chapter 3. Two-dimensional Singular Value Decomposition Based Video Coding

SVD coding scheme achieves about 60% encoding time saving, less frame dependency at the cost of huge, 130%, bitrate increase. Moreover, upon the successful transmission of the Group Information (GI), which only occupies a small number of bits for a Group Of Pictures (GOP), our codec achieves independent frame decoding which facilitates video manipulating/editing flexibility. These properties show the potential of the proposed 2D-SVD coding scheme for applications requiring low complexity and error resilience.

In this chapter, we mainly discuss the issues related with temporal redundancy removal. In Chapter 4, issues related to spatial redundancy removal will be discussed in detail.

Chapter 4

Rotated Orthogonal Transform for Motion-Compensation Residual Coding

In Subsection 2.2.2, we know that DCT is the orthogonal transform that is most commonly used in image and video compression. The Motion-Compensation residual (MC-residual) is also compressed with the DCT in most video codecs. However, the MC-residual has different characteristic from a natural image. In this chapter, we develop a new orthogonal transform—Rotated Orthogonal Transform (ROT) that can perform better on the MC-residual than the DCT. We derive the proposed ROT based on orthogonal-constrained L1-Norm minimization problem for its sparse property. Using the DCT matrix as the starting point, a better orthogonal transform matrix is derived in terms of energy compaction. In addition, by exploring inter-frame dependency and local motion residual activity, transmission of substantial side information is avoided. The experimental results confirm that, with small computation overhead, the ROT is adaptive to the change of local spatial characteristic of MC-residual frame and provides higher compression efficiency for the MC-residual than DCT, especially for videos with high temporal activity.

4.1 Overview

An important component for spatial redundancy removal is the transform in spatial domain. Typically, the same transform is used to transform both image intensities in image coding and the Motion-Compensation residual (MC-residual) in inter-coding for videos. For example, DCT is used to compress a natural image in the JPEG standard as well as MC-residual in many video coding standards, e.g. H.261/3, MPEG-1/2/4 and H.264/AVC. However, MC-residual has different spatial characteristic from natural image. In a natural image, pixels are highly correlated; while in a MC-residual image, pixels have low correlation and their spatial characteristic changes from area to area in the same MC-residual frame [48][49]. Therefore, it is of interest to study if a transform better than DCT can be developed to fit the local spatial characteristic of MC-residual.

DCT is efficient for compression when the pixels are strongly correlated. As shown in Figure 4.1(a), for a natural image, most of the energy after DCT is concentrated with the top-left coefficients. However, there is little correlation between pixels in MC-residual signals, and DCT is not good enough for MC-residual in terms of energy concentration, since high magnitude transform coefficients are distributed among low and high frequency areas, as shown in Figure 4.1(b).

Therefore, DCT is not an optimized transform for MC-residual and it is worth to study the characteristics of MC-residual and develop new transforms to further improve the compression efficiency of MC-residual coding.

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

(a) DCT of original image

(b) DCT of MC-residual image

Figure 4.1 DCT result of an original image and its MC-residual image

A lot of new transforms have been proposed as discussed in Subsection 2.2.2. Most of these works try to design a transform that yields a sparse coefficient matrix by exploring a local spatial characteristic, and thus achieve a higher compression. In this Chapter, we propose a new block-based transform to derive a sparse coefficient matrix directly by setting the matrix sparsity as the objective function. We aim at a new orthogonal transform with which most of the energy of a MC-residual image is compacted within fewer coefficients, so that we can achieve more efficient data compression. Since such compaction is only possible by adaptively selecting a transform according to local visual content, it is desirable to keep the additional side information and computation small.

To be more specific, in this work, we propose a new, adaptive orthogonal transform, Rotated Orthogonal Transform (ROT), for MC-residual compression based on L1-Norm minimization under orthogonal constraints, with little side information. By converting the orthogonal constraints into convex constraints, we are able to conduct the gradient-based search. By setting the DCT matrix as the starting searching matrix, we are

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

guaranteed to obtain a result not worse than using DCT; we further design a weighting matrix for different coefficients according to their positions such that transformed coefficients are zig-zag ready for entropy coding. In addition, by exploring inter-frame dependency and local motion activity, transmission of substantial side information is avoided. With small additional computation, ROT significantly improves the compression efficiency. Remarkably, the proposed ROT yields higher compression efficiency for high/complex temporal activity sequences and this is desirable in video coding, because typical coders perform worse for high and complex temporal activity videos; in other words, the proposed ROT helps to tackle the worst scenario in video coding.

The next section introduces and analyzes the problem which we are going to model as the ROT. We present the detailed solution for the ROT problem in Section 4.3. Experimental results, presented in Section 4.4, show that the proposed solution improves the compression efficiency of coding MC-residual with small computation overhead. Section 4.5 concludes this chapter.

4.2 Problem Formulation

Following the notations in Section 3.2, we divide an image or a video frame into $m \times m$ non-overlapping blocks. The i^{th} block X_i can be transformed as:

$$M_i = Y_l^T X_i Y_r \quad (4.1)$$

For DCT, one fixed matrix Y_{DCT} is used, and $Y_l = Y_r = Y_{DCT}$; since the transform matrix, Y_{DCT} , is fixed, we can simply represent X_i with M_i (i.e., we only need to code M_i for storage and transmission).

For MC-residual, there is much less correlation between pixels (in comparison with

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

the natural image case), and the pixel characteristic varies significantly from frame to frame, or even from area to area within a frame. Therefore, it is more reasonable to use an adaptive transform for the local areas of an MC-residual frame instead of a fixed transform such as DCT. In this work, we divide the MC-residual frame into Transform Regions (TRs), as shown in Figure 4.2, and derive ROTs for each TR separately. Selection of TR size will be discussed in Section 4.3.

The purpose of transform is to concentrate as much energy into fewer coefficients as possible, and in other words, to make the transformed coefficient matrix as sparse as possible [63][54].

Therefore, for each TR, we formulate the compression problem by the following matrix decomposition problem in (4.2) below, since L1-Norm minimization has the sparse property [98]:

$$\begin{aligned} f(Y) = \min_Y \sum_i \| \text{vec}(Y^T X_i Y) \|_1 \\ \text{s.t. } Y^T Y = I \end{aligned} \quad (4.2)$$

where X_i is a given $m \times m$ matrix, which represents the i^{th} block in a TR and each TR contains n blocks; we choose to use the same for the left and right unitary transform matrices (like many transforms), i.e., $Y_l = Y_r = Y$, and I is a $m \times m$ identity matrix; the operator $\text{Vec}()$ is a vectorization operator, and the definition is the same as (2.26). Obviously, if $Y = Y_{DCT}$, then $Y^T X_i Y$ is the classical 2D-DCT.

As we discussed in Subsection 2.3.1.2, the objective function with L1-Norm is not differentiable and we may try to resort to convex optimization technique to solve this problem. However, further analysis shows that the objective function

$$f(Y) = \sum_i \| \text{Vec}(Y^T X_i Y) \|_1 \quad (4.3)$$

is not a convex function, while the constraints, $Y^T Y = I$, are not convex neither.

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Figure 4.2 Transform Regions (areas in boxes) for CIF *Mobile* MC-residual frame

To overcome the this problem, we deal with the orthogonal constraints by a set of rotation matrices. We convert the non-convex constraints in (4.2) into convex constraints. As a result, a better orthogonal matrix is achieved than the given initial matrix in terms of L1-Norm minimization as to be analysed in the rest of this section.

In (4.2), $Y \in R^{m \times m}$ and $Y^T Y = I$. The constraints $Y^T Y = I$ contain $m(m+1)/2$ equations; therefore, although Y seems to have m^2 variables, it has only $m(m-1)/2$ independent variables, because $m(m-1)/2 = m^2 - m(m+1)/2$.

Since an orthogonal matrix could be derived by rotating a predefined orthogonal matrix, e.g. the DCT basis Y_{DCT} , in the $R^{m(m-1)/2}$ space, we represent these $m(m-1)/2$ independent variables with $m(m-1)/2$ rotation degrees $\theta_1, \dots, \theta_{m(m-1)/2}$. We could derive any orthogonal matrix by multiplying a series of $m(m-1)/2$ rotation matrices with a predefined orthogonal matrix (Y_{DCT} in this study). We construct each rotation matrix $R_{p,q}$ as follows:

Given a $m \times m$ identity matrix I , we set

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

$$\begin{aligned} I(p, p) &= \cos \theta_{p,q} & I(p, q) &= \sin \theta_{p,q} \\ I(q, p) &= -\sin \theta_{p,q} & I(q, q) &= \cos \theta_{p,q} \end{aligned}$$

where $1 \leq p < q \leq m$. In this way, we get rotation matrix $R_{p,q}$. If $m = 6$, then

$$R_{1,4}(\theta_{1,4}) = \begin{bmatrix} \cos \theta_{1,4} & 0 & 0 & \sin \theta_{1,4} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -\sin \theta_{1,4} & 0 & 0 & \cos \theta_{1,4} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\theta_{1,4}$ is a rotation degree.

Starting with Y_{DCT} , we can get any other orthogonal basis $Y_{orthogonal}$ by

$$Y_{orthogonal}(\boldsymbol{\theta}) = \left(\prod_{p=1}^{m-1} \prod_{q=p+1}^m R_{p,q}(\theta_{p,q}) \right) \cdot Y_{DCT} \quad (4.4)$$

where $\boldsymbol{\theta}$ is a $m(m-1)/2 \times 1$ column vector. With this conversion, we eliminate the orthogonal constraints and convert the problem in (4.2) into the following problem with a non-convex objective function, subject to convex constraints:

$$\begin{aligned} \min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) &= \min_{\boldsymbol{\theta}} \sum_i \left\| \text{Vec}(Y_{orthogonal}(\boldsymbol{\theta})^T X_i Y_{orthogonal}(\boldsymbol{\theta})) \right\|_1 \\ s.t. \quad & -\frac{\pi}{2} \cdot \mathbf{1} \leq \boldsymbol{\theta} < \frac{\pi}{2} \cdot \mathbf{1} \end{aligned} \quad (4.5)$$

where $\mathbf{1}$ is a $m(m-1)/2 \times 1$ column vector with all the elements equal 1. For $Y_{orthogonal}$, we set Y_{DCT} as the predefined orthogonal matrix to start with, so the proposed minimization method can derive a better orthogonal transform than DCT in terms of L1-Norm minimization.

With this transformation, although we are not guaranteed to get the global minimum, we can search around the predefined orthogonal matrix, e.g., Y_{DCT} , and get the local minimum by a gradient-based method, since now the domain of the variable $\boldsymbol{\theta}$ is convex. More importantly, in this way, we are guaranteed to get a better orthogonal matrix in

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

terms of L1-Norm minimization, unless Y_{DCT} already leads to the minimum.

The scan order in the existing codecs is in a zig-zag manner [1][2]. Therefore, we multiply elements in $Y_{orthogonal}$ by appropriate weights in (4.5) with respect to their positions in the matrix, so that the resultant Y_{opt} can be better coded using an existing entropy-coding scheme. We use the same weighting matrix in [99], which is defined as:

$$W_{r,c} = r+c \quad (4.6)$$

where (r,c) represents the position index in W , and $r,c=1, 2, \dots, m$.

For a 4x4 block, the weighting matrix is

$$W_4 = \begin{bmatrix} 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

The optimization problem in (4.5) turns into the following problem

$$\begin{aligned} \min_{\theta} f(\theta) &= \min_{\theta} \sum_i \left\| \text{Vec}(Y_{orthogonal}(\theta)^T X_i Y_{orthogonal}(\theta) \circ W) \right\|_1 \\ s.t. \quad & -\frac{\pi}{2} \cdot \mathbf{1} \leq \theta < \frac{\pi}{2} \cdot \mathbf{1} \end{aligned} \quad (4.7)$$

where the symbol “ $\circ$ ” denotes the Hadamard product, which is also known as component-wise product; for example, if $A, B \in \mathbb{R}^{m \times n}$, $(A \circ B)_{r,c} = (A)_{r,c} \cdot (B)_{r,c}$.

With this objective function, during the minimization process, the algorithm tends to have high magnitude coefficients in top-left positions for Y_{opt} since the same coefficient will have smaller cost if it is located in a top-left position than that in a bottom-right position. And this result is more suitable for entropy coding with zigzag scan.

To derive the gradient $\nabla \theta$, we resort to the numerical computation, since the L1-Norm in (4.7) is not differentiable. We calculate the partial derivative with

$$\frac{\partial f(\theta)}{\partial \theta_{p,q}} \approx \frac{f(\theta + \delta \cdot \mathbf{e}_{p,q}) - f(\theta - \delta \cdot \mathbf{e}_{p,q})}{2\delta} \quad (4.8)$$

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

where δ is a very small positive number, and $\mathbf{e}_{p,q}$ is a $m(m-1)/2 \times 1$ vector, with all the elements being zeros except the element at the corresponding location of $\theta_{p,q}$ equals to one; we get

$$\nabla \boldsymbol{\theta} = \left(\frac{\partial f(\boldsymbol{\theta})}{\partial \theta_{1,2}}, \frac{\partial f(\boldsymbol{\theta})}{\partial \theta_{1,3}}, \dots, \frac{\partial f(\boldsymbol{\theta})}{\partial \theta_{m-2,m-1}}, \frac{\partial f(\boldsymbol{\theta})}{\partial \theta_{m-2,m}}, \frac{\partial f(\boldsymbol{\theta})}{\partial \theta_{m-1,m}} \right)^T \quad (4.9)$$

where $\nabla \boldsymbol{\theta}$ is a $m(m-1)/2 \times 1$ vector, which is used for gradient-based search specified in the next section.

4.3 The ROT Algorithm

4.3.1 Details of the ROT Algorithm

In this study, 4x4 ROT is used since H.264/AVC adopted 4x4 Integer DCT (ICT) in the main profile [5]. The algorithm used for searching the minimum around the predefined orthogonal matrix is a gradient-based search described in Figure 4.3. The search method follows Armijo's rule [100]. Armijo's rule is one of the simplest condition imposed by inexact searches for computing suitable stepsizes which are no longer optimal. For a fixed β belonging (0,1), Armijo's rule ensures a certain decrease in the objective function while prevents too large steps to be computed. It is popular for non-convex objective function optimization. The parameters in Armijo's rule are set as $\delta=10^{-4}$, tolerance $\epsilon=0.002$, stepsize $S=1.5$ and $\beta=0.9$.

However, we are not going to apply the searching algorithm to all TRs in a residual frame. This is because only for those TRs that contain a lot of texture residuals or edge residuals, DCT could not perform well. Therefore, search algorithm in Figure 4.3 is only conducted for TRs with high motion residual activity. As in work [101][102], we also use variance for TR's temporal activity estimation and we only perform search

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

algorithm for TRs with variance higher than threshold, $Th = 16$. Here the threshold is set empirically as in [102]. Moreover, we explore the temporal correlation to further save search computation. We will first use the optimal ROT matrix of co-located TR in previous frame, $Y_{orthogonal}(\theta_{Opt-Pre})$, as the transformation matrix for the current TR. If the use of $Y_{orthogonal}(\theta_{Opt-Pre})$ gives more than 2% Rate-Distortion (RD) cost reduction than using DCT matrix, we will use $Y_{orthogonal}(\theta_{Opt-Pre})$ as the optimal matrix for the current TR; otherwise, searching algorithm in Figure 4.3 will be used to derive the optimal matrix $Y_{orthogonal}(\theta_{Opt})$. This leads to the ROT algorithm in Figure 4.4.

Inputs: All the blocks from an MC-residual Transform Region (TR) $\{X_1, X_2, \dots, X_n\}$
 $X_i \in R^{mxm}$
The DCT matrix $Y_{DCT} \in R^{mxm}$
Quantization Parameter (QP)
Outputs: The rotated orthogonal transform matrix $Y_{orthogonal}(\theta_{Opt})$

1. Initialization: $\theta^{(1)} = \mathbf{0}$, $0 < \beta < 1$, $j \leftarrow 1$, step size $S > 0$, tolerance ε
 2. Calculate $\nabla \theta^{(j)}$ according to (10), (11)
 - 2.1 $\theta_1^{(j)} = \theta^{(j)} - \beta \cdot S \cdot \nabla \theta^{(j)} / \|\nabla \theta^{(j)}\|_2$, $k \leftarrow 2$;
 - 2.2 $\theta_k^{(j)} = \theta^{(j)} - \beta^k \cdot S \cdot \nabla \theta^{(j)} / \|\nabla \theta^{(j)}\|_2$
 - 2.3 If $f(\theta_k^{(j)}) > f(\theta_{k-1}^{(j)})$ & $f(\theta_k^{(j)}) < f(\theta^{(j)})$
 $\theta^{(j+1)} = \theta_{k-1}^{(j)}$, goto Step 3
elseif $k < 30$
 $k \leftarrow k+1$, goto Step 2.2
else
 $Y_{orthogonal}(\theta_{Opt}) = Y_{DCT}$
end
 3. if $J(Y_{orthogonal}(\theta^{(j+1)}), QP) - J(Y_{orthogonal}(\theta^{(j)}), QP) > 0$
 $\theta_{Opt} = \theta^{(j)}$ Terminate the algorithm
elseif $\frac{|J(Y_{orthogonal}(\theta^{(j+1)}), QP) - J(Y_{orthogonal}(\theta^{(j)}), QP)|}{J(Y_{orthogonal}(\theta^{(j)}), QP)} < \varepsilon$
 $\theta_{Opt} = \theta^{(j+1)}$ Terminate the algorithm
else
 $j \leftarrow j+1$, goto Step 2
end
-

Figure 4.3 The gradient-based searching algorithm

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Inputs: All the blocks from an MC-residual Transform Region (TR) $\{X_1, X_2, \dots, X_n\}$
 $X_i \in R^{mxm}$
 The DCT matrix $Y_{DCT} \in R^{mxm}$
 Quantization Parameter (QP)
 Optimal degree vector of co-located TR in previous frame $\theta_{Opt-Pre}$

Outputs: The rotated orthogonal transform matrix $Y_{orthogonal}(\theta_{Opt})$

```

1. if var(TR) < Th
    Y_orthogonal(θOpt) = YDCT
else
    if the TR belongs to the 1st P-frame
        run Algorithm in Figure 4.3
    else
        if J(Y_orthogonal(θOpt-Pre), QP) < 0.98 • J(YDCT, QP)
            Y_orthogonal(θOpt) = Y_orthogonal(θOpt-Pre)
        else
            run Algorithm in Figure. 3
        end
    end
end
end
    
```

Figure 4.4 The ROT algorithm

Here the R-D cost is calculated as:

$$J(Y, QP) = D_{RDO}(Y, QP) + \lambda_{RDO} \cdot R_{RDO}(Y, QP)$$

$$\lambda_{RDO} = 0.85 \cdot (2^{(QP-12)/6})^2 \quad (4.10)$$

where QP is quantization parameter; $D_{RDO}(Y, QP)$ and $R_{RDO}(Y, QP)$ are the sum of squared error and compressed number of bits of a certain block using tranform matrix Y and quantization parameter QP ; λ_{RDO} is the Lagrangian multiplier. In Figure 4.3 and Figure 4.4, the RD cost $J(Y, QP)$ is calculated by the average R-D cost in the TR. The overall algorithm is described in Figure 4.4.

4.3.2 Analysis with Varying TR Sizes

In our implemetation, 1 bit is used to indicate whether this TR is coded with DCT and 1 bit is used to indicate whether this TR is coded with previous optimal rotation vector $\theta_{Opt-pre}$ or the current optimal rotation vector θ_{Opt} . The optimal rotation vector θ_{Opt}

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

is quantized into 7 bits for each rotation, $\theta_{p,q}$. For the 4x4 orthogonal basis, we have 6 ($=4 \times (4-1)/2$) rotations, which is 42 bits. However, we don't have to transmit θ_{Opt} as overhead, unless when we need to code the TR with current optimal rotation vector θ_{Opt} . The larger the size of TR, the less the portion of side information occupies. On the other hand, the smaller the size of TR, the better the ROT is suited to local spatial characteristic. Therefore, we need to choose a reasonable TR size.

In Figure 4.5, we show the ROT coded average PSNR for 200 frames of CIF *Mobile* sequence with 1, 8, 32 and 64 TRs for each frame (correspond to TR size of 288x352, 144x88, 72x44 and 36x44).

As plotted in Figure 4.5, with TR size decreasing from 288x352 to 72x44, the average PSNR keeps increasing. However, when we code each frame with 64 TRs (TR size = 36x44), the compression efficiency begins to drop. In Table 4.1, we further show the average PSNR (QP=37, 34, 31, 28 and 25) for several CIF sequences by varying TR sizes. As indicated in bold numbers, TR size with 72x44 yields highest averaged PSNR for these four sequences. Therefore, we set dimension of TR from 44 pixels to 80 pixels by dividing the frame width and height by 2^L , where L is a positive integer, i.e., 72x44 for 4CIF, CIF and QCIF resolution, and 80x60 for VGA and QVGA resolution.

Figure 4.5 Comparison of compression efficiency with varying TR sizes

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Table 4.1 Average PSNR(dB) comparison for CIF sequences (200 frames) with varying TR sizes (The bold number indicates the best result among four TR sizes)

Sequences	Number of TRs (TR size)			
	1(288x352)	8(144x88)	32(72x44)	64(36x44)
Flower	33.8	34.1	34.1	33.8
Mobile	32.5	32.6	32.8	32.4
Tempete	34.5	34.5	34.7	34.3
Waterfall	34.9	35.0	35.2	34.8

4.3.3 TR Selection Rate Analysis

Figure 4.6 TR selection results (Solid rectangles: TRs choose to use DCT, Dash rectangles: TRs choose to use ROT)

Table 4.2 TR Selection Rate (SR) in Percentage

Resolution	Sequence	SR in Stage one (SRS1)	SR in Stage two (SRS2)	Overall SR (SRS1xSRS2)
QCIF	Bus	88.7	82.5	73.2
	Coastguard	69.7	80.2	55.9
	Foreman	46.3	73.5	34.0
CIF	Mobile	89.4	77.4	69.2
	Tempete	78.4	74.2	58.2
	Waterfall	57.3	70.5	40.4

As illustrated in Figure 4.4, two TR selection stages are carried out before the gradient-based searching algorithm in Figure 4.3. In the first stage, the variance of TR is used to determine if this TR contains high motion residual activity. In the second stage, for those

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

TRs selected as high motion residual activity TR in the first stage, we further check if they can use optimal rotational matrix derived from previous co-located TRs. In Table 4.2, we show the TR Selection Rate in Stage one (SRS1) and TR Selection Rate in Stage two (SRS2) for 6 sequences with 200 frames each. As observed, in the first stage, the selection rate is closely related to the motion residual activity of the sequence. Usually, sequences with high temporal activity will result in high motion residual activity. To show the temporal activity of each sequence, we measure the temporal activity with Modified Temporal Information (Modified_TI), which is a variation based on Temporal Information defined by ITU-T recommendation [128]. Firstly, the difference between two consecutive frames is calculated as $Df_n(x, y) = F_n(x, y) - F_{n-1}(x, y)$, here $F_n(x, y)$ is the pixel at the x^{th} row and y^{th} column of n^{th} frame in time. The standard deviation (std) over all the pixels in the differential frame, $Df_n(x, y)$, is then computed. The measure of Modified_TI, is computed as the average of the $n-1$ standard deviation values of $Df_n(x, y)$, which is shown in Equation (4.11). If the maximum standard deviation value of all $Df_n(x, y)$ is selected, it will be the TI defined in [128]. The Modified_TI based on 200 frames of 6 sequences is calculated in Table 4.3. From Table 4.2 and Table 4.3, we can observe that for those sequences with high Modified_TI values, i.e. *Mobile*, *Bus*, about 90% of TRs are selected in stage one, while for sequences with low Modified_TI values, i.e. *Foreman*, *Waterfall*, about 50% TRs are selected in stage one. Therefore, we can observe a correlation between sequence's temporal activity and TR Selection Rate in Stage one (SRS1).

$$\text{Modified_TI} = (\sum_{n=2}^N \text{std}[Df_n(x, y)]) / (N - 1) \quad (4.11)$$

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Table 4.3 Result of Modified Temporal Information (Modified _TI)

Resolution	Sequence	Modified _TI
QCIF	Bus	25.4
	Coastguard	13.7
	Foreman	9.8
CIF	Mobile	20.6
	Tempete	15.4
	Waterfall	5.0

At the second stage, the selection rates distributed between 70.5%-82.5% for the six sequences, which is less correlated to the temporal activity of the sequence. Therefore, the overall selection rate is mostly determined by the selection rate in stage one. As listed in the third column in Table 4.2, the overall TR selection rate (selection rate in stage one $\times$ selection rate in stage two) varies from 34.0% for *Foreman* to 73.2% for *Bus*. In Figure 4.6, we give two examples of the overall TR selection result with proposed ROT algorithm for the 3rd residual frame of CIF *Waterfall* and *Mobile* at QP = 31. The TRs in dash rectangle are coded with ROT, while TRs in solid rectangle are coded with DCT. In line with the analysis in Table 4.2, more TRs are coded with ROT for videos with high motion residual activity, e.g., for the sequence *Mobile*.

4.3.4 Convergence Performance Analysis

Figure 4.7 Convergence performance with the proposed algorithm for the top-left TR

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Figure 4.8 Averaged iterations need for each sequence

Figure 4.7 shows the convergence performance with the proposed ROT algorithm for the top-left TRs of each residual frame in Figure 4.6. As demonstrated in Figure 4.7, at $QP = 31$, the algorithm converges with 4 and 5 iterations for *Waterfall* and *Mobile*. Further analysis shows that, more R-D cost reduction $((30476-28943)/30475 \times 100\% = 5.3\%)$ is achieved for TR in *Mobile* compared with that $((10074-9703)/10075 \times 100\% = 3.7\%)$ for TR in *Waterfall*. This is consistent with the residual pixel activities of these two TRs. For the top-left TR in Figure 4.6(a), its variance is 23.5; while the variance of top-left TR in Figure 4.6(b) is 65.3. This result, along with analysis in previous subsection, implies that the proposed ROT has more potential for high motion residual activity video compression.

In Figure 4.8, we plot the averaged iterations for each sequence. The averaged iterations are calculated for all the TRs that need to run iterative algorithms in Figure 4.3 within a sequence of 200 frames for all QPs (37, 34, 31, 28 and 25). As indicated in Figure 4.8, on average, it takes 6.05 iterations to derive the optimal rotation vector θ_{Opt} . The averaged iterations for each sequence is not strongly related to the sequence

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

content. The number of average iterations ranges from 5.31 for *Foreman* to 7.02 for *Soccer*. Compared with 1D-Transform [59] mentioned in Subsection 2.2.2, R-D cost calculation iterations required by the proposed method to derive the ROT are much fewer. This is because the proposed ROT algorithm explores the local motion residual activity and inter-frame dependency so that we don't have to calculate a new ROT for every TR. On average, it requires 6 iterations for each TR to derive the ROT, while 1D-Transform requires 17 R-D cost calculation iterations to determine the best transform for each block.

4.3.5 Energy Compaction Comparison with DCT

In Figure 4.9, we show the base functions of DCT and the base functions derived by ROT with 6 iterations for the TR of CIF *Mobile* sequence (indicated in Figure 4.9(a)). As demonstrated, the base functions of ROT are more adaptive to the specific visual content. The base functions of DCT cannot achieve this because they are not adaptive to the specific content.

Figure 4.10 and Figure 4.11 give a further illustration for the proposed ROT against the DCT. For the same block in Figure 4.10, after quantization, the ROT quantized coefficients have eight non-zero coefficients while DCT generates ten non-zero quantized coefficients. It is also observed that the ROT coefficients are not as zig-zag ready as DCT coefficients. This will incur a compression efficiency loss in entropy coding stage.

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

(a) the MC-residual frame; (b) the DCT base function; (c) the ROT base function derived from the TR in the black box.

Figure 4.9 Comparison of base functions of DCT and the base functions derived by ROT

Figure 4.10 Comparison transformed coefficients of DCT and ROT

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Figure 4.11 Comparison of energy retained by the number of coefficients of DCT and ROT

Table 4.4 Coding Gain (dB) comparison of DCT and ROT

	Sequences	DCT	ROT
QCIF	Bus	2.46	2.63
	Coastguard	1.98	2.20
	Foreman	1.70	1.92
	Football	3.42	3.77
CIF	Flower	1.39	1.57
	Mobile	2.12	2.49
	Tempete	1.73	2.04
	Waterfall	1.37	1.53
4CIF	Soccer	1.26	1.36
	Crew	1.31	1.38
Average Coding gain		1.87	2.09
Average Coding gain improvement over DCT			0.22

Moreover, in Figure 4.11, we show the retained energy versus the number of transform coefficients. As indicated in Figure 4.11, the three largest ROT transformed coefficients retain 91% of energy for the original block, while we need six largest DCT coefficients to recover the same amount of energy. However, this analysis is just based on the

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

assumption that we have coded the location of the most significant ROT coefficients, which is not done by the codec used in our approach. Therefore, to show a more general comparison, we further show the energy compaction advantage of ROT over DCT in terms of coding gain (as defined in (2.1) of Subsection 2.1.1).

In coding gain comparison, we computer the DCT and ROT for blocks in the TRs that code with ROT. The data is calculated with 200 MC-residual frames from four QCIF sequences, four CIF sequences and two 4CIF sequences. Table 4.4 shows the comparison results. As calculated, compared with DCT, ROT gives 0.22dB coding gain improvement. The coding gain improvement for ROT is about 11.7% compared with DCT. Experimental results in Section 4.4 also prove that coding efficiency improvement achieved by energy compaction property of ROT coefficients surpasses the compression efficiency loss incurred by less zig-zag ready coefficients.

4.4 Experimental Results

To provide a comprehensive comparison, ten test sequences with various contents and resolutions are used in our experiments. We code four QCIF (144x176) sequences, four CIF (352x288) sequences and two 4CIF (704x576) sequences with 200 frames for each at 30 Frame-Per-Second (FPS).

Four codecs are used for comparison. H.264 with Adaptive Quantization Matrix Selection (AQMS) [103] is used as a benchmark. In H.264 with AQMS, four types of quantization matrices are proposed and each frame is coded multiple times to obtain the optimal quantization matrix. The implementation software of H.264(with AQMS) is JM11.0KTA2.5 [129] with only AQMS enabled and all other new features disabled in our experiment.

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

The remaining three codecs use Directional Discrete Cosine Transforms (DDCT) [58], 1-D Transform [59] and Adaptive Orthogonal Transform [104]. In the DDCT based codec, new DCT directions as well as scan orders are developed for each block in two-dimensional manner; while in 1-D Transform based codec, new DCT directions and scan orders are developed in one-dimensional way. Similar to ROT, both DDCT and 1-D transform try to derive a new transform that suits the local motion residual activity better than DCT. Adaptive Orthogonal Transform based codec is our previous work [104], in which we only apply the algorithm in Figure 4.3 to the whole residual frame.

Here we list some of the important encoding parameters in the configuration. The Group of Pictures (GOP) is set to 16. The Quantization Parameters (QP) are 37, 34, 31, 28 and 25. Both IPPP and IBBP coding structures are tested and 7 B-frames are inserted between I-frame and P-frame for IBBP coding structure. Quarter-pixel resolution and full-search motion estimation is used with search range of 7 (16 for 4CIF sequences). Entropy coding is performed with Context Adaptive Variable Length Codes (CAVLC). The implementation is based on H.264 reference software JM 11.0 [126]. For ROT, 1bit is used to indicate whether this TR is coded with DCT and 1 bit is used to indicate whether this TR is coded with previous optimal rotation vector $\theta_{Opt-pre}$ or the current optimal rotation vector θ_{Opt} . The optimal rotation vector θ_{Opt} is quantized into 7 bits for each rotation, $\theta_{p,q}$, as the side information if we need to transmit it.

4.4.1 Objective Coding Results Comparison

The Rate-Distortion (R-D) Curves of the four test sequences are plotted in Figure 4.12. The PSNR in Y-axis is calculated by averaging the PSNR of 200 compressed frames for each sequence. Figure 4.12 (a) and (b) are coding results with P-frame only while Figure

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

4.12 (c) and (d) are coding results with both P-frame and B-frame. The curve of “H.264(with AQMS)” is the coding result of H.264 with adaptive quantization matrix selection, which is used as the benchmark for comparison. The curves of “Adaptive Orthogonal Transform”, “ROT”, “DDCT” and “1D Transform” correspond to the coding result of work in [104], the proposed ROT method, Directional Discrete Cosine Transform (DDCT) based codec in [58] and 1-D Transform based codec in [59]. By taking the RD curve of “H.264 (with AQMS)” as benchmark, we also calculate the Bjontegaard-Delta (BD)-Bitrate [36] as introduced in Subsection 2.1.3.

Figure 4.12 R-D performance comparison

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Table 4.5 BD-Bitrate (in percentage) comparison among DDCT, adaptive orthogonal transform, 1D transform and proposed ROT

Resolution	Sequences	DDCT[58]		Adaptive Orthogonal Transform [104]		1D Transform[59]		ROT	
		IPP	IBP	IPP	IBP	IPP	IBP	IPP	IBP
QCIF	Bus	-1.5	-0.5	-4.1	-2.2	-5.7	-2.6	-7.4	-2.9
	Coastguard	-1.7	-0.4	-3.8	-0.7	-4.7	-1.2	-6.3	-2.8
	Foreman	-2.2	-0.0	-2.1	-0.4	-4.2	-1.2	-4.1	-1.6
	Football	-1.7	-0.2	-4.9	-0.8	-6.0	-1.7	-5.9	-3.8
CIF	Flower	-0.2	-0.1	-4.5	-1.4	-5.8	-2.2	-8.1	-2.9
	Mobile	-0.3	-0.3	-4.5	-1.0	-5.1	-2.7	-7.2	-3.2
	Tempete	-1.9	-0.2	-3.2	-0.7	-5.7	-2.5	-5.7	-2.4
	Waterfall	-2.1	-0.1	-1.8	-0.5	-4.1	-1.3	-5.4	-1.3
4CIF	Soccer	-1.2	-0.2	-5.2	-2.2	-6.6	-3.5	-7.1	-4.3
	Crew	-2.2	-0.3	-5.0	-1.9	-6.8	-2.7	-6.5	-3.9
Average		-1.5	-0.2	-3.9	-1.2	-5.5	-2.2	-6.3	-2.9

More experiment data are summarized in Table 4.5. The proposed ROT based approach achieves 6.3% and 2.9% BD-Bitrate saving compared with H.264 with AQMS for IPPP and IBBP coding structures. It is also worth noting that the proposed ROT achieves more bitrate saving (in BD-Bitrate) for video sequences with high temporal activity (i.e., *Mobile* and *Bus*), while comparatively low BD-Bitrate saving for sequences with low temporal activity (*Foreman*, *Waterfall*). This is in line with the analysis in Section 4.3, that for low temporal activity videos, most of the MC-residual pixels has low activities, for which DCT is good enough to compress and leaves little room for our method to improve; for high/complex temporal activity videos, there are more high magnitude MC-residual pixels and this gives our method more room to improve the compression efficiency. We can also observe that when B-frame is used, the gain of the proposed ROT is smaller than that of IPPP coding structure. This is because the application of B-frame achieves better prediction and further reduces the activity of motion compensation residual thus reduces the room for ROT to improve.

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

4.4.2 Visual Quality Comparison

First of all, we show the visual results with I-frame and P-frame only in Figure 4.13. For *Mobile* (at 30 FPS) around 1520 Kbps, the original, H.264+AQMS decoded and ROT decoded frames with their amplified portions are shown. Close observation shows that the ROT results preserve the detailed texture well while the H.264+AQMS result loses the details such as pedals of the flower, as shown in the amplified parts in Figure 4.13 (a), (b) and (c).

To give a comprehensive evaluation on the subjective perceptual quality, we also worked with 14 unpaid voluntary test persons. The tests were performed according to [128]. Six sequences are compressed at similar bitrates as indicated in Table 4.6 with B-frame used. The structure of the Basic Test Cell consists of two consecutive presentations of the sequence under test. First the original version of the video sequence is displayed, followed by the decoded sequence. Then a message is shown for 5 seconds asking the viewers to vote. The presentation of the video clips is preceded by a mid-level grey screen for a duration of one second. For each sequence, we go through the voting of all the coding approaches before we start to vote the next sequence. Test subjects were asked to evaluate the overall quality with Mean Opinion Score (MOS) range from 0 to 10 (10 denotes a quality of reproduction that is perfectly faithful to the original while 0 denotes a quality of reproduction that has no similarity to the original). We show the MOS value of six test sequences in Table 4.6. From Table 4.6, we can conclude that when B-frame is used, all the coding approaches have almost the same perceptual quality.

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Table 4.6 Subjective coding efficiency comparison

Resolution	Sequence	Bitrate	MOS				
			H.264(AQMS)	Adaptive Orthogonal Transform [104]	1D-Transform	DDCT	ROT
QCIF	Bus	100kbps	6.6	6.5	6.6	6.6	6.5
	Coastguard	100kbps	7.0	6.9	7.0	7.1	7.0
CIF	Tempete	600kbps	6.7	6.6	6.7	6.8	6.8
	Mobile	600kbps	6.4	6.3	6.1	6.3	6.4
4CIF	Crew	3500kbps	7.2	7.1	7.2	6.9	7.1
	Soccer	3500kbps	6.8	6.8	6.9	6.9	6.9
Average			6.8	6.7	6.8	6.8	6.8

Figure 4.13 Subjective coding performance comparison of the frame of CIF *Mobile*.

4.4.3 Computational Overhead Analysis

In the proposed method, there is computational overhead for ROT due to the iteration algorithm applied. However, the computation of transform is very small compared with other time-consuming components, such as motion-estimation, content adaptive entropy coding and etc. [130].

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

Figure 4.14 Average coding time increase for 10 testing sequences

In Figure 4.14, we summarize the averaged increase of overall encoding time for 5 QPs (37, 34, 31, 28 and 25) for 200 frames of each sequence with Intel X5560 CPU @ 2.80GHz, 16GB RAM and 64-bit Operation System. The encoding time is calculated with *clock()* function in the reference software [126]. It is observed that the average coding time increase ranges from 8.7% to 13.5%, with 11.8% increase on average. This result is also in line with our previous analysis in Section 4.3. As listed in Table 4.2 and Table 4.3, for sequences with low temporal activity, the overall TR selection rate is lower than that with high motion residual activity. As a result, for sequences with low temporal activity, i.e. *Foreman*, *Waterfall*, the computation overhead introduced by iterative algorithm is lower than that of high temporal activity videos, i.e. *Flower* and *Crew*.

4.5 Concluding Remarks

In video coding, Motion Compensation residual (MC-residual) has different spatial characteristic compared with a natural image, and this makes the current compression schemes less efficient for MC-residual. Moreover, since such residual is visual content related, it is desirable to derive a new transform (non-DCT) that is adaptive to the local characteristic of residual to yield coefficient compaction, while keeping the necessary side information and computation small.

In this work, we have proposed an L1-Norm minimization based algorithm to generate a Rotated Orthogonal Transform (ROT) for coding MC-residual frames. By converting the orthogonal constraints into convex constraints for the problem formulation, we are able to conduct the gradient-based search for the optimization. By setting the DCT matrix as the starting transform matrix, we guarantee to obtain a better matrix in terms of L1-Norm minimization. We further design a weighting matrix by setting proper weights for the transformed coefficients in line with zigzag scan order. In this way, the optimization result is customized to the entropy coding available. In addition, by exploring inter-frame dependency and local temporal activity, transmission of substantial side information is avoided.

Analysis and overall experimental results have confirmed that using the ROT derived from the proposed algorithm, we can improve the compression efficiency of the MC-residual for videos containing diversified visual content and motion with small computational overhead. The better performance of the proposed transform for high/complex temporal activity video is highly desirable in video coding; because this helps to improve the R-D performance at the most difficult scenario and alleviate quality

Chapter 4. Rotated Orthogonal Transform for Motion-Compensation Residual Coding

fluctuation throughout the frames (smaller quality fluctuation during the encoding/streaming has positive impact on the perceived quality by the viewers). In our future work, scan order, entropy coder and error resilience for the ROT coefficients will be studied towards further performance improvement.

This chapter mainly discusses the issue related with spatial redundancy removal. After temporal and spatial redundancy removal, entropy coder is used to remove the statistic redundancy. In Chapter 5, the details related to entropy coding will be discussed.

Chapter 5

Mode Dependent Templates and Scan

Order for Intra Lossless Coding

Subsection 2.1.2 and Subsection 2.2.3 have introduced the application of both lossy and lossless video coding. In H.264/advanced video coding (AVC), lossless coding and lossy coding share the same entropy coding module. However, the entropy coders in the H.264/AVC standard are originally designed for lossy video coding and do not yield adequate performance for lossless video coding. In this chapter, we analyze the problem with the current lossless coding scheme and propose a Mode Dependent Template (MD-Template) based method for intra lossless coding. By exploring the statistical redundancy of the prediction residual in the H.264/AVC intra prediction modes, more zero coefficients are generated. By designing a new scan order for each MD-Template, the scanned coefficients sequence fits the H.264/AVC entropy coders better. A fast implementation algorithm is also designed. With little computation increase, experimental results confirm that the proposed fast algorithm achieves better lossless compression compared with the current H.264/AVC standard.

5.1 Overview

The H.264/AVC standard has been developed by mainly focusing on lossy coding. However, lossless video coding is also important in areas such as source data archive and distribution [24], medical imaging [25] and machine based video analysis. For source video archive and distribution, a video needs to be re-encoded repeatedly; with lossy coding, coding error will accumulate. For medical imaging, disease diagnoses require medical images to have no data loss. Thus, rather than the use of lossy compression with relatively high compression ratio, mathematically lossless compression techniques are favored in this field. For image/video analysis tasks performed by a machine, e.g. military object recognition and automatic product examination, being “perceptual lossless” is meaningless and lossless image/video coding is also preferred. More details about H.264 lossless coding is discussed in Subsection 2.1.2 and Subsection 2.2.3.

In H.264/AVC intra coding, pixels from the already coded neighboring blocks are used to predict the current block. Given a 4x4 block, 9 intra prediction modes are used for spatial redundancy removal as shown in Figure 5.1 [91]. The prediction samples in vertical mode (mode 0) are obtained from the four neighboring pixels (A to D) above the block to be coded. Sample A is copied to every pixel in the first column of the block, sample B is copied to every pixel in the second column of the block, and so on. Similar to mode 0, the prediction samples in horizontal mode (mode 1) are copied from the four neighboring pixels (I to L) at the left of the block to be coded. For the DC mode (mode 2), the average of adjacent samples is calculated as the predicted value. For modes 3-8, the value of a predicted pixel is computed as a weighted average of the neighboring pixels. After intra prediction, we get the residual block. For lossy intra coding, as shown

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

in Figure 5.2(a), Integer Discrete Cosine Transform (ICT) and quantization are first applied to the input intra prediction residual block. Followed by the zigzag scan, the scanned coefficients are sent to the entropy coders for statistical redundancy removal. After ICT, the high magnitude coefficients tend to concentrate at the top-left part of the block and after quantization, the top-left part will have higher probability of having non-zero coefficients compared with the bottom-right part. Therefore, after the zigzag scan, the beginning part is more likely to have non-zero coefficients compared with the ending part. In the meanwhile, since the coefficients at the bottom-right corner tend to be small, after quantization, they are more likely to be quantized to consecutive zeros; the larger the quantization step is, the more zeros will be generated. The two entropy coders in H.264/AVC, context-based adaptive variable length coder (CAVLC) [20], and context-based adaptive binary arithmetic coder (CABAC) [21], are designed to explore these statistical characteristics.

However, in lossless coding, as shown in Figure 5.2(b), the data sent to entropy coders are residual errors, which are the differential pixel values between the original and the predicted pixel values without transform and quantization. Residual data differ from quantized transform coefficients in at least two aspects: 1) it has fewer zero entries; 2) all the positions in the prediction residual block have almost the same probability of having non-zero entries. Thus, the zigzag scan and the conventional entropy coders in H.264/AVC cannot provide the best coding performance for lossless video coding.

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

Figure 5.1 Nine intra prediction modes [91]

(a) A demonstration of H.264 lossy intra coding process.

(b) A demonstration of H.264 lossless intra coding process.

Figure 5.2 Comparison of H.264 intra lossy and lossless coding process.

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

In the High Efficiency Video Coding (HEVC) standard [55], the Mode Dependent Transform (MDT) learns a pair of transform matrices for each intra prediction mode and stores them in both encoder and decoder. Inspired by MDT, we explore the statistical redundancy of prediction residual within intra prediction modes in this work. By designing Mode Dependent Templates (MD-Templates) to be subtracted by the intra prediction residual, more zero entries are generated; by designing the scan order corresponding to each template, the beginning part of the scanned coefficient sequence has a higher probability of having non-zero coefficients compared with the ending part. In this way, the resultant scanned sequence is more suitable for the entropy coding methods in H.264/AVC and experimental results confirm that considerable saving, in terms of bits, is achieved by the proposed MD-Templates method compared with H.264/AVC fidelity range extensions high profile [32].

The rest of this chapter is organized as follows: the methodology we use to derive MD-Templates is described in Section 5.2. In Section 5.3, the two MD-Template based coding algorithms are given and the statistical comparison before and after using the MD-Templates is also illustrated. Experimental results, presented in Section 5.4, show that the proposed scheme improves the compression efficiency in terms of bit saving. Section 5.5 concludes this chapter.

5.2 Problem Formulation

5.2.1 MD-Template Learning

Block based templates are often used for both image/video processing [105] and coding [106] tasks. In this work, we formulate the problem of MD-Template learning for each intra prediction mode with the following matrix decomposition problem:

Firstly, we will classify residual blocks into different sets according to the given templates. Here we define a set Ω_{W_j} as

$$\begin{aligned} \Omega_{W_j} = \{X_i \mid \|vec(X_i)\|_0 - \min_{a_i} \|vec(X_i - a_i W_j)\|_0 \geq 1\} \\ \text{s.t. } W_j(r, c) \in \{0, 1\}, a_i \in \{-255, -254, \dots, -1, 0, 1, \dots, 254, 255\} \end{aligned} \quad (5.1)$$

where X_i is a given 4×4 matrix, which represents an intra-predicted residual block belonging to one of the 9 modes; W_j is a 4×4 binary matrix with each entry being either zero or one. W_j is what we are going to choose for MD-Templates; a_i is a scale integer ranging from -255 to 255. The operator $vec()$ is a vectorization operator with similar definition as (2.26).

As we discussed in Subsection 2.3.1.3, the L0-Norm, $\|\cdot\|_0$, of a vector counts the total number of non-zero entries in the vector. In (5.1), $\min_{a_i} \|vec(X_i - a_i W_j)\|_0$ means that given a W_j , we try to find an a_i , such that for each X_i , after being subtracted by $a_i W_j$ the non-zero entries are minimized. With the condition of $\|vec(X_i)\|_0 - \min_{a_i} \|vec(X_i - a_i W_j)\|_0 \geq 1$, definition (5.1) means that given a W_j , if more zero entries are generated in X_i after being subtracted by $a_i W_j$, then X_i belongs to the set Ω_{W_j} . As we mentioned in Subsection 2.3.1.3, L0-Norm optimization problem is NP-hard. Therefore, to classify X_i to different Ω_{W_j} , we resort to exhaustive search. Since W_j is a 4×4 binary matrix, we have $2^{4 \times 4} = 65536$ sets to derive. To understand this matrix decomposition more intuitively, an

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

example is given in Figure 5.3. Suppose that we are given a W_j as Figure 5.3(b), and a residual block X_i as Figure 5.3(a). If we choose a_i to be either 3 or -4, more zeros entries are generated. However, if $a_i = -4$, only one more zero entry is generated; while if $a_i = 3$, three more zero entries are generated, as shown in Figure 5.3(c). Therefore, we will set $a_i = 3$ since it generated minimal number of non-zero entries as stated in $\min_{a_i} \| \text{vec}(X_i - a_i \cdot W_j) \|_0$. At the same time, we classify X_i to Ω_{W_j} according to the definition in (5.1).

Figure 5.3 A demonstration of X_i classification process

Table 5.1 Training sequences set A

Resolution	No. of frames	Sequences
QCIF (144x176)	200	Container
		Bus
		Silent
		Hall
CIF (288x352)	50	Waterfall
		City
		Paris
		Football
1080P (1080x1920)	5	Ducks_take_off
		Blue_sky
		Sunflower
		Rush_hour

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

After we have Ω_{W_j} for a certain intra mode, we define

$$S(W_j) = \sum_i (B(X_i) - B(X_i - a_i W_j) - SideInfo) \quad (5.2)$$

s.t. $X_i \in \Omega_{W_j}$

Here $B(X_i)$ is the number of bits required for coding residual block X_i using H.264 Intra lossless coding, $B(X_i - a_i W_j)$ is the number of bits required for coding $X_i - a_i W_j$ using H.264 Intra lossless coding, **SideInfo** represents bits required to code a_i and W_j . Therefore, function $S(W_j)$ shows the total number of bits saved before and after using W_j for residual blocks in a certain intra prediction mode.

Table 5.2 Probability distribution of selected Intra prediction modes

Intra Prediction Mode	Probability
Mode 0	0.302
Mode 1	0.262
Mode 2	0.182
Mode 3	0.071
Mode 4	0.074
Mode 5	0.024
Mode 6	0.035
Mode 7	0.021
Mode 8	0.029

Figure 5.4 Probability distribution of absolute value of first 20 non-zero coefficients in residual blocks

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

In [107]-[109], the learning and testing are conducted by learning on a set of sequences and testing on another set of sequences. In our work, we follow this learning methodology. Specifically, the sequences for MD-Templates learning are summarized in Table 5.1. All the training sequences are available at <http://media.xiph.org/video/derf/>. Among the nine intra prediction modes, the first five modes have higher occurrence probabilities than the rest four modes [110] and in [26] only first five modes are modified. We also record the overall selected intra prediction modes based on the learning sequences in Table 5.1, and show the data in Table 5.2. As indicated, Mode 0 to Mode 4 account for more than 89% of prediction modes for intra prediction. Therefore, following the methodology in [26], we only learn the MD-Templates for these five modes.

Furthermore, we also generate the statistical distribution of the absolute values of non-zero entries for the residual blocks derived from the training sequences set A and plot the statistics of the first 20 coefficients in terms of absolute value in Figure 5.4. As the statistical data indicated, more than 75% of the non-zero prediction residual values range from -4 to 4. Therefore, we constrain a_i at $\{-4, -3, -2, -1, 1, 2, 3, 4\}$. Based on the probability distribution of absolute value of first 4 non-zero coefficients in Figure 5.4, we use Huffman coding algorithm to generate the codeword for a_i as shown in Table 5.3. The side information, *SideInfo* in (5.2), includes the codeword for a_i and 4 bits for each MD-Template, since we are going to learn 16 MD-Templates (In Subsection 5.4.1, we will discuss the number of MD-Templates we used).

For a certain intra prediction mode, we try 65,536 W_j and rank the derived $S(W_j)$ in an descending order as $S_1, S_2 \dots S_{65536}$. Then we name W_1 for the MD-Template that corresponds to S_1 , W_2 for the MD-Template that corresponds to S_2 and so on so forth. In

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

other words, W_1 yields the most number of bit saving for a certain intra prediction mode, and W_2 yields the second most number of bit saving for a certain intra prediction mode.

In Figure 5.5, we show W_1 to W_{16} from (a) to (p), which are the top 16 MD-Templates learned from training set A for Mode 0. It is worth noting that these 16 MD-Templates have strong directional patterns. For most of them, the positions of ‘1’s are along the horizontal direction except Figure 5.5 (g), Figure 5.5 (k) and Figure 5.5 (n). On the other hand, if we use the training sequences set B shown in Table 5.4 to learn the MD-Templates for Mode 0, W_1 to W_{16} are shown in Figure 5.6. Most of MD-Templates in Figure 5.6 also have ‘1’ entries distributed along horizontal direction, except Figure 5.6 (k), Figure 5.6 (l) and Figure 5.6 (o). Therefore, we take the directional exceptions of MD-Templates, Figure 5.5 (g), Figure 5.5 (k), Figure 5.5 (n), Figure 5.6 (k), Figure 5.6 (l) and Figure 5.6 (o) in Mode 0, as a result of training set bias and we use the union set of top horizontally distributed MD-Templates in Figure 5.5 and Figure 5.6 as the 16 MD-Templates for Mode 0 as demonstrated in Figure 5.7.

Similarly, we derive the 16 MD-Templates for Mode 1, Mode 3 and Mode 4 in Figure 5.8, Figure 5.9 and Figure 5.10, respectively. For Mode 2, W_1 to W_{16} do not show strong directional patterns, therefore, we just choose the learned 16 MD-Templates, W_1 to W_{16} , as shown in Figure 5.11.

Table 5.3 Codeword table for coefficients a_i

Value of a_i	Sign bit	Code bits
1	1	1
-1	0	1
2	1	01
-2	0	01
3	1	001
-3	0	001
4	1	000
-4	0	000

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

Table 5.4 Training sequences set B

Resolution	No. of frames	Sequences
QCIF (144x176)	200	Carphone
		Foreman
		Mobile
		Salesman
CIF (288x352)	50	Bus
		Hall
		Silent
		Tempete
1080P (1080x1920)	5	Ducks_take_off
		Blue_sky
		Station
		Old_town_cross

Figure 5.5 The top 1st to 16th learned MD-Templates from training set A for Mode 0

Figure 5.6 The top 1st to 16th learned MD-Templates from training set B for Mode 0

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

<table><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	<table><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
1	1	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	1	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	1	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	1	1	1																																																																																																																																				
1	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	1	0	0																																																																																																																																				
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)																																																																																																																																
<table><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	<table><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
0	0	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	1																																																																																																																																				
0	1	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	1	0																																																																																																																																				
(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)																																																																																																																																

Figure 5.7 The 16 MD-Templates used for Mode 0

<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr></table>	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	1																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)																																																																																																																																
<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)																																																																																																																																

Figure 5.8 The 16 MD-Templates used for Mode 1

<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	1																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)																																																																																																																																
<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)																																																																																																																																

Figure 5.9 The 16 MD-Templates used for Mode 3

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0
0	0	0	1																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)																																																																																																																																
<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	1	0																																																																																																																																				
(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)																																																																																																																																

Figure 5.10 The 16 MD-Templates used for Mode 4

<table><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
1	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)																																																																																																																																
<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	<table><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	1																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
(i)	(j)	(k)	(l)	(m)	(n)	(o)	(p)																																																																																																																																

Figure 5.11 The 16 MD-Templates used for Mode 2

5.2.2 Scan Order for Each MD-Template

<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	<table><tr><td>1</td><td>2</td><td>3</td><td>4</td></tr><tr><td>13</td><td>14</td><td>15</td><td>16</td></tr><tr><td>5</td><td>6</td><td>7</td><td>8</td></tr><tr><td>9</td><td>10</td><td>11</td><td>12</td></tr></table>	1	2	3	4	13	14	15	16	5	6	7	8	9	10	11	12	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	<table><tr><td>13</td><td>1</td><td>5</td><td>9</td></tr><tr><td>14</td><td>2</td><td>6</td><td>10</td></tr><tr><td>15</td><td>3</td><td>7</td><td>11</td></tr><tr><td>16</td><td>4</td><td>8</td><td>12</td></tr></table>	13	1	5	9	14	2	6	10	15	3	7	11	16	4	8	12	<table><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	<table><tr><td>14</td><td>6</td><td>2</td><td>1</td></tr><tr><td>10</td><td>15</td><td>5</td><td>3</td></tr><tr><td>11</td><td>9</td><td>16</td><td>4</td></tr><tr><td>13</td><td>12</td><td>8</td><td>7</td></tr></table>	14	6	2	1	10	15	5	3	11	9	16	4	13	12	8	7	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	<table><tr><td>1</td><td>2</td><td>6</td><td>7</td></tr><tr><td>3</td><td>5</td><td>15</td><td>11</td></tr><tr><td>4</td><td>8</td><td>10</td><td>12</td></tr><tr><td>16</td><td>9</td><td>13</td><td>14</td></tr></table>	1	2	6	7	3	5	15	11	4	8	10	12	16	9	13	14
0	0	0	0																																																																																																																																				
1	1	1	1																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	2	3	4																																																																																																																																				
13	14	15	16																																																																																																																																				
5	6	7	8																																																																																																																																				
9	10	11	12																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
13	1	5	9																																																																																																																																				
14	2	6	10																																																																																																																																				
15	3	7	11																																																																																																																																				
16	4	8	12																																																																																																																																				
1	0	0	0																																																																																																																																				
0	1	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	1																																																																																																																																				
14	6	2	1																																																																																																																																				
10	15	5	3																																																																																																																																				
11	9	16	4																																																																																																																																				
13	12	8	7																																																																																																																																				
0	0	0	0																																																																																																																																				
0	0	1	0																																																																																																																																				
0	0	0	0																																																																																																																																				
1	0	0	0																																																																																																																																				
1	2	6	7																																																																																																																																				
3	5	15	11																																																																																																																																				
4	8	10	12																																																																																																																																				
16	9	13	14																																																																																																																																				
(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)																																																																																																																																

- (a) A MD-Template for Mode 1. (b) The corresponding scan order of MD-Template (a).
(c) A MD-Template for mode 0. (d) The corresponding scan order of MD-Template (c).
(e) A MD-Template for mode 3. (f) The corresponding scan order of MD-Template (e).
(g) A MD-Template for mode 4. (h) The corresponding scan order of MD-Template (g).

Figure 5.12 Examples for scan orders

After we subtract the residual block, X_i , by one of the 16 MD-Templates multiplied by the corresponding scaling integer a_i , we also define a new scan order based on the selected MD-Template. For the scan order, any design that reorders zero coefficients to the end of the scanned sequence is applicable. In our work, we use the scan order

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

demonstrated in Figure 5.12. For Mode 0, the scan order is in a row by row, left to right, top to bottom manner. During the scan, if the position labeled with '1' is scanned, it will be skipped first and scan in the same manner when all the '0' labeled positions are scanned. This scan pattern is shown in Figure 5.12 (a) and (b). Similarly, the scan order for Mode 1 is in a column by column, top to bottom, left to right manner with '1' positions reordered as shown in Figure 5.12 (c) and (d). For Mode 3, the scan order is the reflection of the zigzag order with '1' positions reordered, as shown in Figure 5.12 (e) and (f). For Mode 4, the scan order is the zigzag order with '1' positions reordered, as shown in Figure 5.12 (g) and (h). For Mode 2, the scan order of MD-Templates (a) to (b) is the same as in Mode 0, the scan order of MD-Templates (c) to (f) is the same as Mode 1, the scan order of MD-Templates (g) to (k) is the same as Mode 3, and the scan order of MD-Templates (l) to (p) is the same as Mode 4.

(a) A demonstration of H.264 lossless intra coding process.

(b) A demonstration of H.264 lossless intra coding process using MD-Templates.

Figure 5.13 A demonstration of H.264 intra lossless coding with and without MD-Templates

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

In Figure 5.13 we show the comparison of the H.264 intra lossless coding and proposed MD-Template based intra lossless coding. As observed in Figure 5.13 (b), after subtracting a MD-Template multiplied by the corresponding scaling integer a_i , three more zero coefficients are generated; after the mode dependent scan, these three zero coefficients are arranged at the end of the scanned sequence. The resultant sequence fits the entropy coder in H.264 better compared with the sequence in Figure 5.13 (a).

5.3 Algorithms and Analysis

5.3.1 Proposed Algorithms

In this subsection, we give the pseudo-code of the MD-Template based intra lossless coding in Figure 5.14 and Figure 5.15.

Inputs: X_i , $l_{H.264}$

Outputs: The coded bitstream

```

1) for  $a_i = -4:-1, 1:4$ 
    for  $W = \text{MD-Template}(a): \text{MD-Template}(p)$ 
        calculate  $R_i = X_i - a_i W$ ;
        scan  $R_i$  with the corresponding scan order of  $W$ .
        code the scanned  $R_i$  with entropy coders in H.264/AVC and record the length of the derived
        bitstream  $l(a_i, W)$  and add its overhead bits  $H(a_i, W)$  to  $l(a_i, W)$ 
    end
end
2) choose the one with least bits,  $L = \min (H(a_i, W) + l(a_i, W))$ .
3) if  $L < l_{H.264}$ 
    code the residual block with MD-Templates based method
else
    code the residual block with conventional H.264/AVC lossless method
end

```

Figure 5.14 The pseudo-code of Algorithm 1

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

In Figure 5.14, X_i is the residual block belonging to a certain mode, $I_{H.264}$ is the number of bits required to code X_i by H.264 intra lossless (H.264-LS) with CAVLC coder. In Algorithm 1, all the possible combinations of a_i and MD-Templates for each block are tested. Then the coding result with minimal bits, L , is compared with the coded H.264 intra lossless bitstream $I_{H.264}$ and the one with least bits is chosen. However, Algorithm 1 is very time consuming since each combination of a_i and MD-Templates for each block is coded. Therefore, in Figure 5.15, we give a fast algorithm for practical implementation.

In Algorithm 2, not all the combinations of a_i and MD-Templates for each block are coded and compared with the H.264-LS result. For an input residual block X_i , after subtracting the scaled MD-Templates, if more than 1 zero coefficients are generated as compared to X_i , it will be coded to compare the bits length with the H.264-LS result, and decide which method to use. Although Algorithm 2 is much faster, it reduces coding efficiency since not all the combinations of a_i and MD-Templates are coded to compare with the H.264-LS result.

Inputs: $X_i, I_{H.264}$

Outputs: The coded bitstream

```

1) for  $a_i = -4:-1,1:4$ 
    for  $W = \text{MD-Template}(a_i)$ : MD-Template(p)
        calculate  $R_i = X_i - a_i W$ ;
        if  $\| \text{vec}(R_i) \|_0 - \| \text{vec}(X_i) \|_0 \geq 2$ 
            scan  $R_i$  with the corresponding scan order of  $W$ .
            code the scanned  $R_i$  with entropy coders in H.264/AVC and record the length of the derived
            bitstream  $I$ 
            add the corresponding overhead bits  $H$  to  $I$ ,  $L = H + I$ 
            if  $L < I_{H.264}$ 
                code the residual block with MD-Templates based method
                break
            end
        end
    end
    if  $W == \text{MD-Template}(p)$  &  $a_i == 4$ 
        code the residual block with conventional H.264/AVC lossless method
    end
end
end

```

Figure 5.15 The pseudo-code of Algorithm 2

5.3.2 Statistics Before and After Coding with the Proposed Algorithms

Table 5.5 Selection rate comparison

Resolution	Sequences	SR (Algorithm 1)	SR (Algorithm 2)
QCIF	Foreman	81.3	48.4
	Mobile	78.2	49.4
CIF	Tempete	76.6	47.7
	Flowergarden	78.5	51.3
1080P	Crowdrun	68.5	39.3
	Parkrun	65.7	38.8
<i>Average SR</i>		<i>74.8</i>	<i>45.8</i>

To give a more detailed analysis of our method, we define the Selection Rate (SR) as

$$\text{Selection Rate}(\%) = \frac{\#Blocks_{MD-Templates}}{\#Blocks_{All}} \times 100 \quad (5.3)$$

where $\#Blocks_{MD-Templates}$ is the number of blocks chosen to use the MD-Templates to code and $\#Blocks_{All}$ is the total number of blocks in a video.

In Table 5.5, we record the SR for six sequences, each with 100 frames of the YUV 4:2:0 format, coded with Algorithm 1 and Algorithm 2. As we can observe, with Algorithm 1, about 75% of blocks could be further compressed with the proposed MD-Template method. On the other hand, with Algorithm 2, the proposed MD-Template method improves the coding efficiency for 46% of blocks. Anyway, Algorithm 2 still achieves considerable bit saving as recorded in Table 5.8 and Table 5.9.

We further calculate the number of zero coefficients before and after using the MD-Templates in Table 5.6 for the six sequences with different resolutions. As observed, after applying the MD-Templates, 1.47 more zero coefficients are generated on average.

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

Table 5.6 Number of zero coefficients comparison

	Sequences	H.264-LS(CAVLC)	Proposed (Algorithm 2)
QCIF	Foreman	2.25	4.87
	Mobile	1.90	3.78
CIF	Flowergarden	1.37	2.45
	Tempete	1.73	3.02
1080P	Crowdrun	1.37	2.31
	Parkrun	1.26	2.29
Average zero coefficients		1.65	3.12
Average zero coefficients over H.264 Lossless			1.47

In Figure 5.16, we plot the probability distribution of the existing non-zero coefficients versus the scan position with and without using the MD-Templates for the CIF *Tempete* sequence. As observed, without using the MD-Templates, each position almost has the same probability of having non-zero coefficients (the solid curve); with the MD-Templates, the position at the beginning part tends to be more likely to have non-zero coefficients compared with the ending part of the scan position (the dash curve).

Figure 5.16 Probability distribution of non-zero coefficients vs. scan position

5.4 Experimental Results

In this section, five QCIF (144x176) sequences, two CIF (288x352) sequences, five 512x512 images and four HD (1080x1920) sequences are selected to show the effectiveness of the proposed method as well as to provide a comprehensive comparison with the state-of-art intra lossless coding method based on H.264-LS [35][37] with CAVLC entropy coding. The reference software for H.264 lossless coding implementation is JM 16.0 which could be downloaded at [95].

The CAVLC decoding part does not need to change since the MD-Templates are added after CAVLC decoding. However, we need to add some syntax elements in the bitstream for decoding. For each coded block in the bitstream, following the syntax of intra-prediction mode information, 1 bit is used to indicate whether this block is coded with the proposed method. If this block is coded with the standard intra H.264-LS, CAVLC code is followed. If this block is coded with the proposed method, the code of scaling integer a_i (please refer to Table 5.3) and code of MD-Template used for this block (4 bits for each template) are inserted before the CAVLC code. If better entropy coding is used, we can expect even higher compression ratio. CAVLC is used in all the experiments. Here the compression ratio and the bit saving are defined in (2.4) and (2.5).

5.4.1 Number of MD-Templates to Use

Obviously, with more MD-Templates to choose, e.g., 32 MD-Templates or 64 MD-Templates, we have higher probability of getting a block with more zeros and this will give more potential for further compression. However, more MD-Templates will lead to more computation.

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

Table 5.7 Compression efficiency comparison with different MD-Templates numbers

Sequences	H.264-LS (CAVLC)	8 MD-Templates (Algorithm 2)		16 MD-Templates (Algorithm 2)		32 MD-Templates (Algorithm 2)	
	Compression Ratio	Compression Ratio	Saving (%)*	Compression Ratio	Saving (%)*	Compression Ratio	Saving (%)*
Foreman [#]	2.16	2.25	4.02	2.31	6.59	2.31	6.54
Mobile [#]	1.53	1.64	7.14	1.69	9.08	1.68	8.84
Tempete [§]	1.86	1.99	6.35	2.02	8.00	2.02	8.08
Flowergarden [§]	1.76	1.89	6.78	1.93	8.76	1.92	8.46
Crowdrun ^{&}	2.11	2.19	3.65	2.22	5.10	2.21	4.93
Parkrun ^{&}	2.05	2.14	4.18	2.18	5.91	2.17	5.81
Average	1.91	2.02	5.35	2.06	7.24	2.05	7.16

*Bit saving calculated relative to H.264-LS (CAVLC)

[#]150 frames of YUV 4:2:0 format with QCIF (144x176) resolution

[§]150 frames of YUV 4:2:0 format with CIF (288x352) resolution

[&]150 frames of YUV 4:2:0 format with 1080P (1080x1920) resolution

More importantly, we also need more bits to represent these MD-Templates, e.g., 5 bits for 32 MD-Templates and 6 bits for 64 MD-Templates, which will negatively impact the coding efficiency. Therefore, we need to choose a reasonable MD-Templates number. In Table 5.7, we show the coding result with 8 MD-Templates, 16 MD-Templates and 32 MD-Templates for the six sequences. As indicated in Table 5.7, when we increase the number of MD-Templates from 8 to 16, the averaged bit saving increases by about 2%. However, when we further increase the number of MD-Templates from 16 to 32, the bit saving starts to drop. Therefore, we use 16 MD-Templates in our method.

5.4.2 Coding Performance Comparison

In Table 5.8 and Table 5.9, we compare the compression performance among H.264-LS (CAVLC), Heo's method [35], Wei's method [37] and the proposed method with Algorithm 1 and Algorithm 2. All the data for Heo's method and Wei's method are directly extracted from [37] and [35]. If a sequence is used for training, it will not be included in the testing results. As indicated in the bold numbers, the proposed method with fast coding algorithm (Algorithm 2) achieves about 7.2% bit saving compared with H.264-LS on average. The proposed MD-Templates based method with Algorithm 2

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

gives better performance compared with Heo's method and Wei's method, for about 0.4% and 0.8% bit saving improvement respectively. If a longer encoding process is allowed, Algorithm 1 could give even higher performance, about 2.4% and 3% bit saving improvements on average compared with Heo's method and Wei's method, respectively.

Table 5.8 Performance comparison of Heo's[35] method and the proposed method

Sequences	H.264-LS (CAVLC)	Heo [35]		Proposed (Algorithm 2)		Proposed (Algorithm 1)	
	Compression Ratio	Compression Ratio	Saving (%) [*]	Compression Ratio	Saving (%) [*]	Compression Ratio	Saving (%) [*]
Foreman [#]	2.16	2.27	4.93	2.31	6.59	2.36	8.60
Mobile [#]	1.53	1.71	10.67	1.69	9.08	1.71	10.88
Tempete ^{\$}	1.86	2.01	7.30	2.02	8.00	2.07	10.24
Flowergarden ^{\$}	1.76	1.96	9.88	1.93	8.76	1.97	10.89
Crowdrun ^{&}	2.11	2.19	3.58	2.22	5.10	2.27	7.12
Parkrun ^{&}	2.05	2.16	4.85	2.18	5.91	2.23	8.14
Average	1.91	2.05	6.87	2.06	7.24	2.10	9.31

^{*}Bit saving calculated relative to H.264-LS (CAVLC)

[#]150 frames of YUV 4:2:0 format with QCIF (144x176) resolution

^{\$}150 frames of YUV 4:2:0 format with CIF (288x352) resolution

[&]150 frames of YUV 4:2:0 format with 1080P (1080x1920) resolution

Table 5.9 Performance comparison of Wei's[37] method and the proposed method

Sequences	H.264-LS (CAVLC)	Wei [37]		Proposed (Algorithm 2)		Proposed (Algorithm 1)	
	Compression Ratio	Compression Ratio	Saving (%) [*]	Compression Ratio	Saving (%) [*]	Compression Ratio	Saving (%) [*]
Akiyo [#]	2.53	2.74	7.81	2.74	7.63	2.80	9.79
Coastguard [#]	2.13	2.25	5.74	2.29	6.83	2.34	8.80
Foreman [#]	2.23	2.43	8.23	2.41	7.62	2.47	9.79
Mobile [#]	1.45	1.57	8.16	1.58	8.49	1.62	10.75
News [#]	2.25	2.43	7.50	2.46	8.62	2.52	10.76
Avion [@]	2.09	2.26	7.77	2.25	7.13	2.31	9.33
Baboon [@]	1.24	1.35	8.77	1.35	8.53	1.39	10.70
House [@]	1.88	2.03	7.75	2.05	8.25	2.10	10.31
Lena [@]	1.78	1.86	4.47	1.92	7.12	1.96	9.24
Peppers [@]	1.63	1.66	2.17	1.72	5.02	1.75	7.04
InToTree ^{&}	1.73	1.82	5.05	1.83	5.29	1.87	7.27
TownCross ^{&}	1.62	1.67	3.33	1.71	5.42	1.75	7.41
Average	1.88	2.00	6.40	2.03	7.16	2.07	9.27

^{*}Bit saving calculated relative to H.264-LS (CAVLC)

[#]30 frames of YUV 4:4:4 format with QCIF (144x176) resolution

[@]Image with 512x512 resolution

[&]10 frames of YUV 4:4:4 format with 1080P (1080x1920) resolution

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

5.4.3 Computational Complexity

For intra lossless coding, the most time consuming part is the CAVLC module. The encoder needs to code each block nine times to decide which mode this block belongs to, thus it requires to conduct CAVLC for nine times. In Algorithm 1, all the possible combinations of a_i and MD-Templates for each block are tested. Then the combination with minimal bits, L , is compared with the coded intra H.264-LS bitstream $I_{H.264}$ and the one with least bits is chosen. Since we have 8 values for a_i and 16 MD-Templates, we need to code (with CAVLC) each block $8 \times 16 = 128$ times, which takes almost 13 times more computation than that of the standard intra H.264-LS coding (to conduct CAVLC for 9 times). For Algorithm 2, for most of cases, CAVLC is conducted once to decide whether the residual block X_i is coded with the MD-Template based method or not. In Table 5.10, we show the distribution of times of CAVLC used in Algorithm 2. The computer platform is Intel X5560 CPU @ 2.80GHz, 16GB RAM and 64-bit Operation System. We just use *clock()* function in the reference software to calculate the time.

As summarized in Table 5.10, on average, for 62% of residual blocks, Algorithm 2 requires to conduct CAVLC only once to decide whether to code with the MD-Template based method or not. For 15% of residual blocks, Algorithm 2 requires to conduct CAVLC more than twice to decide whether to code with the MD-Template based method. Therefore, the increased computation by Algorithm 2 is very limited. In Figure 5.17, we show the computation time increase comparison between intra H.264-LS and Algorithm 2. Sequences *Akiyo*, *News* and *Coastguard* are QCIF sequences with 30 frames. Sequences *Tempete* and *Flowergarden* are CIF sequences with 30 frames. *House* and *Lena* are 512x512 images. As indicated in Figure 5.17, the coding time is increased modestly by 26.8% on average with Algorithm 2.

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

Table 5.10 Probabilities of times of CAVLC used in Algorithm 2

	Sequences/Images	Times of CAVLC used		
		Once	Twice	More than Twice
QCIF	Akiyo	0.508	0.250	0.242
	News	0.655	0.170	0.175
	Coastguard	0.648	0.257	0.095
CIF	FlowerGarden	0.612	0.219	0.169
	Tempete	0.582	0.254	0.164
Image	House	0.704	0.213	0.083
	Lena	0.628	0.238	0.134
<i>Average</i>		<i>0.620</i>	<i>0.229</i>	<i>0.151</i>

Figure 5.17 Percentage of coding time increase of Algorithm2 as compared to H.264-LS

5.5 Concluding Remarks

The entropy coding module in the H.264/advanced video coding (AVC) standard was originally designed for lossy video coding and did not yield adequate performance for lossless video coding. In this work, we explore the statistical correlation of the intra prediction residual for the 5 intra prediction modes. By designing templates to be subtracted for each intra prediction modes, more zero entries are generated. By further

Chapter 5. Mode Dependent Templates and Scan Order for Intra Lossless Coding

designing the scan order for each template, the beginning part of the scanned coefficient sequence has higher probability of having non-zero coefficients compared with the ending part. In this way, the resultant sequence is more suitable for entropy coding in H.264/AVC. With insignificant computational increase, the proposed method with fast coding algorithm (Algorithm 2) achieves about 7.2% bit saving compared with intra H.264-LS. If the coding condition allows a longer encoding process, the Algorithm 1 that codes all the possible combinations of a_i and MD-Templates will give even better performance, about 9.3% of bit saving compared with intra H.264-LS.

Chapter 6

Summary and Future Work

6.1 Summary

To improve the performance of current video coding scheme, this thesis has presented approaches which can better explore the redundancies in the following three aspects: a scheme for temporal redundancy removal without motion estimation; a transform for better spatial redundancy removal for motion compensated residual and a scheme for further statistical redundancy removal for intra lossless video coding.

The major technical contributions of this project have been highlighted in Section 1.3, and in this section, we will give a summary of the research that has been performed in this project, with links back into the preceding chapters.

6.1.1 2D-SVD for Non-Motion Estimation Based Video Coding

The state of the art video codecs such as H.264/AVC (as well as other motion-estimation (ME) based codecs) lead to high computational complexity. This often results in longer processing time in a software codec, shorter battery-life for mobile devices and high cost of implementation. In Chapter 3, we have proposed a temporal redundancy

Chapter 6. Summary and Future Work

removal methodology for video coding based upon 2D-SVD (two-dimensional Singular Value Decomposition). By exploring the energy compaction property of 2D-SVD coefficient matrices, we have designed a video coding scheme without ME. In essence, the most common (stable) content of a group of blocks along the time axis is represented by two 2D-SVD transformation matrices, while the varying content for each individual block is captured by a highly energy compact coefficient matrix. In other words, with 2D-SVD, we use the SVD-like transformation matrices to represent (and to be transmitted once) the most common visual content in video, as a way to exploit temporal redundancy without resorting to ME.

In comparison with the state of the art intra frame codec, H.264/AVC Intra, which also provides frame independency for decoding, our codec outperforms significantly in terms of average BD-Bitrate. Compared with other non-ME based low complexity codecs, our codec gives the higher coding efficiency in terms of average BD-Bitrate. On the other hand, compared with H.264/AVC inter coding with optimized configuration, the proposed 2D-SVD approach achieves about 60% computational complexity reduction with 130% BD-Bitrate loss due to the lack of motion estimation. Moreover, upon the successful transmission of the Group Information (GI), which only occupies a small number of bits for a Group Of Pictures (GOP), our codec achieves independent frame decoding which facilitates video manipulating/editing flexibility. The experiments with 14 video sequences have confirmed the effectiveness of the proposed codec and the suitability for scenarios such as mobile video phone and surveillance video coding, where low complexity compression with high coding efficiency are required.

6.1.2 Rotated Orthogonal Transform for MC-residual Coding

In video coding, Motion Compensation residual (MC-residual) has different spatial characteristic compared with a natural image, and this makes the current compression schemes less efficient for MC-residual coding. Moreover, since such residual is visual content related, it is desirable to derive a new transform (non-DCT) that is adaptive to the local characteristic of residual to yield coefficient compaction, while keeps the necessary side information and computation small.

In Chapter 4, we have proposed an L1-Norm minimization based algorithm to generate a Rotated Orthogonal Transform (ROT) for coding MC-residual frames. By converting the orthogonal constraints into convex constraints for the problem formulation, we are able to conduct the gradient-based search for the optimization. By setting the DCT matrix as the starting transform matrix, we obtain a better matrix in terms of L1-Norm minimization. We further design a weighting matrix by setting proper weights for the transformed coefficients in line with zigzag scan order. In this way, the optimization result is customized to the entropy coding available. In addition, by exploring inter-frame dependency and local temporal activity, transmission of substantial side information is avoided.

Analysis and overall experimental results have confirmed that using the ROT derived from the proposed algorithm, we can improve the compression efficiency of the MC-residual for videos containing diversified visual content and motion with small computational overhead. The better outperformance of the proposed transform for high/complex temporal activity video is highly desirable in video coding; because this helps to improve the R-D performance at the most difficult scenario and alleviate

Chapter 6. Summary and Future Work

quality fluctuation throughout the frames (smaller quality fluctuation during the encoding/streaming has positive impact on the perceived quality by the viewers). In our future work, scan order, entropy coder and error resilience for the ROT coefficients will be studied towards further performance improvement.

6.1.3 MD-Templates and Scan Orders for H.264 Intra Lossless coding

The entropy coding module in the H.264/ AVC standard is originally designed for lossy video coding and does not yield adequate performance for lossless video coding.

In Chapter 5, we have explored the statistical correlation of the intra prediction residual for the 5 intra prediction modes. By designing templates to be subtracted for each intra prediction modes, more zero entries are generated. By further designing the scan order for each template, the beginning part of the scanned coefficient sequence has higher probability of having non-zero coefficients compared with the ending part. In this way, the resultant sequence is more suitable for entropy coding in H.264/AVC. With insignificant computational increase, the proposed method with fast coding algorithm (Algorithm 2) achieves about 7.2% bit saving compared with intra H.264-LS. If the coding condition allows a longer encoding process, the Algorithm 1 that codes all the possible combinations of a_i and MD-Templates will give even better performance, about 9.3% of bit saving compared with intra H.264-LS.

6.2 Future work

From the review of both theoretical and experimental results of current work and the tracking of the latest development of the relevant technology, there are several

Chapter 6. Summary and Future Work

interesting avenues for further research as the possible extension of the project work performed so far.

Firstly, the proposed 2D-SVD based coding scheme uses fixed GOP size which cannot deal with scene change well. We can incorporate a simple yet effective scene change detection algorithm, such as method that combined with color histogram and pixel difference proposed in [125], to make the coding scheme more practical. Moreover, 2D-SVD technology in Chapter 3 could also be extended to video categorization, video retrieval, saliency detection and etc. For tasks such as video categorization and video retrieval, the key step is to extract a set of features to represent the video. Since videos contain both spatial and temporal information, video feature extraction is still a challenge topic compared with image feature extraction. As we shown in Figure 3.3, after 2D-SVD, the most common content of a group of blocks/frames along the time axis is represented by two 2D-SVD transformation matrices, Y_l and Y_r , while the varying content for each individual block/frame is captured by a group of coefficient matrices $M_1, \dots, M_n$. These two components effectively represent the temporal feature and spatial feature of a video. Thus 2D-SVD coding could be a new tool not only for video compression but also for tasks related to video representation.

Secondly, in Chapter 4, the Rotated Orthogonal Transform (ROT) based residual coding only uses one orthogonal basis, Y , derived from (6.1) (the same as (4.2)) .

$$\begin{aligned} f(Y) &= \min_Y \sum_i \| \text{vec}(Y^T X_i Y) \|_1 \\ \text{s.t. } Y^T \cdot Y &= I \end{aligned} \quad (6.1)$$

However, we can further relax this problem to get a more sparse result by separating the left and right transforms into two different transforms. Then the problem is stated as

Chapter 6. Summary and Future Work

follows:

$$f(Y_l, Y_r) = \min_{Y_l, Y_r} \sum_i \| \text{vec}(Y_l^T X_i Y_r) \|_1 \quad (6.2)$$

$$s.t \quad Y_l^T \cdot Y_l = I; \quad Y_r^T \cdot Y_r = I$$

Although we need to transmit two sets of rotation degrees, which will increase the overhead of the encoded bitstream, we give more degrees of freedom for the transformation matrix. Therefore, we can expect that the transformed coefficient will be sparser and give more room for compression.

Moreover, the ROT is not only limited to be used for MC-Residual frame. Actually for frames with less autocorrelation, e.g. intra predicted frame, the enhancement layer in scalable video coding, the compensation signal in multi-view video coding and the texture signal after total-variation transformation. Applying ROT to these signals is also worth researching.

Lastly, the lossless coding component still shares the same entropy coders with lossy coding part in HEVC. The major difference between HEVC and H.264 in intra coding is that HEVC introduces more intra prediction modes for more block sizes. Since the methodology, e.g. templates learning and implementation, of the proposed method is independent of number of prediction modes and block size, we could extend the proposed MD-Template based approach to HEVC standard with little modification under the same framework.

Furthermore, in the current work, the proposed MD-Template is only used for intra lossless coding. It is also possible to implement it for inter lossless coding, if we can classify the inter-predicted (e.g. motion compensated) residual blocks in to several classes (like the 9 modes for intra prediction). One possible implementation is to classify the inter predicted residual to several classes according to the length and

Chapter 6. Summary and Future Work

direction of their corresponding motion vectors and explore the correlation within each class for further statistical redundancy removal.

On the other hand, since the human eye is the ultimate receiver of the vast majority of signals (being natural or computer generated) after acquisition, processing and transmission, incorporating proper human perception characteristics into video coding performance evaluation makes built systems user-oriented. The well-known perceptual metrics proposed in recent ten years include SSIM Structural Similarity [111], Visual Information Fidelity [112], Visual Signal-to-Noise Ratio [113] and Most Apparent Distortion [114]. We will explore the use of perceptual criteria in video coding as the next step of research.

References

References

- [1] L. Chiariglione. “The development of an integrated audiovisual coding standard: MPEG.” *Proceeds of the IEEE*, vol. 83, no. 2, pp. 151–157, 1995.
- [2] Information Technology—Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/s—Part 2:Video. *ISO/IEC 11 172–2*, 1993.
- [3] Information Technology—Generic Coding of Moving Pictures and Associated Audio Information: Video. *ISO/IEC 13 818–2*, 1996.
- [4] J. Watkinson, The MPEG handbook: MPEG-I, MPEG-2. MPEG-4, *Focal Press*, Boston, 2001.
- [5] T. Wiegand, G. J. Sullivan, G. Bjntegaard, A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Trans. Circuits Syst. Video Technol*, vol. 13, no. 7, pp. 560–576, 2003.
- [6] M. Sarwer, Q. Wu, “Adaptive Variable Block-Size Early Motion Estimation Termination Algorithm for H.264/AVC Video Coding Standard,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 8, pp. 1196–1201, 2009.
- [7] L. Po, C. Ting, K. Wong, K. Ng, “Novel Point-Oriented Inner Searches for Fast Block Motion Estimation,” *IEEE Trans. Multimedia*, vol. 9, no. 1, pp. 9–15, 2007.
- [8] N. Ahmed, T. Natarajan, K. R. Rao, “Discrete cosine transform,” *IEEE*

References

- Transactions on Computer.*, vol. C-23, pp. 90-93, 1974.
- [9] G.Calvagno, G.A.Mian, R.Rinaldo, "Computation of the coding gain for subband coders," *IEEE Trans. on Commun.*, vol. 44, no. 4, pp. 475-487, 1996.
- [10] C. X. Zhang, L. Yu, J. Lou, W. K. Cham, J. Dong, "The Technique of Prescaled Integer Transform: Concept, Design and Applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 1, pp. 84-97, 2008.
- [11] Y. Huang, B. Hsieh, S. Chien, S. Ma, L. Chen, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol.16, no.4, pp.507–522, 2006.
- [12] H. S. Malvar et al., "Low-Complexity Transform and Quantization in H.264/AVC", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 598–603, 2003.
- [13] A. Bahari, T. Arslan, A. Erdogan, "Low-Power H.264 Video Compression Architectures for Mobile Communication," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 9, pp. 1251–1261, 2009.
- [14] C. Chen, S. Chien, Y. Huang, T. Chen, T. Wang, L. Chen, "Analysis and Architecture Design of Variable Block-Size Motion Estimation for H.264/AVC," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 53, no. 3, pp. 578–593, 2006.
- [15] A. Asif "A practical software-only noncausal predictive video codec for low-bit-rate multimedia applications" *Canadian Journal of Electrical and Computer Engineering* , vol. 30, no. 2, pp. 63-71, 2005.
- [16] T. V. Dung, T.Q. Nguyen, "Quality Enhancement for Motion JPEG Using Temporal Redundancies," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 5,

References

- 2008.
- [17] S. Kamath, J. R. Jackson, “Low-Bit Rate Motion JPEG Using Differential Encoding,” Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, vol. 2, pp. 1723-1726, 2004.
- [18] J. Meessen, C. Parisot, X. Desurmont, J.-F. Delaigle, “Scene Analysis for Reducing Motion JPEG 2000 Video Surveillance Delivery Bandwidth and Complexity,” *International Conference on Image Processing*, vol. 1, pp 77-80, 2005.
- [19] Y. J. Chiu, T. Berger, “A software-only video codec using pixelwise conditional differential replenishment and perceptual enhancements,” *IEEE Trans. Circuits Syst. Video Technol*, vol. 9, no. 3, pp. 438–450, 1999.
- [20] Context-Adaptive VLC (CVLC) Coding of Coefficients, document JVTC028.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/SG16m, May 2002.
- [21] D. Marpe, H. Schwarz, T. Wiegand, “Context-based adaptive binary arithmetic coding in the H.264/AVC video compression,” *IEEE Trans. Circuits Syst. Video Technol.* vol.13, no.7, pp. 620–636, 2003.
- [22] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, G. Sullivan, “Rate-constrained coder control and comparison of video coding standards,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 688–703, 2003.
- [23] E. Yang, X. Yu, “Rate Distortion Optimization for H.264 Interframe Coding: A General Framework and Algorithms,” *IEEE Trans. Image Process.*, vol. 16, no. 7, pp. 1774–1784, 2007.

References

- [24] “Lossless Video Compression for Archives: Motion JPEG2k and Other Options,” <http://www.media-matters.net/docs/WhitePapers/WPMJ2k.pdf>
- [25] D. Clunie, “Lossless Compression of Grayscale Medical Images - effectiveness of Traditional and State of the Art Approaches,” in *Proc. SPIE (Medical Imaging)*, vol. 3980, pp.74-84, 2000.
- [26] Y.L. Lee, K.H. Han, G.J. Sullivan, “Improved lossless intra-coding for H.264/MPEG-4 AVC,” *IEEE Trans. Image Process.*, vol. 15, no. 9, pp. 2610–2615, 2006.
- [27] Draft Text of H.264/AVC Fidelity Range Extensions Amendment to ITUT Rec. H.264, document JVT-L047.doc, Joint Video Team of ISO/IEC 14496-10 AVC, ISO/IEC JTC1/SC29/WG11, and ITU-T Q.6/SG16, Jul. 2004.
- [28] G.J. Sullivan, P. Topiwala, A. Luthra, “The H.264/AVC standard: Overview and introduction to the fidelity range extensions,” in *Proc. SPIE Conf., Special Session Adv. New Emerg. Standard: H.264/AVC*, Denver, CO, pp. 454–474, Aug. 2004.
- [29] Intra-Lossless Coding and QP Range Selection, document JVT-C023.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, May 2002.
- [30] Lossless Intra-Coding for Improved 4:4:4 Coding in H.264/MPEG-4 AVC, document JVT-P016.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Jul. 2005.
- [31] Complexity of the Proposed Lossless Intra, document JVT-Q035r1.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Oct. 2005.
- [32] Draft Text of H.264/AVC Advanced 4:4:4 Profile Amendment to ITU-T Rec. H.264|ISO/IEC 14496-10 AVC, document JVT-Q209.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Oct. 2005.

References

- [33] Y.W. Huang et al., “Analysis, Fast Algorithm, and VLSI Architecture Design for H.264/AVC Intra Frame Coder,” *IEEE Trans. Circuits Syst. Video Technol.*, vol.15, no.3, pp.378-401, 2005.
- [34] J.H. Nam, D. Sim, “Lossless video coding based on pixel-wise prediction,” *Multimedia Syst.*, vol. 14, pp. 291–298, 2008.
- [35] J. Heo, Y.S. Ho, “Efficient level and zero coding methods for H.264/ AVC lossless intra coding,” *IEEE Signal Process. Lett.*, vol. 17, no.1, pp. 87–90, 2010 .
- [36] G. Bjontegaard, “Calculation of average PSNR differences between rd-curves,” *VCEG Contribution VCEG-M33*, April 2001
- [37] S.T. Wei, C.W. Tien, B.D. Liu, J.F. Yang, “Adaptive Truncation Algorithm for Hadamard-Transformed H.264/AVC Lossless Video Coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 5, pp. 538-549, 2011.
- [38] Information Technology – JPEG 2000 Image Coding System, Part 1: Core Coding System, *ISO/IEC 15444-1:2000 (ISO/IEC JTC/SC 29/WG 1 N1646R, March 2000)*
- [39] Information Technology – JPEG 2000 Image Coding System, Part 3: Motion JPEG 2000, *ISO/IEC 15444-3:2002*
- [40] M. Ouaret, F. Dufaux, T. Ebrahimi, “On comparing JPEG2000 and Intraframe AVC,” *Proceedings of SPIE, Applications of Digital Image Processing XXIX*, Aug, 2006.
- [41] D. Marpe, V. George, H. L. Cycon, “Performance evaluation of Motion-JPEG2000 in comparison with H.264/AVC operated in intra coding mode,” *Wavelet Applications in Industrial Processing, Proceedings of SPIE*, Oct, 2003.
- [42] Y. L. Law, T. Q. Nguyen, “Motion wavelet difference reduction (MWDR) video

References

- codec,” *International Conference on Image Processing*, vol. 4, pp 2303-2306, 2004.
- [43] M. Wu, G.G. Hua, C. W. Chen, “Syndrome-Based Light-Weight Video Coding for Mobile Wireless Application,” *IEEE International Conference on Multimedia and Expo*, pp.2013-2016, 2006.
- [44] D. Slepian, J. Wolf, “Noiseless Coding of correlated information sources,” *IEEE Trans. Inform. Theory*, vol. 19, no. 4, pp. 471–480, 1973.
- [45] A. Wyner, J. Ziv, “The rate-distortion function for source coding with side information at the decoder,” *IEEE Transactions on Information. Theory*, vol. 22, no. 1, pp. 1–10, 1976.
- [46] A. Aaron, R. Zhang, B. Girod, “Wyner-Ziv coding of motion video,” *Proc. Asilomar Conf. Signals, Syst. Comput.*, vol. 1. pp. 240–244, 2002.
- [47] L.M. Liu, Z. Li, E.J. Delp, “Efficient and Low-Complexity Surveillance Video Compression Using Backward-Channel Aware Wyner–Ziv Video Coding,” *IEEE Trans. Circuits Syst. Video Technol*, vol. 19, no. 4, pp. 453-465, 2009.
- [48] F. Chen, K. Pang, “The optimal transform of motion-compensated frame difference images in a hybrid coder,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol.40, no.6, pp. 393–397, 1993.
- [49] W. Niehsen, M. Brunig, “Covariance analysis of motion-compensated frame differences,” *IEEE Trans. Circuits Syst. Video Technol.*, vol.9, no.4, pp. 536–539, 1999.
- [50] E. Candes, D. Donoho, “Curvelets, multiresolution representation, and scalinglaws.” *Wavelet Applications in Signal and Image Processing VIII. Proc.*

References

- SPIE*, 2000.
- [51] M. Do, M. Vetterli, “The contourlet transform: An efficient directional multiresolution image representation.” *IEEE Trans. Image Process.* vol. 14, no. 12, pp. 2091 – 2106, 2005.
- [52] E. Le Pennec, S. Mallat, “Sparse geometric image representations with bandelets.” *IEEE Trans. Image Process.* vol. 14, no. 4, pp. 423–438, 2005.
- [53] G. Peyre, S. Mallat, “Discrete bandelets with geometric orthogonal filters.” in *Proc. IEEE Int. Conf. Image Processing*, pp. 65–68, 2005.
- [54] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli and P. Dragotti, “Directionlets: Anisotropic multidirectional representation with separable filtering”. *IEEE Trans. Image Process.*, vol.15, no. 7, pp. 1916-1933, 2005.
- [55] G. J. Sullivan, J. R. Ohm, “Recent Developments in Standardization of High Efficiency Video Coding (HEVC),” *SPIE Applications of Digital Image Processing XXXIII, Proc. SPIE*, vol. 7798, pp.7798-30, 2010.
- [56] W. Ding, F. Wu, X. Wu, S. Li, H. Li, “Adaptive directional lifting-based wavelet transform for image coding,” *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 416–427, 2007.
- [57] C. L. Chang, B. Girod, “Direction-adaptive discrete wavelet transform for image compression,” *IEEE Trans. Image Process.*, vol. 16, pp. 1289–1302, 2007.
- [58] B. Zeng, J. Fu, “Directional discrete cosine transforms—a new framework for image coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 3, pp. 305–313, 2008.

References

- [59] F. Kamisli, J. Lim, “1-D Transforms for the Motion Compensation Residual,” *IEEE Trans. Image Process.*, vol. 20, no. 4, pp. 1036 – 1046, 2011.
- [60] H. Xu, J. Xu, F. Wu, “Lifting-based directional DCT-like transform for image coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1325–1335, 2007.
- [61] Y. Ye, M. Karczewicz, “Improved H.264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning,” in *Proc. IEEE Int. Conf. Image Processing*, pp. 2116–2119, 2008.
- [62] C. L. Chang, B. Girod, “Direction-adaptive partitioned block transform for image coding,” in *Proc. IEEE Int. Conf. Image Processing*, pp. 145–148, 2008.
- [63] M. Makar, S. Tsai, B. Girod, “Direction-Adaptive Partitioned Block Transform for Color Image Coding”, *IEEE Trans. Image Process.*, vol. 19, no. 7, pp. 1740–1755, 2010.
- [64] "Samsung's Response to the Call for Proposals on Video Compression Technology", JCT-VC Document JCTVC-A124, April 2010.
- [65] S.T. Wei, C.W. Tien, B.D. Liu, J.F. Yang, “Adaptive Truncation Algorithm for Hadamard-Transformed H.264/AVC Lossless Video Coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 5, pp 538-549, 2011.
- [66] J.R. Ding, J.F. Yang, “Adaptive entropy coding with (5, 3) DWT for H.264 lossless image coding,” in *Proc. IEEE TENCON*, pp. 1–4, 2007.
- [67] J.R. Ding, J.Y. Chen, F.C. Yang, J.F. Yang, “Two-layer and adaptive entropy coding algorithms for H.264-based lossless image coding,” in *Proc. IEEE ICASSP*, pp. 1369–1372, 2008.

References

- [68] Q. Zhang, Y. Dai, C. J. Kuo, “Lossless video compression with residual image prediction and coding (RIPC),” in *Proc. IEEE ISCAS*, pp. 617–620, 2009.
- [69] Y. C. Wei, “Performance improvements for intra-prediction in advanced video coding,” Ph.D. dissertation, Dept. Electric. Eng., Natl. Cheng Kung Univ., Tainan, Taiwan, 2008.
- [70] S. H. Kim, J. W. Kang, C. J. Kuo, “Improved H.264/AVC Lossless Intra Coding With Two-Layered Residual Coding (TRC),” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 1005-1010, 2011
- [71] S. H. Kim, J. Heo, Y.-S. Ho, “Efficient entropy coding scheme for H.264 lossless video coding,” *Signal Process.: Image Commun.*, vol. 25, no. 9, pp. 687–696, 2010.
- [72] D.P. Bertsekas, A. Nedic, A.E. Ozdaglar, *Convex Analysis and Optimization*, Athena Scientific, Belmont, MA, 2003.
- [73] M. Elad, M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Trans. Image Process.*, vol. 54, no. 12, pp. 3736-3745, 2006
- [74] R. Grosse, R. Raina, H. Kwong, A. Y. Ng, “Shift-invariant sparse coding for audio classification,” *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007
- [75] J. Mairal, F. Bach, J. Ponce, G. Sapiro, A. Zisserman, “Discriminative learned dictionaries for local image analysis,” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2008
- [76] J. Mairal, M. Elad, G. Sapiro, “Sparse representation for color image restoration,”

References

- IEEE Trans. Image Process.*, vol.17 no.1, pp. 53–69, 2008
- [77] N. Kwak, “Principal component analysis based on L1-norm maximization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.3, no.9, pp. 1672–1680, 2008
- [78] D. D. Lee, H. S. Seung, “Algorithms for non-negative matrix factorization.” *Advances in Neural Information Processing Systems(NIPS)*, pp.556–562, 2001
- [79] C. Ding, H. Huang, D. Luo, “Tensor reduction error analysis – applications to video compression and classification,” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2008.
- [80] K. Inoue, K. Urahama, “Equivalence of non-iterative algorithms for simultaneous low rank approximations of matrices,” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 154-159, 2006.
- [81] C. Ding, J. Ye, “Two-dimensional singular value decomposition (2dsvd) for 2d maps and images,” *SIAM Int’l Conf. Data Mining*, pp. 32–43, 2005.
- [82] J. Ye, “Generalized low rank approximations of matrices,” *International Conference on Machine Learning*, 2004.
- [83] J. Liu, S.C. Chen, Z. H. Zhou, X.Y. Tan, “Generalized Low-Rank Approximations of Matrices Revisited,” *IEEE Transactions on Neural Network*, vol. 21, no. 4, pp. 621-632, 2010.
- [84] J. F. Yang, C. L. Lu, “Combined techniques of singular value decomposition and vector quantization for image coding,” *IEEE Trans. Image Process.*, vol. 4, pp. 1141–1146, Aug. 1995.
- [85] H. Ochoa, K.R. Rao, “A Hybrid DWT-SVD Image-Coding System (HDWTSVD)

References

- for Monochromatic Images,” *WSEAS Transaction on Circuit and Systems*, vol. 4, pp. 419 – 424, 2005
- [86] Y. Linde, A. Buzo, R. M. Gray, “An algorithm for Vector Quantizer design”, *IEEE Trans. Communications.*, vol. 28, no. 1, pp. 85-94. 1980.
- [87] H. Schwarz, D. Marpe, T. Wiegand, “Analysis of Hierarchical B Pictures and MCTF,” in *Proc. Int. Conf. Multimedia and Expo*, pp. 1929–1932, 2006.
- [88] C. H. Fu, Y. L. Chan, T. P. Ip, W. C. Siu, “New Architecture for MPEG Video Streaming System With Backward Playback Support,” *IEEE Trans. Image Process.*, vol. 16, no. 9, pp. 2169 – 2183, 2007
- [89] M.-Y. Ma, O. C. Au, S. H. Chan, L.W. Guo “Error Resilient Video Coding Using B Pictures in H.264,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 10, 2009
- [90] C. H. Fu, Y. L. Chan, W. C. Siu, “Efficient Reverse-Play Algorithms for MPEG Video With VCR Support,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 1, pp. 19 – 30, 2006
- [91] B. Girod, A. Aaron, S. Rane, D. Rebollo-Monedero, “Distributed video coding,” in *Proc. IEEE: Special Issue Advances Video Coding and Delivery*, vol. 93, no. 1, pp. 71–83, 2005.
- [92] L. M. Liu, Z. Li, E.J. Delp, “Efficient and Low-Complexity Surveillance Video Compression Using Backward-Channel Aware Wyner–Ziv Video Coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 4, pp. 453-465, 2009.
- [93] Y. C. Tourapis, A. M. Tourapis, “Fast motion estimation within the H.264 codec,” in *Proc. IEEE Int. Conf. Multimedia Expo*, pp. 517–520, 2003.
- [94] R. Li, B. Zeng, M. L. Liou, “A new three-step search algorithm for block motion

References

- estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, 1994
- [95] H.264/AVC Reference Software, JM16.0, <http://iphome.hhi.de/suehring/tml/>
- [96] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, M. Ouaret, “The DISCOVER Codec: Architecture, Techniques and Evaluation,” *Picture Coding Symposium (PCS'07)*, 2007
- [97] R. Martins, C. Brites, J. Ascenso, F. Pereira, “Refining Side Information for Improved Transform Domain Wyner–Ziv Video Coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 9, 2009
- [98] D. L. Donoho, M. Elad, “Maximal sparsity representation via l_1 minimization,” *Proceedings of the National Academy of Sciences, USA*, vol. 100, no. 5, pp. 2197–2202, 2003.
- [99] “Adaptive Quantization Matrix Selection on KTA Software” VCEG-AD06, Oct, 2006.
- [100] M. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. New York: Academic, 1982.
- [101] B. Tao, M. T. Orchard, “Gradient-based residual variance modeling and its applications to motion-compensated video coding,” *IEEE Trans. Image Process.*, vol. 10, no. 1, pp. 24–35, 2001.
- [102] J. Vaisey, A. Gersho, “Image compression with variable block size segmentation,” *IEEE Trans. Signal Process.*, vol. 40, no. 8, pp. 2040–2060, 1992.
- [103] A. Tanizawa, T. Chujoh, “Adaptive Quantization Matrix Selection,” ITU-T Q.6/SG16 VCEG, D.266, 2006.

References

- [104] Z. Gu, W. Lin, B. S. Lee, C. T. Lau, “Adaptive Orthogonal Transform for Motion Compensation Residual in Video Compression”, in *International Conference on Multimedia Modeling*, 2011, pp. 40-50.
- [105] Y.H. Huang, K.L. Chung, “Texture- and Multiple-Template-Based Algorithm for Lossless Compression of Error-Diffused Images,” *IEEE Trans. Image Process.*, vol. 16, no. 5, pp.1258-1268, 2007.
- [106] M. Paul, W. Lin, C. T. Lau, B.S. Lee, “Direct Inter-Mode Selection for H.264 Video Coding using Phase Correlation,” *IEEE Trans. Image Process.*, vol. 20, no. 2, pp. 461-473, 2011.
- [107] O.G. Sezer, R.A. Cohen, A. Vetro, “Robust Learning of 2-D Separable Transforms for Next-Generation Video Coding,” in *Proc. DCC*, pp.63-72, 2011.
- [108] Y. Ye, M. Karczewicz, “Improved Intra Coding,” ITU-T Q.6/SG11 VCEG, VCEG-AG11, Shenzhen, China. Oct, 2007.
- [109] M. Budagavi, M. Zhou, “Orthogonal MDDT and Mode Dependent DCT,” ITU-T Q.6/SG16 VCEG, VCEG-AM20_r1, Kyoto, Japan, Jan. 2010.
- [110] H.Q. Zeng, K.K. Ma, C.H. Cai, “Hierarchical Intra Mode Decision for H.264/AVC,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 6, pp. 907-912, 2010.
- [111] Z. Wang, A. Bovik, H. Sheikh, E. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Trans. Image Process.*, vol. 13 no. 4, pp. 600–612, 2004.
- [112] H. Sheikh, A. Bovik, “Image information and visual quality,” *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 430–444, 2006.

References

- [113] D. Chandler, S. Hemami, “VSNR: A wavelet-based visual signal-to-noise-ratio for natural images,” *IEEE Trans. Image Process.*, vol. 16, no. 9, pp. 2284–2298, 2007.
- [114] E. Larson, D. Chandler, “Most apparent distortion: full-reference image quality assessment and the role of strategy,” *Journal of Electronic Imaging*, vol. 19, no. 1, pp. 011006:1–21, 2010.
- [115] G. K. Wallace, “The JPEG still-picture compression standard,” *Commun. ACM*, vol. 34, pp. 30–44, 1991.
- [116] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) standard,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1648–1667, 2012.
- [117] Norm (mathematics) [http://en.wikipedia.org/wiki/Norm_\(mathematics\)](http://en.wikipedia.org/wiki/Norm_(mathematics))
- [118] M. Bauer and K. Sayood, “Video coding using 3 dimensional DCT and dynamic code selection,” *Proceedings of Data Compression Conference*, p. 451, Mar 1995.
- [119] M. Servais and G. de Jager, “Video compression using the three dimensional discrete cosine transform (3D-DCT),” *Proceedings of the South African Symposium on Commun. and Signal Process.*, pp. 27–32, 1997.
- [120] B.L. Yeo and B. Liu, “Volume rendering of DCT-based compressed 3D scalar data,” *IEEE Trans. Visualization and Computer Graphics*, vol. 1, no. 1, pp. 29–43, 1995.
- [121] B.J. Kim, Z. Xiong, and W.A. Pearlman, “Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT),” *IEEE Trans. Circuits Syst. Video Technol.*, Vol.10, pp.1374–1378, 2000.
- [122] S. Lim J. Xu, Z. Xiong and Y. Zhang, “3-D embedded subband coding with

References

- optimal truncation (3-D ESCOT)”, *Applied and Computational Harmonic Analysis*, Vol.10, pp.290–315, 2001
- [123] A. Secker, D. Taubman, “Highly scalable video compression using a lifting-based 3D wavelet transform with deformable mesh motion compensation.” *Proc. IEEE Int. Conf. Image Processing*, pp. 749-752, 2002.
- [124] A. M. Tourapis, “Enhanced predictive zonal search for single and multiple frame motion estimation,” *Proc. IEEE Int. Conf. Image Processing*, pp. 1069–1079, 2002.
- [125] Q. Liu, L. Yang, “Two-difference algorithm for video’s abrupt shot change detection,” *Proc. IEEE Int. Conf. Neural Network and Signal Processing*, pp. 1177-1180. 2003.
- [126] H.264/AVC Reference Software, JM11.0, <http://iphome.hhi.de/suehring/tml/>
- [127] ffmpeg version 1.2 <http://ffmpeg.zeranoe.com/builds/win64/static/>
- [128] *Subjective video quality assessment methods for multimedia applications*, ITU-T Rec. P.910 1999.
- [129] Key technical area software of the ITU-T, version JM11.0KTA2.5. <http://iphome.hhi.de/suehring/tml/download/KTA/>
- [130] S. Li, L. Yan, W. Feng, L. S. Peng, and G. Wen, “Complexity-constrained H.264 video encoding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, pp. 477-490, 2009.
- [131] X. Cai, J. S. Lim “Algorithms for transform selection in multiple-transform video compression,” *Proc. IEEE Int. Conf. Image Processing*, pp. 2481 – 2484, 2012.

Publications

Journal Papers

Zhouye Gu, Weisi Lin, Bu-Sung Lee, Chiew Tong Lau, “Low Complexity Video Coding Based on Two-dimensional Singular Value Decomposition (2D-SVD)”, IEEE Transaction on Image Processing (T-IP), vol. 21, no. 2, pp. 674-687, 2012

(A part of Chapter 3 in this thesis is based upon this paper)

Zhouye Gu, Weisi Lin, Bu-Sung Lee, Chiew Tong Lau, “Rotated Orthogonal Transform (ROT) for Motion-Compensation Residual Coding”, IEEE Transaction on Image Processing (T-IP), vol. 21, no. 12, pp. 4770 – 4781, 2012

(A part of Chapter 4 in this thesis is based upon this paper)

Zhouye Gu, Weisi Lin, Bu-Sung Lee, Chiew Tong Lau, Ming-Ting Sun, “Mode Dependent Templates and Scan Order for H.264/AVC Based Intra Lossless Coding”, IEEE Transaction on Image Processing (T-IP), vol. 21, no. 9, pp. 4106-4116, 2012

(A part of Chapter 5 in this thesis is based upon this paper)

Conference Papers

Zhouye Gu, Weisi Lin, Bu-Sung Lee, Chiew Tong Lau, Manoranjan Paul, “Two dimensional Singular Value Decomposition (2D-SVD) based video coding.” IEEE International Conference on Image Processing (ICIP) 2010, pp. 181-184

(A part of Chapter 3 in this thesis is based upon this paper)

Zhouye Gu, Weisi Lin, Bu-Sung Lee, Chiew Tong Lau, “Adaptive Orthogonal Transform for Motion Compensation Residual in Video Compression.” International Conference on MultiMedia Modeling (MMM) 2011, pp. 40-50

(A part of Chapter 4 in this thesis is based upon this paper)

Zhouye Gu, Weisi Lin, Bu-Sung Lee, Chiew Tong Lau, “Improve H.264/AVC lossless coding by coefficients reordering” International Conference on Systems, Signals and Image Processing (IWSSIP), 2012, pp. 397-400