# A Novel Multi-Symbol Curve Fit based CABAC Framework for Hybrid Video Codec's with Improved Coding Efficiency and Throughput

by

Krishnakanth Rapaka

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2012

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

## Abstract

Video compression is an essential component of present-day applications and a decisive factor between the success or failure of a business model. There is an ever increasing demand to transmit larger number of superior-quality video channels into the available transmission bandwidth. Consumers are increasingly discerning about the quality and performance of video-based products and there is therefore a strong incentive for continuous improvement in video coding technology for companies to have market edge over its competitors. Even though processor speeds and network bandwidths continue to increase, a better video compression results in a more competitive product. This drive to improve video compression technology has led to a revolution in the last decade. In this thesis we addresses some of these data compression problems in a practical multimedia system that employ Hybrid video coding schemes. [20, 61]

Typically Real life video signals show non-stationary statistical behavior. The statistics of these signals largely depend on the video content and the acquisition process. Hybrid video coding schemes like H264/AVC [74, 61] exploits some of the non-stationary characteristics but certainly not all of it. Moreover, higher order statistical dependencies on a syntax element level are mostly neglected in existing video coding schemes. Designing a video coding scheme for a video coder by taking into consideration these typically observed statistical properties, however, offers room for significant improvements in coding efficiency. In this thesis work a new frequency domain curve-fitting compression framework is proposed as an extension to H264 Context Adaptive Binary Arithmetic Coder (CABAC)[53] that achieves better compression efficiency at reduced complexity. The proposed Curve-fitting extension to H264 CABAC, henceforth called as CF-CABAC, is modularly designed to conveniently fit into existing block based H264 Hybrid video Entropy coding algorithms. [74, 61]

Traditionally there have been many proposals in the literature to fuse surfaces/curve fitting with Block-based, Region based, Training-based (VQ, fractals) compression algorithms primarily to exploiting pixel-domain redundancies. Though the compression efficiency of these are expectantly better than DCT transform based compression, but their main drawback is the high computational demand which make the former techniques non-competitive for real-time applications over the latter.

The curve fitting techniques proposed so far have been on the pixel domain. The video characteristic on the pixel domain are highly non-stationary making curve fitting techniques not very efficient in terms of video quality, compression ratio and complexity. In this thesis, we explore using curve fitting techniques to Quantized frequency domain coefficients. we

fuse this powerful technique to H264 CABAC Entropy coding. Based on some predictable characteristics of Quantized DCT coefficients, a computationally in-expensive curve fitting technique is explored that fits into the existing H264 CABAC framework.

Also Due to the lossy nature of video compression and the strong demand for bandwidth and computation resources in a multimedia system, one of the key design issues for video coding is to optimize trade-off among quality (distortion) vS compression (rate) vS complexity. This thesis also briefly studies the existing rate distortion (RD) optimization approaches proposed to video coding for exploring the best RD performance of a video codec. Further, we propose a graph based algorithm for Rate-distortion optimization of quantized coefficient indices for the proposed CF-CABAC entropy coding.

The Proposed Multi-symbol Curve-fit CABAC was incorporated into the encoder and decoder of the JM 18.3 software. It was applied to the coding of significant coefficients. Experiments were performed using common conditions specified in [77] for nine different set of sequences. Table 4.15 lists the comparison results of proposed MSCF-CABAC vS H264 CABAC for three different operating points (High Rate, Medium Rate and Low Rate). The QP values used for High rate range from 22-26, for Medium rate from 28-32 and for low Rate from 34-38.

- **Compression efficiency** Average Percentage Bitrate gain for same PSNR of proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 0.36%, 1.07% and 0.83% respectively. Peak Percentage Bitrate gain for these sequences in same order are 0.62%, 3.87%, 3.02%.

- **Throughput improvement/Bin Reduction** Average Percentage Total Picture Bins Reduction at the above mentioned operating points for proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 0.55%, 1.33% and 1.05% respectively. verage Percentage Total Picture Bins Reduction for these sequences in same order are 0.83%,4.11%,2.65%.

Total coefficients Bin Reduction (significant Map, Last Coefficient Map and Residual Level and sign coding).

Average Percentage Total coefficients Bin Reduction at the above mentioned operating points for proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 1.17%, 2.63% and 3.59% respectively. Peak Percentage Bitrate gain for these sequences in same order are 1.67%, 4.94%, 5.67%.

Also we investigate trade-off between compression efficiency and complexity of the proposed CF-CABAC scheme and suggest some optimal trade-off solutions.

**Acknowledgements**

There are many people I wish to thank for their support and contributions throughout my graduate studies at University of Waterloo. First of all, I wish to express my sincere gratitude to my advisor, Professor En-hui Yang for his insightful guidance, valuable advice and support. The extensive research training under his supervision in Video Coding and Information Theory has greatly benefited me. His cutting-edge research and exceptional knowledge and innovations that have made an impact in the real world has been a great source of inspiration to me.

I would also like to thank Professor Zhou Wang and Professor Liang-Liang Xie for their review of my thesis and their constructive comments and helpful advice.

I would like to thank all my friends and colleagues at the Multimedia Communications Laboratory for all the helpful discussions and their wonderful friendship.

Last but not the least, I would like to thank my wife and my family for their love and continued support.

*To my loving wife*
*Anu*

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Over the last decade, there has been revolution in the way we create, share and watch videos. The analogue television, VHS video tapes, Cell phones, consumer video cameras has changed to digital television, Blu-ray, HD-DVDs, smartphones, High end HD video camera's. Many factors have contributed to the shift towards digital video, importantly the technological advances. From the technology viewpoint, some of these factors include better communications infrastructure, with widespread, relatively inexpensive access to broadband networks, 3G mobile networks, cheap and effective wireless local networks and higher-capacity carrier transmission systems but more importantly the key to the widespread adoption of digital video technology - video compression. [53, 20, 27, 28, 59]

For a True HD broadcast content, the effective raw bandwidth it can consume per second for a 10-bit video at 29.97 frames/second would be (1125*2200*29.97*20) = 1.485 Gbps. Just one second of True HD video content would consume as much as 1.485 Gbps, which is 1,485 million bits every second. With most applications infrequently having to share the network with other data intensive applications, this is very rarely the bandwidth available. To circumvent this problem, advanced Video compression techniques have been derived to reduce this high bit-rate. Their ability to perform this task is quantified by the compression ratio. Higher the compression ratio, the smaller is the bandwidth consumption. However, there is a price to pay for this compression: increasing compression causes an increasing degradation of the image. Nowadays for Broadcast applications, the true content is compressed to around $\sim$ 8 to 12 Mbps. To achieve such high compression ratio's with reasonable good quality is a real challenge.

Typically Real life video signals show non-stationary statistical behavior. The statistics of these signals largely depend on the video content and the acquisition process. Hybrid

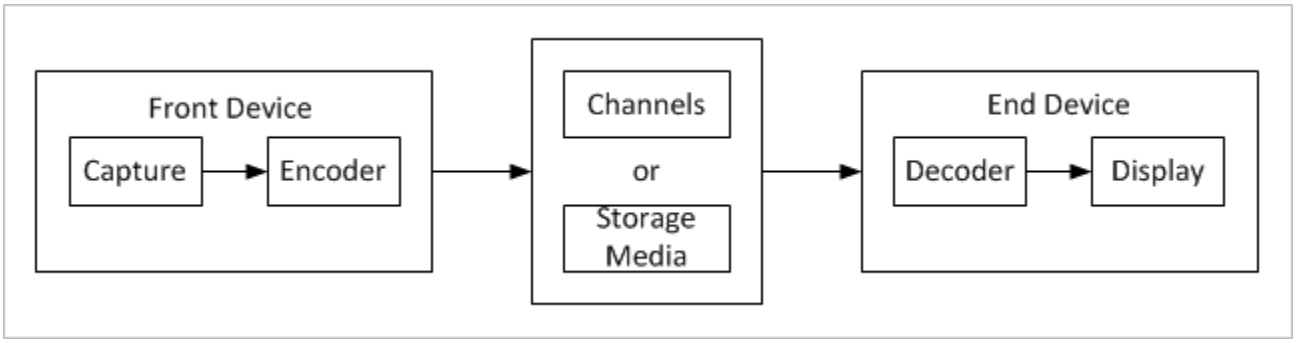


Figure 1.1: Practical multimedia system

video coding schemes like H.264/AVC exploits some of the non-stationary characteristics but certainly not all of it. Moreover, higher order statistical dependencies on a syntax element level are mostly neglected in existing video coding schemes. Designing a video coding scheme for a video coder by taking into consideration these typically observed statistical properties, however, offers room for significant improvements in coding efficiency.

Broadly speaking, this thesis addresses some of these data compression problems [69] in a practical multimedia system that employ Hybrid video coding schemes.

A practical multimedia system as shown in Figure 1.1 involves a Front-end, a Back-end device, and a connection in between them through channels or storage media. A conventional system setting for researching on video compression is the pair of encoder and decoder, assuming abundant computation power for encoding, limited computation power for decoding and limited channel bandwidth/storage for transmission of data.

When designing such a system, critical questions for consideration are:

- What is the available network bandwidth?
- What image degradation is allowed due to compression or expected video quality?
- What is the complexity budget for the system?
- If there is a scheme that provides better compression over existing schemes that simplifies decoding complexity and reduces channel bandwidth/storage requirement which is the first problem addressed in this thesis.
- Further, for such a system with given channel bandwidth, what is the best quality trade-off that can be achieved, which is the second major problem also addressed

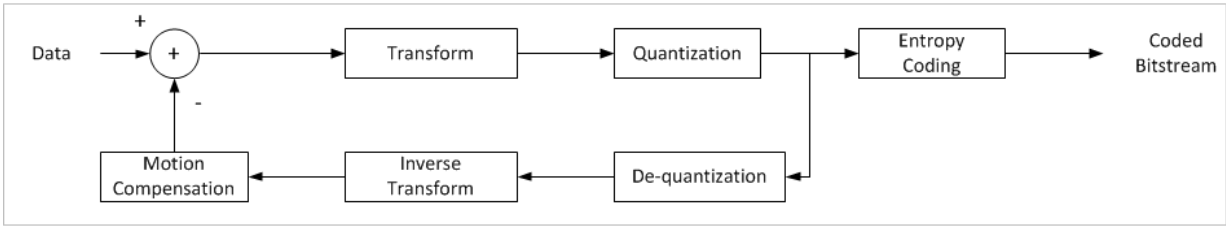


Figure 1.2: Illustration of Hybrid Video coding structure with Motion compensation, Transform,Quantization and Entropy Coding.

by this thesis, to achieve the best RD trade-off for the proposed scheme for lossy compression coding.

Lossy video compression under the conventional system setting with abundant encoding power generally adopts a hybrid structure as shown in Figure 1.2, where several different compression techniques such as motion prediction[27, 28], transform [29], quantization[16], and entropy coding [10, 53] are employed together. In general, this is referred to as hybrid video compression. This structure is adopted in almost all lossy video coding standards in the industry [58, 61] that exploits to some extent the temporal redundancy (similarity between frames), the spatial redundancy (similarity between neighboring pixels), the psycho-visual redundancy (limited sensitivity to spatial details by human eyes).

Section 1.3 discusses few of these compression techniques in some details. In this thesis, we mainly concentrate on the Entropy coding feature of the hybrid video coding structure.

## 1.1 Thesis Motivations

Work in this thesis is mainly motivated by a desire to answer the following questions in the multimedia system shown in Figure 1.1.

1. Is there a coding scheme that fits into existing Hybrid Video coding framework that provides better compression with high throughput over existing schemes.

Video compression is an essential part in any applications because of the enormous volume of video data. As digital video has become a ubiquitous and essential component of the entertainment, broadcasting, and communications industries, there is a never ending pursuit of more bandwidth/storage space for accommodating the explosively growing video data. This is fueling the demand for video compression to pursue the possibly best compression efficiency. Another important motivation is increasing the throughput (number

of bins/symbol coded at given time). Entropy coding compression, in particular,Context adaptive Binary arithmetic coding (CABAC) is inherently serial due to strong bin-to-bin data dependencies, and typically only a single bin is coded at a time. Consequently, the CABAC is often the bottleneck in the Video codec. Also for High Definition or Low delay video coding, symbols need to be coded at very high rates, which further pushes CABAC to be the main bottleneck in the video Coder/decoder.

In this thesis, we mainly concentrate on the Entropy coding compression scheme of the hybrid video coding structure and propose a new frequency domain High throughput Multi-Symbol curve-fitting based compression framework as an extension to Hybrid video coding H.264 Context Adaptive Binary Arithmetic Coder (CABAC) frame work that achieves better compression efficiency at higher throughput. The proposed scheme is modularly designed to conveniently fit into existing block based H.264 Hybrid video Entropy coding algorithms. Further, we propose Rate-distortion optimization framework for quantized coefficient indices to achieve the better RD cost for the proposed Multi-Symbol curve-fit-CABAC (MSCF-CABAC) entropy coding.

2. What is the best Rate Distortion [8, 86, 31, 56, 73] coding performance of the proposed scheme on H.264-compatible codec can achieve?

Video coding standards provide a solid base for the development of digital Video industries by promoting worldwide interoperability. Therefore, our study on the best RD coding performance will be within a standard coding scheme, i.e., to maintain compatibility with an industrial coding standard with above proposed extension. H.264, the newest hybrid video compression standard[74, 61], has proved its superiority in coding efficiency over its precedents, e.g., it shows a more than 40% rate reduction over H.263[61].

In this thesis we briefly study the existing rate distortion (RD) optimization approaches proposed to video coding for exploring the best RD performance of a video codec. Further, we propose a graph based algorithm for Rate-distortion optimization of quantized coefficient indices for the proposed MSCF-CABAC entropy coding.

## 1.2 Thesis Contributions

Contributions in this thesis are summarized as follow:

- A new frequency domain Multi Symbol curve-fitting Entropy coding framework is proposed as an extension to H.264 Context Adaptive Binary Arithmetic Coder (CABAC) that achieves better compression efficiency at reduced complexity.

Traditionally there have been many proposals in the literature to fuse surfaces/curve fitting [2, 4] with Block-based, Region based, Training-based (VQ[24], fractals[22]) compression techniques algorithms primarily to exploiting pixel-domain redundancies. Though the compression efficiency of these are expectedly better than DCT transform based compression, but their main drawback is the high computational demand which make the former techniques non-competitive for real-time applications over the latter. Also as far as Author knows, most of these techniques were applied to spatial domain. In this work, based on some predictable characteristics of Quantized DCT frequency coefficients a simple computationally in-expensive curve fitting technique is explored on the frequency domain data and the results are compared to existing Hybrid video coder (H.264).

- Rate distortion optimization framework for the proposed MSCF-CABAC entropy coding scheme

  In general, different entropy coding methods require different algorithms for Soft Decision Quantization (SDQ) [81, 84, 59]. Depending on the entropy coding method involved, the problem of designing algorithms for optimal or near optimal SDQ in conjunction with that specific entropy coding method could be very challenging, especially when the involved entropy coding method is complicated. Fortunately, in this thesis, we are able to solve the design problems of SDQ in conjunction of MSCF-CABAC. It is shown that given quantization step sizes, the proposed SDQ algorithms based on MSCF-CABAC achieve near-optimal residual quantization in the sense of minimizing the actual RD cost.

## 1.3 Thesis Organization

Chapter 2 presents a brief overview of traditional Image/video compression schemes. In Chapter 3, we review the four coding components in a typical hybrid coding structure, i.e., motion compensation, transform, quantization, and entropy coding. Since practices of data compression take root in Shannons information theory[39], the discussion is intended to explain some underlying principles for those four coding parts from an information theoretic point of view. However, the theoretic discussion is limited to an introductory level. Some other discussions are presented from an algorithm design point of view, explaining corresponding state-of-the-art techniques and how they can be applied. Then, the next section introduces the development of video coding standards from the early MPEG-1 to the newest H.264 (also referred to as MPEG-4, part-10) as background material and

motivations to our study on RD optimization for video compression. Finally, the last section is devoted to in-depth details of the H.264 Context adaptive Binary Arithmetic coding, based on which we will develop algorithms for applying our proposed Multi Symbol Curve fitting based CABAC to achieve best compression and R-D coding performance

Chapter 4 the proposed Multi- symbol curve-fitting based CABAC entropy coding. We begin with motivation for using curve fitting schemes and proceed to defining new syntax elements based on the statistical data collected over a large set of samples.

In the third section we demonstrate the coding of MSCF-CABAC using few examples and in the last section, we provide experimental results for implementing these algorithms based on H.264 reference codec JM 18.3.

Chapter 5 presents the RD optimization framework for hybrid video compression. We begin with a brief survey on related work in the literature, highlighting the difficulty of using the actual RD cost in conventional RD optimization approaches. To tackle this issue, we discover an SDQ mechanism based on a universal fixed slope lossy coding scheme. Using SDQ instead of the conventional HDQ, we then establish an RD optimization framework for obtaining best coefficient indices that minimizing the RD cost. Specifically, in the second section, we review the universal fixed-slope lossy coding scheme and apply it to optimizing hybrid video compression, obtaining SDQ. Based on the idea of SDQ, in the third section, we then formulate an RD optimization problem mathematically and based on which we construct a graph structure so that the RD cost can be computed in an additive manner. As a result, the additive computation of the RD cost enables us to use dynamic programming techniques to search for quantization outputs to minimize the actual RD cost, yielding an SDQ design based on MSCF-CABAC. In the final section, we provide experimental results for implementing these algorithms based on H.264 reference codec Jm82.

Finally, Chapter 6 concludes the thesis and discusses future research.

# Chapter 2

# Image / Video Compression Schemes

Over the last two decades a lot of image compression techniques have been proposed [20] that can be categorized on a high level into four subgroups: pixel-based, block-based, sub-band-based and region based.

## 2.1 Review of Compression techniques

### 2.1.1 Pixel-based Techniques

In this technique the operations are performed at pixel-level and are mostly based on predictive methods. These methods are usually loss-less and can provide compression gains of up to 4:1 (depending on the content) but these techniques are computationally highly intensive. The idea behind predictive methods is to encode the value of the difference between the previously encoded pixel and the current pixel. Due to the correlation existing in a natural image, the resulting values to be encoded typically have a lower dynamic range than the original values. That is, in predictive schemes while decompressing a pixel value X(i,j) is predicted based on the coded past $\widehat{X}(i,j)$. From an encoder perspective, the resulting difference between the original value and the predicted value (is called the prediction error) is coded in the compressed stream.

$$e(i,j) = X(i,j) - \widehat{X}(i,j) \tag{2.1}$$

Further, the successive values of $e(i, j)$ are typically quantized and coded into the compressed stream. If loss-less compression is required, the signal must have a limited number of possible values and is not quantized.

### 2.1.2 Block-based Techniques

These compression techniques were suggested in [35] where operations are performed on non-overlapping blocks of the image. This technique is predominantly used for Video compression due to better efficiency over pixel based techniques. However, at higher compression rates, these techniques suffer from visually annoying artifacts at block boundaries and requires post-processing like de-blocking filters.

Block based techniques can be further categorized into training based and non-training based techniques. In training-based techniques, some off-line learning or training is required before applying these techniques. Training based techniques include: vector quantization (VQ) [44], and neural networks (NN) [36] and [41]. Iterated functions or fractals [79] can also be considered as a category of VQ with a virtual code-book composed of blocks surrounding the current block.

Non-training type techniques include: block truncation coding (BTC) [18], transform coding (TC) (e.g., Discrete Cosine Transform (DCT) [23]), and surface fitting [51].

### 2.1.3 Sub-Band Based Techniques

Sub-band coding (wavelets) [48] differs from block-based techniques in performing the transformation on the whole image rather than part of it. However, some of these techniques can operate on large blocks. Hence, it has less blocking artifacts compared to block based coding techniques; however, the reconstructed image tends to be blurry. Nevertheless, it is known that its performance is much better than traditional block-based techniques [42] and [12].

The sub-bands are constructed through successive filtering and down-sampling (up-sampling at the decoder) [47]. This technique can be viewed as performing block processing in the frequency domain. In wavelet image compression [68], the image is decomposed into four bands, namely: Low-Low, Low-High, High-Low, and High-High where Low and High correspond to a low and a high pass filter respectively for each direction (Horizontal and Vertical). The high pass filter can be obtained by subtracting the low pass filtered output from the original image. The sequence of filter application is arbitrary, i.e., we can

apply filtering to the horizontal and then to the vertical direction and vice versa. Further a down-sampling by a factor of two is applied and the process can be applied for several times on the resulting Low-low(1) band etc.. followed by bands Low-Low(n), Low-High(n), High-Low(n), High-High(n) and so on. The coefficients at each band are then quantized to reach the desired compression. Since most of the energy is concentrated in the Low-Low band, larger quantization steps are used in other bands and hence a small portion of the bit budget is allocated to high frequency sub-bands [15], [88]. Many filter models have been proposed in the literature [48] for wavelet image compression. The JPEG2000 standard uses the Daubechies (9,7) filter due to its superior empirical performance on a wide range of images. To reduce computational burden, the filter is broken to four 2x2 matrix multiplications using the lifting scheme [1].

### 2.1.4 Region-Based Compression

Region based techniques are also called as second generation compression technique because of their superior performance over the previously mentioned schemes [14], [66] and [70]. Traditional transform coding techniques have a limit on the compression ratio that it can achieve.[25] and [37]. Region- or segmentation-based techniques were proposed to exceed this barrier [38] [65] and It has been shown by [50] and [60] that at higher bit-rates Video quality of region-based techniques exceeds that of DCT. But these involve a lot of computations over block-based schemes which are typically fast and involve less computations.

The general idea of region or segmentation-based compression is to divide the image into regions that may/may not necessarily of regular shape. These regions are decided using some clustering or segmentation procedure that depends various features like motion [65] ,shape, brightness, texture etc. In a simple Region based segmentation technique each region might be represented by two codes. The first (preferably a chain code) describes the location of boundary pixels. The second represents the best approximation of the region enclosed by this boundary. In addition, coding gains can be obtained by avoiding repetition of the common boundary points between adjacent regions. The usual compromise between quality and Compression Ratio is dependent to a certain extent on the number of regions.

Lots of algorithms have been proposed in literature to segment the image into different regions like Watershed algorithm, morphological operators [11], [57] and [66], region growing [80], genetic algorithms GA combined with gradient information, and many other algorithms [54] A pre-processing can be used to better describe the texture regions [33] through some statistical test to separate edges from uniform or texture regions.

After segmentation, each region can now be approximated in one of many different ways. [50], for example, implemented up to second order surfaces to describe each region. A similar procedure is to use some basis functions [25] followed by an orthogonalization routine. Colored regions [62] can be described according to HVS sensitivity by describing G component using a 2nd order polynomial, while R and B components are constructed using a linear function of G. Successive approximation (from an orthonormal set) was implemented in [37] independently on each region. A linear system is then solved to find the weights corresponding to the selected orthogonal functions. In [57] a suggestion was made to implement successive approximation between frames and/or resolution layers. VQ can also be combined with polynomials [21] to reach a better compromise between quality and performance. [38] implemented a DCT scheme defined on the smallest MxN circumscribing rectangle.

## 2.2 Curve / Surface Fitting

The process of constructing a compact representation to model the Curve/surface of an object based on a fairly large number of given data points is called curve/surface fitting. In Image compression, surface fitting is also sometimes called as polynomial fitting. It has been predominantly used in image segmentation [45] and quality improvement of block-based compression in [55] and [3] respectively.

This powerful technique has been fused in some of the techniques described in the previous section. In the general field of image processing, surface (or polynomial) fitting has been used in image segmentation [46], image noise reduction [71] and quality improvement of block-based compression [40] by O. Egger, et. Al and [43]. Lost sub-band coefficients [32] can be reconstructed by fitting the known samples to some surface. Splines can be used [7] to encode the lowest frequency band in sub-band coding. RBF networks [41] can be combined with surface fitting to perform compression using a predefined set of patterns for the centers. The term surface fitting was also used by [Chen et al 1994] to describe successive mean approximation. Polynomial fitting was implemented [13] in contour coding of black and white images. Splines were used in block-based compression [78] to preserve continuity between the pixels inside the block. Image representation by verge (high curvature) points [64] is an elegant suggestion to emphasize the importance of boundary pixels (edges) in producing perceptually pleasant pictures. Another implementation of polynomial fitting is in predicting motion compensation vectors in video coding [52]. Segmentation-based [9] image compression also uses 1D and 2D polynomial fitting. The former is used to encode boundary points while the latter to approximate slowly varying areas enclosed by these

points. To reduce complexity, slowly varying regions are usually approximated by a constant intensity depending on the split and merge technique. A flexible way of constructing variable size triangular blocks through split and merge was implemented in [49] and [19].

Lot of curve fitting algorithms has been proposed to reduce complexity. A simplified method for first order Curve fitting was proposed by [72]. In [5], H. Aydinoglu proposed incomplete polynomial transformation to quantize the coefficients. Polynomial fitting schemes are investigated by [8] as multiplication limited and division free implementations. In 2006, a very simple plane fitting scheme using three parameters was implemented [4]. Later in 2008, a linear mapping scheme using 8x8 block size was proposed [3]. In 2008, a plane fitting scheme with inter-block prediction using 1st curve order with 8x8 block size was also proposed [3].

Curve fitting can obtain a superior value of PSNR and subjectively quality of an image over the compression techniques discussed in previous sections if it is possible is to find efficient representations of an image using polynomial fitting. Due to non-stationary characteristics of the video content it is not possible most of the time. Also, it is very computationally intensive and can become extremely difficult to find the best fitting scheme for different contents that best trades-off Video quality and compression ratio. More complex fitting schemes require more data to convey to the decoder. Different fitting schemes have been proposed such as First Order Polynomial (Curve) Fitting which is fast and multiplication limited and division free model [20]. Higher order polynomial fitting increases the quality of the image but decreases the compression ratio. Higher order polynomial also increases the computational complexity, compared to 1st order curve fitting. Implementation of polynomial schemes with orders greater than two is difficult [2].

The curve fitting techniques proposed so far have been only on the pixel domain. The video characteristic on the pixel domain are highly non-stationary making curve fitting techniques not very efficient in terms of video quality, compression ratio and complexity. In this thesis, we explore using curve fitting techniques to Quantized frequency domain coefficients. we fuse this powerful technique to H.264 CABAC Entropy coding. Based on some predictable characteristics of Quantized DCT coefficients, a computationally inexpensive curve fitting technique is explored that fits into the existing H.264 CABAC framework.

# Chapter 3

# Hybrid Video Coding Standards

Many different new and innovative compression techniques for video coding have been proposed and researched though commercial video coding applications tend to use a limited number of standardized techniques for video compression. International standards for video compression have played an important role in the development of the digital video industry. Since early 1980's, many standards have been developed. Standardized video coding formats have potential benefits like:

1. interoperability between encoders and decoders from different manufacturers
2. makes it possible to build platforms that incorporate video, in which many different applications such as video codecs, audio codecs, transport protocols, security and rights management, interact in well-defined and consistent ways.
3. Takes care of patent infringements patent(s). The techniques and algorithms required to implement a standard are well-defined and the cost of licensing patents that cover these techniques, i.e. licensing the right to use the technology embodied in the patents, can be clearly defined.

With the ubiquitous presence of technologies such as DVD/Blu-Ray, digital television, Internet video and mobile video, the dominance of video coding standards is set to continue for some time to come.

It is interesting to have a quick look at the development of these standards.

## 3.1 Overview of Compression Standards

### 3.1.1 JPEG

The JPEG standard, ISO/IEC 10918, is the single most widespread picture compression format of today. It offers the flexibility to either select high picture quality with fairly high compression ratio or to get a very high compression ratio at the expense of a reasonable lower picture quality. Systems, such as cameras and viewers, can be made inexpensive due to the low complexity of the technique. JPEG image compression contains a series of advanced techniques. It uses Discrete Cosine Transform (DCT) for de-correlating the image and these de-correlated frequency domain coefficients are followed by a quantization that removes the redundant information (the "invisible" parts).

### 3.1.2 Motion JPEG

A digital video sequence can be represented as a series of JPEG pictures. The advantages are the same as with single still JPEG pictures  flexibility both in terms of quality and compression ratio. The main disadvantage of Motion JPEG (a.k.a. MJPEG) is that since it uses only a series of still pictures it makes no use of video compression techniques. The result is a lower compression ratio for video sequences compared to more advanced video compression techniques like MPEG. The benefit is in its robustness, low latency requirement, error resilient as with no dependency between the frames, even if one frame is dropped/corrupted in the channel, the rest of the video will be unaffected and recovered.

### 3.1.3 JPEG 2000

JPEG 2000 was created as the follow-up to the successful JPEG compression, with better compression ratios. The basis was to incorporate new advances in picture compression research into an international standard. Instead of the DCT transformation, JPEG 2000, ISO/IEC 15444, uses the Wavelet transformation.

The advantage of JPEG 2000 is that the blockiness faced by JPEG is removed, but replaced with a more overall fuzzy picture. Whether this fuzziness of JPEG 2000 is preferred compared to the "blockiness" of JPEG is a matter of personal preference. Regardless, JPEG 2000 never took off and is still not widely supported either.

### 3.1.4 MPEG-1

The first video coding standard MPEG-1[10], still popular compression standard for videoCD, the common video distribution format throughout much of Asia, was developed by ISO/IEC MPEG. MPEG-1 was designed for bit rates ranging up to 1.5 Mbps mainly for CD-ROM kind of video applications.

Though, MPEG-1 is much less complex compared to contemporary standard. However, it has utilized all the main coding techniques for hybrid video coding such DCT, Motion estimation/compensation, as bi-directional inter-frame coding, variable length entropy coding, etc., to a limited degree. For motion prediction, MPEG-1 supports the three main types of frames, i.e., I-frame for intra prediction, P-frame for inter prediction, and B-frame for bi-directional prediction. The block partition in I-frames is $8 \times 8$, while the block size for inter prediction in P-frames is fixed as $16 \times 16$. Also, the prediction in MPEG-1 is based on full-pixels, while later on it advances to support half-pixel in MPEG-2, and quarter-pixel in H.264. DCT in MPEG-1 uses an $8 \times 8$ block size. For quantization in MPEG-1, there is one step size for the DC coefficient, and 31 step sizes for the AC coefficients. The 31 step sizes take the even values from 2 to 62. For AC coefficients of inter-coded blocks, there is also a dead-zone around zero. Finally, entropy coding in MPEG-1 uses a simple scheme of concatenating run-length coding with variable length coding (VLC). A small VLC table is defined for most frequent run-level pairs, while other run-level combinations are coded as a sequence of 6-bit escape, 6-bit codeword for run, and 8-bit codeword for levels within [-127, 127] or 16-bit codewords for other levels.

### 3.1.5 MPEG-2

MPEG-2 [45] was developed soon after MPEG-1 mainly to enhance the quality for television applications like set-up boxes,DVD etc. that were not satisfied by the MPEG-1. Ever since it was finalized in November 1994, MPEG-2 has become a fundamental international standard for delivering digital video. The worldwide acceptance of MPEG-2 opens a clear path to worldwide interoperability. Today, MPEG-2 plays an important role in the market and it will continue to do the same in the near future. MPEG-2 based video products are developed for a wide range of applications. Following are some of the applications to name a few of them:

1. DVD: As a new generation of optical disc storage technology, DVD offers an up to 10G storage space for MPEG-2 video distribution. Ever since its introduction, DVD has become the most popular MPEG-2 based video product.

2. HDTV: MPEG-2 compression is used in HDTV applications to transmit moving pictures with resolution up to $1080 \times 1920$ at rate up to 30frame/second (requiring 20MHz bandwidth) through 8MHz channels.

3. Digital Camcorders: Originally, all digital camcorders use the Digital Video (DV) standard and record onto digital tape cassettes. However, the latest generation of camcorders turns to use MPEG-2 because it provides a high compression with high quality. Video data can be recorded directly onto flash memory or even to a hard disk. While transferring video files from a tape is slow because it requires real time play-back, a flash card/DVD/hard disk provides a much faster access to the video data.

Many issues such as the block size and prediction accuracy were not effectively addressed. In particular, motion compensation in MPEG-2 is based on a fixed size of $16 \times 16$, which leads to poor prediction when there are a lot of details in images. The prediction accuracy is fixed at half-pixel, while studies by [26] show that quarter-pixel accuracy is required for efficient motion compensation when distortion is small.

MPEG-2 utilizes $8 \times 8$ DCT where, the $8 \times 8$ block is the fundamental unit for residual coding in MPEG-2. Scalar quantization is applied to each $8 \times 8$ block of DCT coefficients in MPEG-2 with lower frequency coefficients taking smaller quantization step sizes and higher frequency components taking larger quantization step sizes. Quantization for intra blocks is slightly different. For an intra block, its DC components are quantized using one of 4 quantization step sizes, i.e., 1, 2, 4, 8, Accordingly, the 11-bit dynamic range of the DC coefficient is rendered to accuracy of 11, 10, 9, or 8 bits, respectively.

Each quantized coefficient in MPEG-2 is encoded as two parts, i.e., its absolute value and the sign. A set of variable length coding tables is designed to code the absolute values of quantized coefficients and other syntax elements. These tables are often referred to as modified Huffman tables, in the sense that they are not optimized for a limited range of bit rates. Coefficient signs are coded using fixed length codes with an underlying assumption that positive and negative coefficients are equally probable.

In summary, for a given macroblock, a motion vector is found by matching its $16 \times 16$ luma block with blocks in previously coded images, called reference frames. Predictions for both the luma block and two chroma blocks are computed based on this vector. Then, residuals are partitioned into $8 \times 8$ blocks and transformed using DCT. Scalar quantization is applied to the transform coefficients. Finally, variable length codes are used to encode the quantized coefficients.

### 3.1.6 MPEG-4/H.264

H.264 also called as Advanced Video Coding standard was co-published by two international standards bodies, the ITU-T (International Telecommunication Union) and the ISO/IEC (International Organization for Standardization/International Electro technical Commission). The H.264/AVC standard was first published in 2003, with several revisions and updates published since then. It builds on the concepts of earlier standards such as MPEG-2 and MPEG-4 Visual and offers the potential for better compression efficiency, i.e. better-quality compressed video, and greater flexibility in compressing, transmitting and storing video. One of the most important drivers for the standardization of H.264 and its subsequent adoption by industry is its improved performance compared with earlier standards. The benchmark for mass-market applications such as digital TV and consumer video storage on DVD-Video is the earlier MPEG-2 standard. H.264 offers significantly better compression performance than MPEG-2 Visual. Using H.264 it is possible to compress video into a much smaller number of bits than using MPEG-2, for the same video resolution and image quality. This means, for example, that much more video material can be stored on a disk or transmitted over a broadcast channel by using the H.264 format.

## 3.2 Detailed overview of H.264 Hybrid Video Compression

In this section we briefly discuss important aspects of H.264 Hybrid video compression, a hybrid coding structure [76], which was employed from the earliest MPEG-1 to the newest H.264 [63].

In this chapter, we first review the basic structure of hybrid video coding. Then, we briefly introduce the technical aspects of H.264 video coding techniques. Finally, we present a detailed review of H.264 Context adaptive Binary Arithmetic Coding, since one of the main objectives in this thesis is to come up a new frequency domain curve-fitting compression framework as an extension to H.264 Context Adaptive Binary Arithmetic Coder (CABAC) that achieves better compression efficiency at reduced complexity while optimizing its RD trade-off.

### 3.2.1 Hybrid Coding Structure

There are four coding techniques in the hybrid coding structure, i.e., motion compensation, transform, quantization, and entropy coding. This Section briefly reviews these principles and some design issues.

### 3.2.2 Motion Prediction

Perhaps the most important reason for the widespread adoption of H.264/AVC is its compression performance and much of the performance gain compared with previous standards is due to H.264/AVCs efficient prediction methods.

Video signals display a distinct kind of redundancy called temporal redundancy, with high correlation between neighboring frames. While in Image processing, data are well known for spatial redundancy among neighboring pixels; for video compression, higher correlation are often observed among adjacent frames which marks an important difference between still image and video coding, the latter more oriented towards the temporal redundancy processing.

For every macroblock, a prediction is created, an attempt to duplicate the information contained in the macroblock using previously coded data, and subtracted from the macroblock to form a residual as shown in Figure 3.1. The efficiency or accuracy of this prediction process has a significant impact on compression performance. An accurate prediction means that the residual contains very little data and this in turn leads to good compression performance. Consider a typical scenario when the object moves from location 'A' to location 'B'. Once the object from location 'A' is encoded in one frame, its appearance in the succeeding frames can be well represented with two factors, i.e., its shape and the displacement of motion from 'A' to 'B'. Motion compensation that allows arbitrary shapes is conceptually advanced and seldom used in real time video processing applications though they find importance in advanced compression systems [67] that don't have real time constraints. The major drawback of object-based motion compensation in terms the coding performance of compared to that of a block-based coding scheme [61], is the high rate required to coding the shape. Thus, block-based motion compensation is more widely used in video compression standards. An important factor for block-based coding is the block size. In general, a small block size will lead to more motion vectors, which means more overhead bits. However, it also means a better prediction. H.264 uses square/rectangle blocks for motion compensation with various block sizes, e.g., $16 \times 16$, $16 \times 8$, $8 \times 16$, $8 \times 8$, $8 \times 4$, $4 \times 8$, $4 \times 4$, resulting in more flexibility for this new standard to achieve superior coding efficiency.

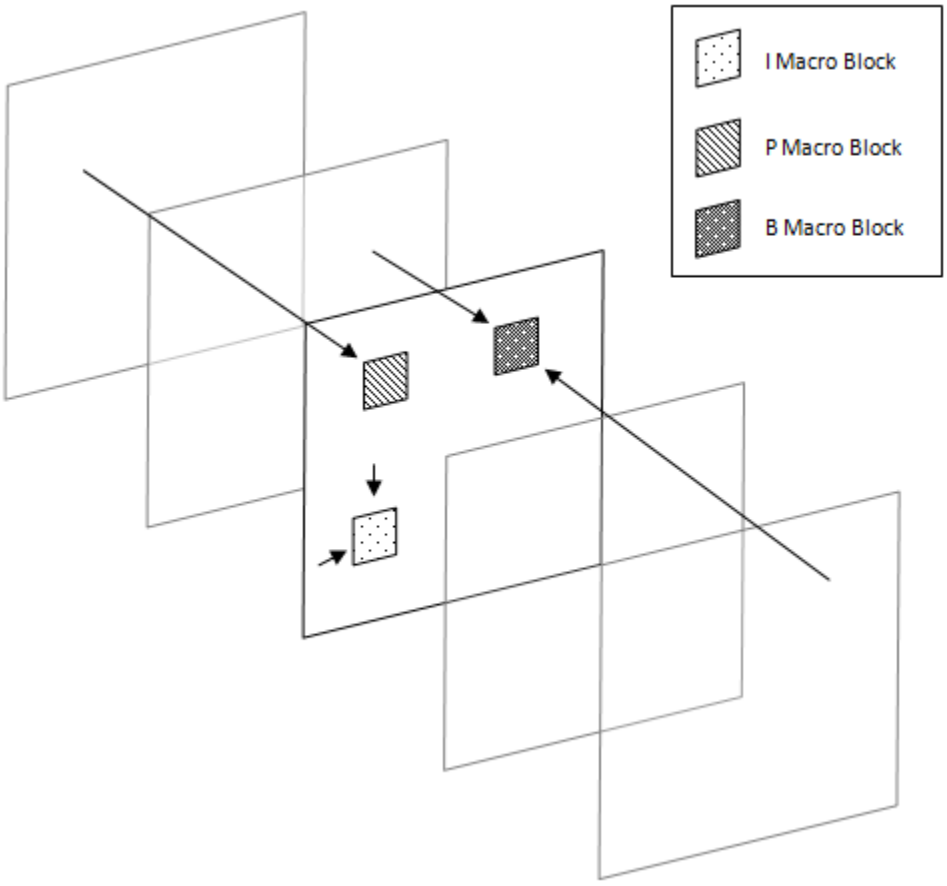


Figure 3.1: Example of H.264 Macroblock types and prediction sources

H.264/AVC supports a wide range of prediction options  intra prediction using data within the current frame, inter prediction using motion compensated prediction from previously coded frames, multiple prediction block sizes, multiple reference frames and special modes such as Direct and Weighted prediction. Another important factor for motion compensation is the prediction accuracy. In the early standard H.261, motion compensation is conducted on the original pixel map, so-called full-pixel prediction. The newest H.264 supports up to 1/4-pixel accuracy for the luma component. The samples at sub-pixel positions are created by interpolation in the reference frames. In general, the higher the prediction resolution is, the more effective motion compensation will be. However, studies by Girod in [28] show that the gain by using higher 11 prediction accuracy is limited in the sense that it becomes very small beyond a certain critical accuracy. It is suggested that 1/2-pixel accuracy be sufficient for motion compensation based on videophone signals, while 1/4-pixel accuracy be desirable for broadcast TV applications [28]. By selecting the best prediction options for an individual macroblock, an encoder can minimize the residual size to produce a highly compressed bitstream.

In Intra-Prediction an intra (I) macroblock is coded without referring to any data outside the current slice. I macroblocks may occur in any slice type. Every macroblock in an I slice is an I-macroblock. I macroblocks are coded using intra prediction, i.e. prediction from previously-coded data in the same slice. For a typical block of luma or chroma samples, there is a relatively high correlation between samples in the block and samples that are immediately adjacent to the block. Intra prediction therefore uses samples from adjacent, previously coded blocks to predict the values in the current block.

Inter prediction is the process of predicting a block of luma and chroma samples from a picture that has previously been coded and transmitted, a reference picture. This involves selecting a prediction region, generating a prediction block and subtracting this from the original block of samples to form a residual that is then coded and transmitted. The block of samples to be predicted, a macroblock partition or sub-macroblock partition, can range in size from a complete macroblock, i.e. $16 \times 16$ luma samples and corresponding chroma samples, down to a $4 \times 4$ block of luma samples and corresponding chroma samples.

The reference picture is chosen from a list of previously coded pictures, stored in a Decoded Picture Buffer, which may include pictures before and after the current picture in display order. The offset between the position of the current partition and the prediction region in the reference picture is a motion vector. The motion vector may point to integer, half- or quarter-sample positions in the luma component of the reference picture. Half- or quarter-sample positions are generated by interpolating the samples of the reference picture.

Each motion vector is differentially coded from the motion vectors of neighboring blocks. The prediction block may be generated from a single prediction region in a reference picture, for a P or B macroblock, or from two prediction regions in reference pictures, for a B macroblock [i]. Optionally, the prediction block may be weighted according to the temporal distance between the current and reference picture(s), known as weighted prediction. In a B macroblock, a block may be predicted in direct mode, in which case no residual samples or motion vectors are sent and the decoder infers the motion vector from previously received vectors.

### 3.2.3 Transform

In the previous section, we discussed about the prediction processes that remove some redundancy by creating and subtracting an estimate of the current block of image data. This 'front end' prediction stage is loss-less, i.e. it is a process that is fully reversible without loss of data. However, H.264 is fundamentally a lossy compression format, in which a degree of visual distortion is introduced into the video signal as a trade-off for higher compression performance. This distortion occurs in the transform/quantization process.

Transform coding has been widely used for lossy compression of video, image, and audio applications [25] [41] to de-correlate signals so that the outputs can be efficiently coded using techniques such as scalar quantization. Along with de-correlating the data, the key design criteria the transform should be satisfy are its reversibility and computationally tractability. Among many block-based transforms, the most popular one is the discrete cosine transform (DCT), which has been adopted in all lossy video coding standards. While an $8 \times 8$ DCT was used in early standards (MPEG1/2), the H.264 also uses a $4 \times 4$ DCT, which is know to give better coding efficiency and less blocky effect. As suggested from [49], the coding gain for using a small block size comes from the reduced inter-block correlation.

In earlier standards, the transform is defined as in equation below. For example, Equation 3.1 defines a two-dimensional inverse DCT for blocks of size $4 \times 4$, where $Y_{xy}$ are input coefficients and $X_{ij}$ are output image or residual samples.

$$X_{ij} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} C_x C_y Y_{xy} cos \frac{(2j+1)y\pi}{2N} cos \frac{(2i+1)x\pi}{2N} \tag{3.1}$$

Also in earlier standards there was an obvious boundary between the transform, converting a block of image samples into a different domain, and quantization, reducing the

precision of transform coefficients.However in H.264 codec, the transform and quantization stages are overlapped. This, together with the new approach of exactly specifying a reversible integer transform 'core', makes the H.264 transform and quantization stage significantly different from earlier compression standards. For example Implementation of (Equation 3.1) for $N > 2$ on a practical processor requires approximations to certain irrational multiplication factors, $cos(a\pi/2N)$. Different approximations can significantly change the output of the forward or inverse transform, leading to mismatch between different implementations of encoders and decoders. To limit the magnitude of this mismatch, the earlier standards specify that the inverse transform must meet accuracy criteria based on IEEE Standard 1180-1990 [iv]. Nevertheless, there is still likely to be a mismatch between inverse DCTs in an encoder, which must carry out the inverse DCT to reconstruct frames for inter prediction, and a decoder. This leads to discrepancies between the prediction references in encoder and decoder and to 'drift' or cumulative distortion in the decoder output. In MPEG-2 Video and MPEG-4 Visual, this is mitigated by ensuring that coded blocks are periodically refreshed by intra coding. In H.264/AVC and in other recent standards such as VC-1 [v], the transform and quantization processes are designed to minimize computational complexity, to be suitable for implementation using limited-precision integer arithmetic and to avoid encoder/decoder mismatch [vi, vii].

This is achieved by: using a core transform, an integer transform, that can be carried out using integer or fixed-point arithmetic and, integrating a normalization step with the quantization process to minimize the number of multiplications required to process a block of residual data. The scaling and inverse transform processes carried out by a decoder are exactly specified in the standard so that every H.264 implementation should produce identical results, eliminating mismatch between different transform implementations.

From the correlation point of view, however, the concatenation of motion compensation and transform coding is non-optimal. Intuitively speaking, the more effective is motion compensation, the less correlated are the residuals, thus the less interesting for transforming the residual to the frequency domain. Studies in [20, 22, 46] pointed out that residuals after motion compensation are only weakly correlated.

From the information theoretic point of view, the transform plus scalar quantization and entropy coding method is questionable too. The DCT transform tends to generate coefficients with Gaussian distributions when the block size is large, which may be justified by applying the central limit theorem. Particularly, Eude et al. showed by mathematical analysis that DCT coefficients of images could be well modeled with a finite mixture of Gaussian distributions [14]. Information theory shows that the rate distortion function of a stationary source achieves its upper bound with Gaussian distributions [1], indicating that Gaussian source is the most difficult for lossy compression either by vector quantization

or by a scheme with scalar quantization and entropy coding [26]. The fact that a small block size gives a better performance for using DCT transform possibly indicates that DCT transform in the hybrid structure is of much interest for reconsideration.

### 3.2.4 Quantization

The application of quantization to video compression is inspired by some cognitive studies on human visual systems. Human visual systems show excellent robustness in extracting information from video signals [59]. Bioelectrically, the human eyes response to spatial details is limited. Thus, a certain amount of distortion may be introduced into video signals while a human observer would not notice it. Furthermore, the human visual system allows a wide range of even noticeable distortion while it is still able to obtain critical information from the video signals. In other words, there exist much psycho-visual redundancy in image/video signals. From information theoretical point of view, this psycho-visual redundancy makes it possible to balance bandwidth and distortion according to given channel conditions, leading to the application of quantization.

Most video compression designs use scalar quantization, which is basically a simple arithmetic operation to shrink the dynamic range of inputs [61] [55]. It is a hard decision based operation in the sense that the quantization output for a given input is directly computed from the input itself and a quantization step size. Another method is to introduce soft decision quantization [84], by which we mean that quantization outputs are generated based on a rate distortion cost for an array of inputs, as to be discussed later. An intuitive interpretation of soft decision quantization is to adapt quantization outputs to the coding context of a given lossless coding algorithm. For hard decision quantization, the output is totally unrelated to the entropy coding part. Under such a circumstance, the best rate performance of the whole scheme is bounded by the entropy of quantization outputs. Then the gap between the entropy and the Shannon lower bound is an inevitable loss. However, studies in [36] show that the original entropy bound can be exceeded by optimizing quantization outputs with respect to the following lossless coding. As a result, the coding rate of the lossless algorithm will asymptotically approach the optimum given by the rate-distortion function.

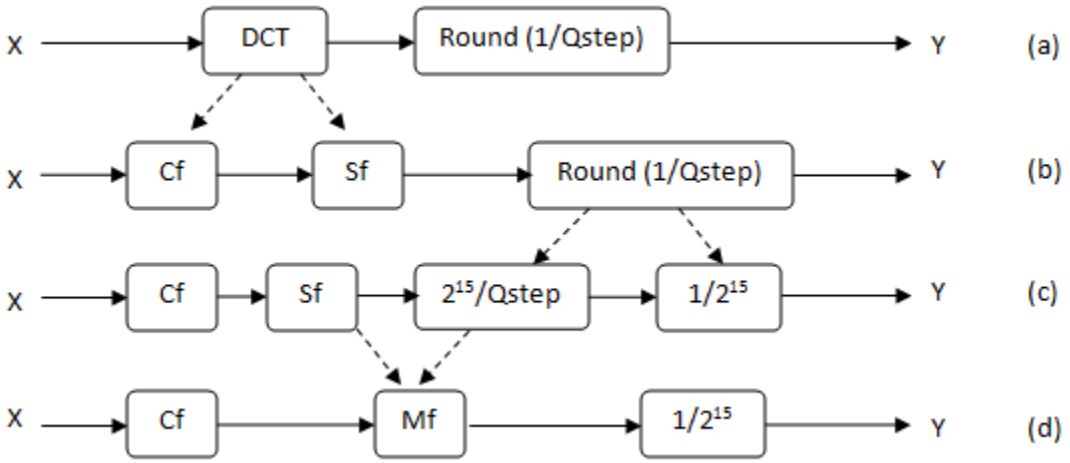


Figure 3.2: Development of the forward transform and quantization process

### 3.2.5 Deriving H.264 Forward Transform and Quantization process

The forward and inverse integer transform processes for $4 \times 4$ blocks are developed as follows. Starting from a $4 \times 4$ DCT, a scaled, rounded integer approximation to the DCT is derived to which a normalization step added for maintaining the orthonormal property of the DCT. Then this normalization step is integrated with the quantization process. The process is explained in detail in Figure 3.2

(a) Consider a block of pixel data that is processed by a two-dimensional Discrete Cosine Transform (DCT) followed by quantization, i.e. rounded division by a quantization step size, $Q_{step}$ ((a) of Figure 3.2).

(b) Rearrange the DCT process into a core transform ($C_{f4}$) and a scaling matrix ($S_{f4}$) ((b) of Figure 3.2) where ($C_{f4}$) and ($S_{f4}$) are given by:

$$C_{f4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

and

$$S_{f4} = R_{f4} \bullet R_{f4}^{T} = \begin{bmatrix} 1/4 & 1/2\sqrt{10} & 1/4 & 1/2\sqrt{10} \\ 1/2\sqrt{10} & 1/10 & 1/2\sqrt{10} & 1/10 \\ 1/4 & 1/2\sqrt{10} & 1/4 & 1/2\sqrt{10} \\ 1/2\sqrt{10} & 1/10 & 1/2\sqrt{10} & 1/10 \end{bmatrix}$$

(c) Scale the quantization process by a constant ($2^{15}$) and compensate by dividing and rounding the final result ((c) of Figure 3.2). The constant factor $2^{15}$ is chosen as a compromise between higher accuracy and limited arithmetic precision.

(d) Combine $S_{f4}$ and the quantization process into $M_{f4}$ ((d) of Figure 3.2), where:

$$M_f \approx \frac{S_f . 2^{15}}{Q_{step}} \tag{3.2}$$

The complete forward transform, scaling and quantization process for $4 \times 4$ blocks.

$$Y = round\left( \left[ C_{f4} \right] . \left[ X \right] . \left[ C_{f4}^{T} \right] \bullet m\left( QP\%6, n \right) / \left[ 2^{floor(QP/6)} \right] . \frac{1}{2^{15}} \right) \tag{3.3}$$

### 3.2.6 Developing the inverse quantization and inverse transform process

1. Consider a re-scaling or 'inverse quantization' operation followed by a two-dimensional inverse DCT (IDCT) ((a) of Figure 3.3).

2. Rearrange the IDCT process into a core transform (Ci) and a scaling matrix (Si) ((b) of Figure 3.3).

3. Scale the re-scaling process by a constant (26) and compensate by dividing and rounding the final result ((c) of Figure 3.3). Note that rounding need not be to the nearest integer.

4. Combine the re-scaling process and Si into Vi ((d) of Figure 3.3), where:

$$V_i \approx S_i \cdot 26 \cdot Q_{step} \tag{3.4}$$

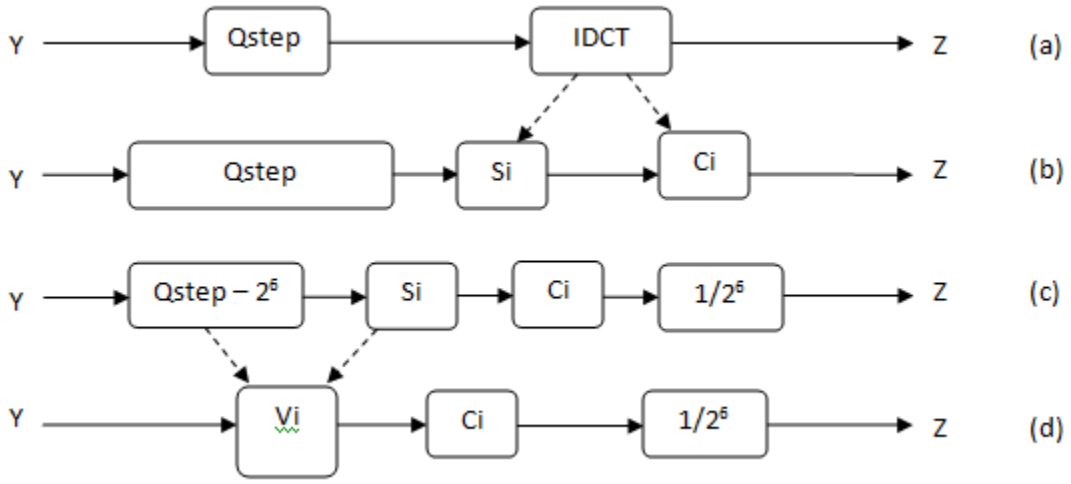


Figure 3.3: 2D Inverse DCT

### 3.2.7 Entropy Coding

The entropy encoder converts a series of symbols representing elements of the video sequence into a compressed bitstream suitable for transmission or storage. Input symbols may include quantized transform coefficients, run-level or motion vectors with integer or sub-pixel resolution, start/marker codes that indicate a resynchronization point in the sequence, sequence/picture/macroblock headers and may also contain side information that is not essential for normative decoding.

In Shannon's information theory, entropy means the amount of information presented in a source, which is quantitatively defined as the minimum average message length that must be sent to communicate by the encoder to decoder.

While most of the multimedia compression is usually lossy due to quantization, the entropy coding part is lossless that is the quantized outputs and other information discussed above are coded precisely/loss-less with possibly minimum number of bits. According to Shannon's source coding theorem, the optimal number of bits for coding a source symbol is $-log_2 p$, where p is the probability of the input symbol. An entropy coder seeks for the minimal number of bits for coding a given set of symbols [26].

The two most popular entropy coding methods are Huffman coding [28] and arithmetic coding [51]. The basic idea of Huffman coding is to encode a symbol with least number of that bits that has highest probability, which exactly follows Shannon's guideline of $-log_2 p$. The one major drawback of this scheme is that in most of the cases $-log_2 p$ may not be an

integer leading to a loss of coding efficiency by up to 1 bit/symbol.

For example, a source symbol with p = 0.6 would transmit 0.73 bits of information, but it consumes 1-bits if Huffman coding is used which is the efficiency loss of Huffman Coding as this coding scheme can only assign an integer number of bits to code a source symbol. The second major drawback is with Huffman

These drawback are solved to some extent by Arithmetic coding which is in theory superior to Huffman coding because:

1. It can assign a non-integer number of bits to code a symbol.
2. It can adapt to symbol statistics pretty quickly.

Though Arithmetic coding is provides better compression efficiency but there is a cost for arithmetic coding to pay for its high compression efficiency, i.e., the high computational complexity. Since an entropy codec is designed based on a mathematical model, the coding efficiency of an entropy codec in a real-world application is largely dependent on how well we can establish a mathematical model for the data to be compressed. Shannon's source coding theorem establishes a relationship between the symbol probability and the corresponding coding bits. In order to find the optimal representation, i.e., the minimal number of bits, the probability distributions of all symbols are required to be known, which unfortunately is not true for most real world applications. The solution is to estimate the distributions. In general, this is a big challenge for designing entropy coding methods. It requires complicated design and extensive computation. E.g., extensive experiments are conducted to study the empirical distributions of various syntax elements in H.264. Eventually, there are over 400 context models developed and complicated criteria are defined for context selection in the CABAC method[51].

## 3.3 Detailed Review of H.264 CABAC

H.264 supports two entropy coding methods for residual coding, i.e., context adaptive variable length coding (CAVLC) [3] in its baseline profile and context adaptive binary arithmetic coding (CABAC) [51] in its main profile.

Figure 3.4 shows the generic block diagram for encoding a single syntax element in CABAC. The encoding process consists of, at most, three elementary steps:

1. Binarization;

2. Context modeling;

3. Binary arithmetic coding.

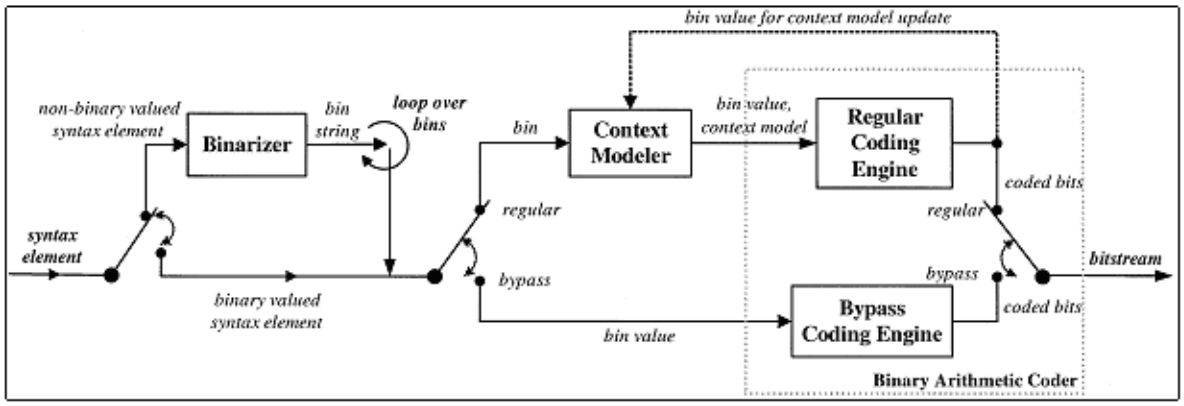


Figure 3.4: CABAC Framework

## 3.4 Binarization

A given non-binary valued syntax element is uniquely mapped to a binary sequence, a so-called ***bin*** string. When a binary valued syntax element is given, this initial step is bypassed, as shown in Figure 3.4. For each element of the bin string or for each binary valued syntax element, one or two subsequent steps may follow depending on the coding mode.

The Binarization is designed such that a binary representation for a given non-binary valued syntax element provided by the binarization process should be close to a minimum-redundancy code. On the one hand, this allows easy access to the most probable symbols by means of the binary decisions located at or close to the root node for the subsequent modeling stage. On the other hand, such a code tree minimizes the number of binary symbols to encode on the average, hence minimizing the computational workload induced by the binary arithmetic coding stage.

There are four such basic types defined in H.264 Standard: the unary code, the truncated unary code, the $K^{th}$ order Exp-Golomb code, and the fixed-length coding. In addition, there are binarization schemes based on a concatenation of these elementary types.

As an exception of these structured types, there are five specific, mostly unstructured binary trees that have been manually chosen for the coding of macroblock types and sub-macroblock types.

## 3.5 Context Modeling

Context modeling helps determine the code and its efficiency and is designed to explore the statistical dependencies to a large degree and to keep "up to date" during encoding. In H.264 CABAC, this problem is solved by imposing two severe restrictions on the choice of the context models. First, very limited context templates consisting of a few neighbors of the current symbol to encode are employed such that only a small number of different context models is effectively used. Second, context modeling is restricted to selected bins of the binarized symbols.

Four basic design types of context models can be distinguished in CABAC. The first type involves a context template with up to two neighboring syntax elements in the past of the current syntax element to encode, where the specific definition of the kind of neighborhood depends on the syntax element. The second type of context models is only defined for the syntax elements of mb_type and sub_mb_type. For this kind of context models, the values of prior coded bins are used for the choice of a model for a given bin with index. Both the third and fourth type of context models is applied to residual data only. In contrast to all other types of context models, both types depend on the context categories of different block types, as specified below. Moreover, the third type does not rely on past coded data, but on the position in the scanning path. For the fourth type, modeling functions are specified that involve the evaluation of the accumulated number of encoded (decoded) levels with a specific value prior to the current level bin to encode (decode).

The entity of probability models used in CABAC can be arranged in a linear fashion such that each model can be identified by a unique so-called context index. Each probability model related to a given context index is determined by a pair of two values, a 6-bit probability state index and the (binary) value of the most probable symbol (MPS), Thus, the pairs (bin, MPS) for and hence the models themselves can be efficiently represented by 7-bit unsigned integer values.

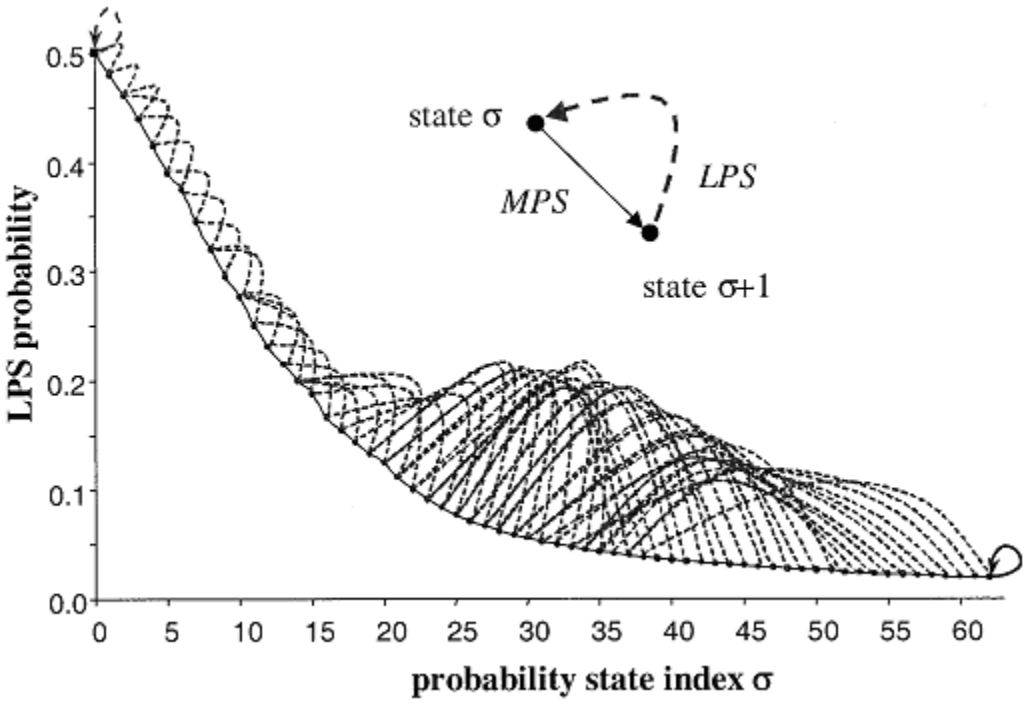


Figure 3.5: Context modeling

## 3.6 Binary Arithmetic Coding

Binary arithmetic coding is based on the principle of recursive interval subdivision that involves the following elementary multiplication operation. Suppose that an estimate of the probability of the *least probable symbol* (LPS) is given and that the given interval is represented by its lower bound and its width (range). Based on that setting, the given interval is subdivided into two subintervals: one interval of width which is associated with the LPS, and the dual interval of width, which is assigned to the most probable symbol (MPS) having a probability estimate of. Depending on the observed binary decision, either identified as the LPS or the MPS, the corresponding subinterval is then chosen as the new current interval. A binary value pointing into that interval represents the sequence of binary decisions processed so far, whereas the range of that interval corresponds to the product of the probabilities of those binary symbols. Thus, to unambiguously identify that interval and hence the coded sequence of binary decisions, the Shannon lower bound

on the entropy of the sequence is asymptotically approximated by using the minimum precision of bits specifying the lower bound of the final interval.

## 3.7 Residual Coefficient Data coding Process

Listing 3.7 and 3.7 illustrates the CABAC encoding scheme for a single block of transform coefficients. If the coded block flag is non-zero, a significance map specifying the positions of significant coefficients is encoded. Finally, the absolute value of the level as well as the sign is encoded for each significant transform coefficient. These values are transmitted in reverse scanning order.

Listing 3.1: Significance Map

```
for (i=0; i<MaxNumCoeff(BlockType)-1; i++)
{
  Encode significant_coeff_flag[i];
  if (significant_coeff_flag[i])
    Encode last_significant_coeff_flag[i];
  if (last_significant_coeff_flag[i])
        break;
}
```

Listing 3.2: Level Information

```
for (i=MaxNumCoeff(BlockType)-1; i>=0; i--)
{
  if (significant_coeff_flag[i])
  {
    Encode coeff_abs_level_minus1[i];
    Encode coeff_sign_flag[i];
  }
}
```

| Scanning Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Quantized coefficient Level | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Significant coefficient Flag | 0 | 0 | 1 | - | - | - | - | - | - | - | - |
| Last coefficient Flag | 0 | 0 | 1 | - | - | - | - | - | - | - | - |

Table 3.1: Significant Coefficient Mapping in H264 CABAC : Example

| Scanning Position | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Quantized coefficient Level | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Significant coefficient Flag | 0 | 0 | 1 | 1 | - | - | - | - | - | - | - |
| Last coefficient Flag | 0 | 0 | 0 | 1 | - | - | - | - | - | - | - |

Table 3.2: Last Coefficient Mapping in H264 CABAC : Example

# Chapter 4

# Proposed Curve Fitting Based Entropy Codec Extension to CABAC

As per H.264 standard syntax, each coefficient in the block is coded in the inverse scanning order. For example for a 4×4 block, a maximum of 16 coefficients could be coded depending on the absolute value of the coefficients. Henceforth we call absolute value of the significant coefficient as Coefficient Index. In this section, we discuss the proposed MSCF-CABAC Model, where two or more coefficient indices may be coded together which enables us to use a simplified linear curve fitting technique to exploit the higher order statistical dependencies thereby achieving better compression and throughput. Based on observed statistics for a wide range of video sequences of the coefficient indices, new syntax elements and its respective context modeling have been defined. In this work we apply Multi Symbol Curve fit CABAC for H264. This is applied to both luma and chroma for difference transform sizes supported in H264 standard.

This chapter is divided into six categories starting with Section 4.1 Motivation, Section 4.2 details the overview of MSCF-CABAC and new proposed syntax elements and their respective probabilities of occurrence for real life sequences. Section 4.3 details the context modeling for the proposed syntax elements. Section 4.4, details some special handling for blocks with only one significant coefficient or odd number of significant coefficients in the block. Section 4.5, discusses complete block encoding with MSCF-CABAC followed by a detailed Encoder/Decoder Syntax. In section 4.6, we finish this section by comparing the MSCF-CABAC encoding and H.264 CABAC encoding.

## 4.1 Motivation

The statistics of video signals largely depend on the video content and the acquisition process. Though most of the real life video signals exhibit non-stationary statistical behavior, the Quantized coefficients show some predictable statistical characteristics. H.264/AVC Entropy Coding, say CABAC, exploits some of the characteristics but certainly not all of it. Moreover, higher order statistical dependencies on a syntax element level are mostly neglected in existing Entropy coding schemes.

Figure 4.1 shows the typical distribution of Quantized DCT coefficient's at different operating QP's and picture types. It can be observed that the most probable absolute coefficient index is "one" followed by "two" and so on. The probability decreases as the value of index increases. H.264 CABAC tries to exploit this behavior to some extent by coding the coefficient Index of "one" with different context to that of coefficient Indices "greater than one". The major short coming of this approach is that it cannot consider the higher-order correlation between coefficient indices, since it always codes each coefficient index separately. In other words, H.264 CABAC is designed to work efficiently on overall probability of coefficient indices but not on the block-level or Multi-symbol level probability distribution of coefficient indices. Though it tries to address this indirectly to a small extent by using five different contexts for coefficient index "one" or "greater than one" based on the scanning position.

Designing an Entropy coding scheme that can exploit these observed higher order statistical properties, however, offers room for significant improvements in coding efficiency and which is the one of the primary motivation of the proposal.

Another important motivation is reducing the throughput (number of bins that can be coded in parallel). CABAC is inherently serial due to strong bin-to-bin data dependencies, and typically only a single bin is coded at a time. Consequently, the CABAC is often the bottleneck in the codec. Also for High Definition or Low delay video coding, symbols need to be coded at very high rates, which further pushes CABAC to be the main bottleneck in the video Coder/decoder. One solution is to allow CABAC Engine to be Operating at high frequencies which limits our ability to voltage scale and results in significant power consumption which is undesirable for battery operated devices. Hence there is a strong need to improve the CABAC throughput.

## 4.2 Overview of MSCF-CABAC

The encoding process consists of, at most, four elementary steps:

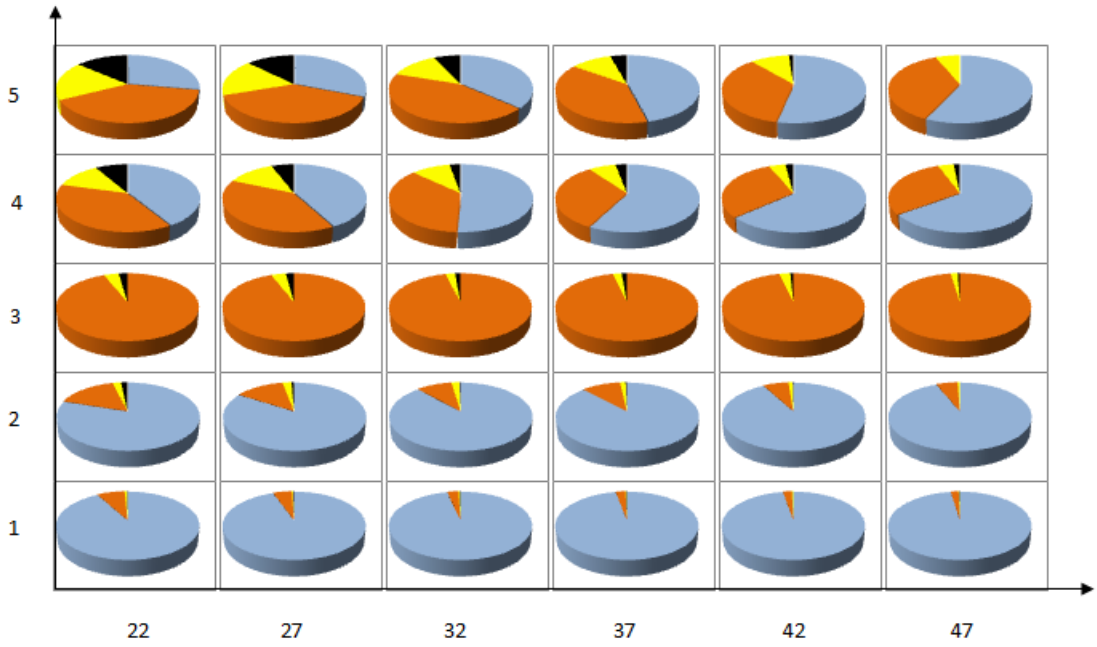


Figure 4.1: Coefficient Indices Pie Chart

1. Classification of coefficient Indices;
2. Binarization;
3. Context Modeling for proposed new Syntax
4. Binary arithmetic coding;

Binarization and Binary Arithmetic Coding steps are exactly same as defined in H.264 CABAC and illustrated in Chapter 3.

### 4.2.1 Classification of Coefficient Indices

Typically adjacent coefficients Indices of a block are correlated which can be coded together to achieve better compression compared to coding them independently. The number of coefficients that can be grouped/coded together largely depends on the video statistics.From the statistics collected from wide range of sequence at different operating QP's and picture types, below classification/grouping of the coefficient indices are proposed.

#### Coefficient Index Block (CIB)

All the coefficient Indices of the block are coded together as a single bin element. Below we explain the process taking an example. Consider a block with four significant coefficients as shown in the table 4.1. As shown in the table, the significant coefficients at scanning position 11, 8, 4, 0 that form a block are coded together.

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index | 1 | 1 | 1 | 1 |
| MSCF-CABAC Blocks | Coded Together | | | |
| H.264 CABAC | Coded | Coded | Coded | Coded |

Table 4.1: Example coding behavior of MSCF-CABAC CIB Block vs H.264 CABAC

### Coefficient Index Pairs (CIP)

The indices of two consecutive significant coefficients (henceforth called as pairs) of a block are coded together as one Bin element. Below we explain the process by taking an example. Consider a block with four significant coefficients as shown in the table 4.2. As shown in the table, the significant coefficients at scanning position 11 and 8 that form a pair are coded together as one bin symbol.

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index | 4 | 2 | 1 | 1 |
| MSCF-CABAC Pairs | Coded Together | | Coded Together | |
| H.264 CABAC | Coded | Coded | Coded | Coded |

Table 4.2: Example coding behavior of MSCF-CABAC Pairs vs H.264 CABAC

Typically the pairs in a block follow specific characteristics which enables us to better fit them linearly. It may be worth noting that the proposed MSCF-CABAC algorithm support coding together of either pair or an entire block which we feel is a good trade-off between Compression and complexity though extending to coding more than two coefficient indices together is a possible design but out of scope of this thesis.

### Single Coefficient Index (SCI)

For special case of the blocks with only one significant Coefficient or the Last significant coefficient in the reverse scanning order when the number of significant coefficients are odd, these coefficients marked in ***bold-italic*** are coded independently as in H.264 CABAC.

Example in table 4.3 shows such cases for 2 blocks:

| Scanning Position | ***11*** | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index Block 1 | ***3*** | - | - | - |
| Absolute Quantized coefficient Level / Index Block 2 | ***4*** | 2 | 1 | - |

Table 4.3: Single Coefficient Index

In the section 4.2.2, we define in detail new syntax elements that define the classification.

### 4.2.2 New Coding Syntax

Following information can be coded to represent significant coefficients

1. **Order_block**: This binary syntax element is coded to convey to the decoder if the absolute value of all the significant coefficients in the block are equal to one. If the absolute value of all significant coefficients in the block happens to be one, the decoder skips the decoding of the block and derives/sets the value of significant coefficient (the position of significant coefficient from the significant coefficient Map) to "one".

As shown in Table 4.4 and Table 4.5, the value of Order_block is set to "one" if all the absolute value of all the significant coefficient in a block is equal to "one" else is set to zero.

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index | 1 | 1 | 1 | 1 |
| Order_block | 1 | | | |

Table 4.4: Significant coefficient in a block and Order Block:1

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index | 1 | 2 | 1 | 1 |
| Order_block | 0 | | | |

Table 4.5: Significant coefficient in a block and Order Block:0

Typically in real life sequences we can find majority of the blocks with all their significant coefficients equal to one. The table 4.6 summarizes the probability of occurrence of such block for ten sequences. It can be seen that the average occurrence of Order_block is greater than 80% for QP 22 and monotonically increases as QP increases as more and more coefficient tend to become one. Also, the probabilities for P-pic and B-pic types follow a similar trend as I-pic type.

2. **Order_pair**: Order_pair syntax elements are similar to order_block but signifies per pair(2 bins) to decoder if the absolute value all the significant coefficients in the pair are equal to one. If the absolute value of all significant coefficient in the pair happens to be one, the decoder skips the decoding of the pair and derives / sets the value of significant coefficient (the position of significant coefficient from the significant coefficient Map) to

| **Streams (I-Pic Analysis)** | **QP 22 In %** | **QP 27 In %** | **QP 32 In %** | **QP 37 In %** | **QP 42 In %** | **QP 47 In %** |
|---|---|---|---|---|---|---|
| BasketballDrive | 85.08 | 87.14 | 93.45 | 96.25 | 97.79 | 98.28 |
| BlowingBubbles | 70.27 | 76.66 | 84.4 | 89.66 | 95.74 | 96.89 |
| BQMall | 73.33 | 77 | 81.36 | 86.17 | 92.01 | 94.35 |
| BQSquare | 82.43 | 81.94 | 77.61 | 77.98 | 78.73 | 81.7 |
| BQTerrace | 82.01 | 78.53 | 80.21 | 83.68 | 88.85 | 91.35 |
| Cactus | 76.04 | 80.14 | 86.05 | 90.13 | 94.51 | 95.71 |
| ChinaSpeed | 83.72 | 88.26 | 89.06 | 88.6 | 89.88 | 90.19 |
| Kimono1 | 78.22 | 86.08 | 93.96 | 97.21 | 98.85 | 99.16 |
| ParkScene | 74.08 | 78.22 | 86.38 | 91.48 | 95.15 | 96.47 |
| SlideShow | 87.07 | 89.63 | 89.72 | 85.26 | 85.19 | 87.22 |
| Tennis | 78.9 | 84.1 | 91.9 | 95.7 | 97.5 | 98.09 |
| vidyo1 | 82.37 | 83.69 | 86.86 | 91.39 | 95.34 | 96.09 |

Table 4.6: Percentage of occurrence of Order_block for I-pictures

"one". This syntax is coded only if Order_block is zero. Also if there are less than three significant coefficients then this syntax is same as Order_block and hence not coded.

As shown in the example (Table 4.7), the value of ***Order_pair*** is set to "one" if all the absolute value of all the significant coefficient in a pair is equal to "one" else is set to zero.

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level/Index | 1 | 2 | 1 | 1 |
| Order_pair | 0 | | 1 | |

Table 4.7: Significant coefficient in a pair and Order Pair

Typically in real life sequences we can find majority of the pairs with all their significant coefficients equal to one. The table 4.8 summarizes the probability of occurrence of such block for ten sequences.

3. **Primitive_Model**: This syntax element is coded to convey the decoder that the absolute value of the syntax elements in the pair can be linearly fitted or precisely conveyed to the decoder if the coefficient indices in the pair are either (1,2) or (2,1). If the absolute value of all significant coefficient in the pair happens to be one of above values, the decoder skips the decoding of the pair and derives/sets the value of significant coefficient (the

| **Streams (I-Pic Analysis)** | **QP 22 In %** | **QP 27 In %** | **QP 32 In %** | **QP 37 In %** | **QP 42 In %** | **QP 47 In %** |
|---|---|---|---|---|---|---|
| BasketballDrive | 72.96 | - | 73.52 | 81.67 | 87.05 | 86.82 |
| BlowingBubbles | 69.48 | 72.04 | 78.49 | 80.32 | 90.82 | 92.08 |
| BQMall | 65.03 | 67.26 | 73 | 77.59 | 84.02 | 87.74 |
| BQSquare | 80.66 | 73.09 | 70.94 | 75.09 | 77.38 | 79 |
| BQTerrace | 78.35 | 71.7 | 72.77 | 75.7 | 80.86 | 82.83 |
| Cactus | 67.68 | - | 72.9 | 76.35 | 82.83 | 84.71 |
| ChinaSpeed | 73.06 | 74.28 | 72.87 | 73.98 | 79.03 | 80.05 |
| Kimono1 | 63.24 | 67.95 | 71.24 | 68.13 | 68.26 | 67.6 |
| ParkScene | 67.58 | 70.98 | 77.37 | 79.05 | 81.45 | 80.47 |
| SlideShow | 70.98 | 64.47 | 66.15 | 69.21 | 76.99 | 81.57 |
| Tennis | 66.69 | 71.15 | 78.57 | 79.58 | 81.57 | 83.63 |
| vidyo1 | 68.01 | 63.9 | 66.39 | 70.75 | 80.42 | 81.34 |

Table 4.8: Percentage of occurrence of Order pair for I-pictures

position of significant coefficient from the significant coefficient Map) to either (1,2) or (2,1) depending on the value of syntax "Slope" detailed in table 4.9. This syntax is coded only if **Order_block** and **order_sub** are zero.

As shown in example (Table 4.9), the value of **Primitive_Model** is set to "one" if the absolute value of the syntax elements in the pair follow a pattern of (1,2) or (2,1) else is set to zero.

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index | 4 | 2 | 1 | 2 |
| Order_pair | 0 | | 1 | |

Table 4.9: Primitive Model: Example

The table 4.10 summarizes the probability of occurrence of such block for ten sequences.

4. **Slope**: This syntax element conveys to the decoder the slope of the absolute value of significant coefficients in the pair. If the index of the coefficients in the pairs are in the increasing order, the syntax Slope is set to one else is set to zero (Table 4.11).

The typical probability of the slope for a set of 10 sequences are detailed in Table 4.12.

| **Streams (I-Pic Analysis)** | **QP 22 In %** | **QP 27 In %** | **QP 32 In %** | **QP 37 In %** | **QP 42 In %** | **QP 47 In %** |
|---|---|---|---|---|---|---|
| BasketballDrive | 17.95 | 0 | 18.22 | 13.79 | 9.47 | 9.92 |
| BlowingBubbles | 19.32 | 17.91 | 15.25 | 15.24 | 8.07 | 7.1 |
| BQMall | 20.67 | 19.63 | 17.6 | 15.8 | 12.12 | 9.71 |
| BQSquare | 12.6 | 16.11 | 18.1 | 15.08 | 15.85 | 14.98 |
| BQTerrace | 13.38 | 17.48 | 17.79 | 16.36 | 13.96 | 12.9 |
| Cactus | 20.66 | 0 | 17.78 | 16.24 | 12.42 | 11.47 |
| ChinaSpeed | 16.61 | 14.28 | 14.91 | 15.07 | 13.17 | 12.76 |
| Kimono1 | 24.76 | 22.15 | 21.04 | 24.03 | 22.91 | 24.58 |
| ParkScene | 20.01 | 18.82 | 15.62 | 15.09 | 13.78 | 15.26 |
| SlideShow | 15.62 | 15.76 | 13.5 | 14.56 | 14.06 | 11.51 |
| Tennis | 20.8 | 19.42 | 14.95 | 14.58 | 14.72 | 13.17 |
| vidyo1 | 18.7 | 21.21 | 21.72 | 19.58 | 14.59 | 13.6 |

Table 4.10: Percentage of occurrence of primitive Model for I-pictures

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index | 4 | 2 | 1 | 2 |
| Slope | 0 | | 1 | |

Table 4.11: Significant coefficients in the pair: Slope

5. **Offset**: This syntax element conveys to the decoder the residual value of the coefficient index that needs to added.

For a coefficient pair, the offset is computed by the encoder as follows:

$$Offset\_least\_significant\_index\_in\_pair = least\_significant\_index\_in\_pair - 1$$
$$Offset\_most\_significant\_index\_in\_pair = most\_significant\_index\_in\_pair - Offset\_least\_significant\_index\_in\_pair$$

| Streams (I-Pic Analysis) | QP 22 | | QP 27 | | QP 32 | | QP 37 | | QP 42 | | QP 47 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Slope** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |
| BasketballDrive | 5.61 | 19.11 | 0 | 0 | 6.56 | 17.13 | 4.46 | 12.21 | 2.52 | 8.9 | 1.83 | 9.59 |
| BlowingBubbles | 7.74 | 19.61 | 7.25 | 17.58 | 4.87 | 14.28 | 5.45 | 11.91 | 1.53 | 6.82 | 1.37 | 5.74 |
| BQMall | 10.21 | 20.86 | 9.27 | 19.5 | 8.31 | 15.27 | 6.9 | 13.14 | 4.25 | 10.44 | 2.67 | 8.64 |
| BQSquare | 6.36 | 10.71 | 7.88 | 15.8 | 9.7 | 16.02 | 7.45 | 13.6 | 5.58 | 14.44 | 5.34 | 13.17 |
| BQTerrace | 6.54 | 12.74 | 8.38 | 16.61 | 8.02 | 16.04 | 6.34 | 15.06 | 4.62 | 12.16 | 4.04 | 10.89 |
| Cactus | 7.95 | 21.11 | 0 | 0 | 7.21 | 16.71 | 5.43 | 15.3 | 3.18 | 12.43 | 2.21 | 11.83 |
| ChinaSpeed | 7.19 | 17.12 | 7.03 | 15.84 | 8.23 | 14.95 | 8.25 | 13.63 | 5.9 | 12.03 | 6 | 11.64 |
| Kimono1 | 4.78 | 29.61 | 2.82 | 27.28 | 1.88 | 25.38 | 1.01 | 29.69 | 1.19 | 29.12 | 0 | 31.28 |
| ParkScene | 8.35 | 20.19 | 6.97 | 18.81 | 4.6 | 15.51 | 2.52 | 16.35 | 1.6 | 15.49 | 0.93 | 17.41 |
| SlideShow | 6.4 | 19.64 | 10.31 | 20.78 | 12.36 | 15.8 | 9.64 | 15.38 | 7.45 | 11.52 | 4.9 | 9.9 |
| Tennis | 7.58 | 22.44 | 5.88 | 20.3 | 3.57 | 15.84 | 2.14 | 16.25 | 1.38 | 16.21 | 0.6 | 15.37 |
| vidyo1 | 6.31 | 22.54 | 7.12 | 25.25 | 7.16 | 23.29 | 4.94 | 21.12 | 2.62 | 15.1 | 1.37 | 14.64 |

Table 4.12: Percentage of occurrence of Slope for I-pictures

## 4.3 Context Modeling

For most sequences and coding conditions some of the statistics are very similar. To keep the number of different context models used for coefficient coding reasonably small, the contexts are classified into categories specified below. For each of these categories, a special set of context models is used for all syntax elements related to residual data.

1. **Ctx_Order_block**: Order_block represents whether all the significant coefficients in the block are equal to "one". The chosen probability models for this syntax element depends on the bin index. The context index increment for Ctx_Order_block for a given block is given by

$$Ctx\_Order\_block(C) = (Ctx\_Order\_block(A) \,!= 0)\,?\,0 \,:\, 1 + 2 * (Ctx\_Order\_block(B) \,!= 0)\,?\,0 \,:\, 1$$

   where Ctx_Order_block(A) and Ctx_Order_block(B) represent the Order_block pattern to the left block A and on the top block B respectively.

The Reason behind predicting the context increment from the neighboring blocks is that typically in most of the real-life video sequences the blocks are correlated both spatially and temporally to neighboring blocks.

2. **Ctx_Order_pair**: Order_pair represents whether all the significant coefficients in the pair are equal to "one". The context index increment for Ctx_Order_block for a given block is given by following set of rules. The context increment is initialized to one at the start and continues to be incremented until Order_pair is equal to "zero". If the Order_pair is "zero", the context index is set to Zero is always chosen for further pairs.

   Context Modeling for proposed Order_Pair.

   Order_Pair Context is modeled as shown in Eq: 2

   $ctxOrderPairInc = ((numDecodOrderPair! = 0)?0 : Min(4, 1+numDecodOrderPair))$

   Typically for majority of real life video sequences, the coefficients increase monotonically in the reverse encoding order. So if we encounter indices $> 1$ at a specific scanning position it is highly likely the indices succeeding this scanning position is going to be greater than one. Hence a separate context is used for these cases.

3. **Ctx_PM**: Primitive_mode represents whether that the absolute value of the syntax elements in the pair follow a pattern of (1,2) or (2,1). There is only one context increment for Ctx_PM as the occurrence is Random across the pairs.

4. **Ctx_Slope**: There is only one context increment define for Ctx_Slope.

5. **Ctx_Offset(2)**: Coded as defined in H.264 CABAC.

## 4.4 Handling Special Cases

### 4.4.1 Last Significant coefficient in reverse Scanning order of the Blocks

Here we discuss a special case of the blocks with only one significant Coefficient or the Last significant coefficient in the reverse scanning order when the number of significant coefficients are odd. Example (Table 4.13) shows such cases for two blocks:

The Table 4.14 summarizes the probabilities of Coefficient index for such cases for different $Q_p$ range and picture types (I, P, B).

| Scanning Position | 11 | 8 | 4 | 0 |
|---|---|---|---|---|
| Absolute Quantized coefficient Level / Index Block 1 | 3 | - | - | - |
| Absolute Quantized coefficient Level / Index Block 2 | 4 | 2 | 1 | - |

Table 4.13: First significant coefficient in the reverse scanning order

| **Streams (I-Pic Analysis)** | **QP 22** | | **QP 27** | | **QP 32** | | **QP 37** | | **QP 42** | | **QP 47** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Slope** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |
| BasketballDrive | 95.36 | 4.64 | 0 | 0 | 94.82 | 5.18 | 96.04 | 3.96 | 97.49 | 2.51 | 98.36 | 1.64 |
| BlowingBubbles | 94.01 | 5.99 | 94.17 | 5.83 | 95.42 | 4.58 | 96.32 | 3.68 | 98.53 | 1.47 | 99.11 | 0.89 |
| BQMall | 91.33 | 8.67 | 92.59 | 7.41 | 93.02 | 6.98 | 94.98 | 5.02 | 97.22 | 2.78 | 97.6 | 2.4 |
| BQSquare | 93.06 | 6.94 | 91.63 | 8.37 | 92.06 | 7.94 | 94.09 | 5.91 | 97.32 | 2.68 | 96.3 | 3.7 |
| BQTerrace | 95.6 | 4.4 | 94.57 | 5.43 | 94.92 | 5.08 | 95.65 | 4.35 | 96.29 | 3.71 | 96.25 | 3.75 |
| Cactus | 94.83 | 5.17 | 0 | 0 | 95.7 | 4.3 | 96.05 | 3.95 | 96.83 | 3.17 | 97.19 | 2.81 |
| ChinaSpeed | 91.57 | 8.43 | 92.07 | 7.93 | 91.86 | 8.14 | 92.22 | 7.78 | 93.78 | 6.22 | 93.5 | 6.5 |
| Kimono1 | 94.75 | 5.25 | 95.43 | 4.57 | 96.41 | 3.59 | 98.03 | 1.97 | 97.12 | 2.88 | 97.18 | 2.82 |
| ParkScene | 94.77 | 5.23 | 95.66 | 4.34 | 96.28 | 3.72 | 97.32 | 2.68 | 98.03 | 1.97 | 97.75 | 2.25 |
| SlideShow | 91.86 | 8.14 | 87.44 | 12.56 | 85.25 | 14.75 | 89.88 | 10.12 | 93.4 | 6.6 | 94.53 | 5.47 |
| Tennis | 94.97 | 5.03 | 95.64 | 4.36 | 96.61 | 3.39 | 97.09 | 2.91 | 97.95 | 2.05 | 98.59 | 1.41 |
| vidyo1 | 94.27 | 5.73 | 94.1 | 5.9 | 93.65 | 6.35 | 93.97 | 6.03 | 96.42 | 3.58 | 96.43 | 3.57 |

Table 4.14: Probabilities of Coefficient index for picture type:I, different $Q_p$ range

If there is only one significant coefficient in the block, the coding is done similar to H.264 CABAC.

### 4.4.2 Even and Odd pair Encoding

If the number of significant coefficients are even in a block, then all coefficient indices are coded as pairs as detailed in the section 4.2. But if there are odd significant coefficients in a block, the Last significant coefficient is coded separately as detailed in section 4.4.1 and rest are coded in pairs.

## 4.5 Encoder/Decoder Psuedocode Syntax

The Encoder/Decoder pseudocode/syntax is illustrated in the fig.4.2.

Further, a simplified Psuedocode Syntax is also illustrated in the fig.4.3. In the simplified proposal the number of contexts are significantly reduced and in comparison to H264 CABAC only six contexts are additional. The performance simplified MSCF-CABAC is almost similar to MSCF-CABAC. Henceforth all the results comparison with respect to H264 CABAC are discussed with simplified MSCF-CABAC.

In the simplified pseudocode syntax version of MSCF-CABAC, a pair of bins (two bins in this case) are coded together. If the significant coefficient indices of all indices in the air are equal to 1, only the sign of the indices are decoded and the absolute value of the indices are derived as "one". The position of the indices are derived from the significant Map. If all the indices in the pair are not equal to one, a slope is coded indicating whether the consecutive significant coefficients are increasing or decreasing. Typically for CABAC it is always better to code the lower significant code first followed by higher as in that case contexts modeling is efficient. After coding the lower significant index of the pair in the same fashion is done in H264 CABAC, the delta of the higher significant index is coded instead of index. As a special case if the lower significant index happens to be 1, higher significant index -2 is coded to avoid redundancy. For the cases when the significant coefficients in block are odd, then the Last significant coefficient in the inverse scanning order is coded in the similar fashion as H264.

## 4.6 Comparison with H.264 CABAC

The Proposed Multi-symbol Curve-fit CABAC was incorporated into the encoder and decoder of the JM 18.3 software. It was applied to the coding of significant coefficients. Experiments were performed using common conditions specified in [77] for nine different set of sequences. Table 4.15 lists the comparison results of proposed MSCF-CABAC vS H264 CABAC for three different operating points (High Rate, Medium Rate and Low Rate). The QP values used for High rate range from 22-26, for Medium rate from 28-32 and for low Rate from 34-38.

- **Compression efficiency**

  Average Percentage Bitrate gain for same PSNR of proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 0.94%, 0.68% and

| residual_block_cabac(coeffLevel, maxNumCoeff) { | C | Descriptor |
|---|---|---|
| coded_block_flag | 3 \| 4 | ae(v) |
| If( coded_block_flag ) { | | |
| numCoeff = maxNumCoeff | | |
| i = 0 | | |
| do { | | |
| significant_coeff_flag[ i ] | 3 \| 4 | ae(v) |
| if( significant_coeff_flag[ i ] ) { | | |
| last_significant_coeff_flag[ i ] | 3 \| 4 | ae(v) |
| if( last_significant_coeff_flag[ I ] ) { | | |
| numCoeff = i + 1 | | |
| For( j = numCoeff; j < maxNumCoeff; j++ ) | | |
| coeffLevel[ j ] = 0 | | |
| } | | |
| } | | |
| i++ | | |
| } while( i < numCoeff - 1 ) | | |
| for( i = numCoeff - 1; I >= 0; i-- ) | | |
| Coeff_sign_flag[i] | 3 \| 4 | ae(v) |
| Order_block | 3 \| 4 | ae(v) |
| if(order_block == 0) | | |
| { // | | |
| if(numcoeff & 0x01) | | |
| { | | |
| if(numcoeff == 1) | | |
| coeff_abs_level_minus1[ 0 ] | 3 \| 4 | ae(v) |
| Else | | |
| { | | |
| Order_pair | 3 \| 4 | ae(v) |
| if(order_pair == 0) | | |
| coeff_abs_level_minus1[ 0 ] | 3 \| 4 | ae(v) |
| } //numcoeff != 1 | | |
| } // numcoeff is odd | | |
| else | | |
| { //numcoeff is even | | |
| for(i=0;i<numcoeff/2;i++) | | |
| Order_pair[i] | 3 \| 4 | ae(v) |
| if(order_pair == 0) | | |
| { | | |
| primitive_model[i] | 3 \| 4 | ae(v) |
| if(primitive_model == 0) | | |
| { | | |
| Slope[i] | 3 \| 4 | ae(v) |
| Offset_low[i] | 3 \| 4 | ae(v) |
| Offset_high[i] | 3 \| 4 | ae(v) |
| } // primitive_model == 0 | | |
| } //order_pair == 0 | | |
| } //numcoeff is even | | |
| } //order_block == 0 | | |
| } // coded_block_flag | | |
| } // | | |

Figure 4.2: Syntax Decoding Order for MSCF CABAC

| residual_block_cabac(coeffLevel, maxNumCoeff) { | C | Descriptor |
|---|---|---|
| **coded_block_flag** | 3 \| 4 | ae(v) |
| if( coded_block_flag ) { | | |
| numCoeff = maxNumCoeff | | |
| i = 0 | | |
| do { | | |
| **significant_coeff_flag[** i **]** | 3 \| 4 | ae(v) |
| if( significant_coeff_flag[ i ] ) { | | |
| **last_significant_coeff_flag[** i **]** | 3 \| 4 | ae(v) |
| if( last_significant_coeff_flag[ i ] ) { | | |
| numCoeff = i + 1 | | |
| } | | |
| } | | |
| i++ | | |
| } while( i < maxNumCoeff - 1 ) | | |
| for( i = numCoeff - 1; i>= 0; i-- ) { | | |
| **order_pair** | 3 \| 4 | ae(v) |
| if(order_pair) { | | |
| **coeff_sign_flag[i]** | 3 \| 4 | ae(v) |
| **coeff_sign_flag[i-1]** | 3 \| 4 | ae(v) |
| } else { | | |
| **Slope** | 3 \| 4 | ae(v) |
| if(Slope) { | | |
| **coeff_abs_level_minus1[i]** | 3 \| 4 | ae(v) |
| **coeff_abs_level_minus1[i-1]** | 3 \| 4 | ae(v) |
| Coeff[i-1]+=(Coeff[i]==1)? (Coeff[i]) : (Coeff[i] -1) | | |
| } else { | | |
| **coeff_abs_level_minus1[i-1]** | 3 \| 4 | ae(v) |
| **coeff_abs_level_minus1[i]** | 3 \| 4 | ae(v) |
| Coeff[i]+=Coeff[i-1] | | |
| **}** | | |
| **coeff_sign_flag[i]** | 3 \| 4 | ae(v) |
| **coeff_sign_flag[i-1]** | 3 \| 4 | ae(v) |
| **if(i == 0) {** | | |
| **coeff_abs_level_minus1[i]** | 3 \| 4 | ae(v) |
| coeff_sign_flag[i] | | |
| **}** | | |
| } | | |
| } | | |
| } | | |

Figure 4.3: Simplified Syntax Decoding Order for MSCF CABAC

1.30% respectively. Peak Percentage Bitrate gain for these sequences in same order are 5.87%, 3.91%, 5.51%.

- **Throughput improvement/Bin Reduction**

  Average Percentage Total Picture Bins Reduction at the above mentioned operating points for proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 1.09%, 0.80% and 1.34% respectively. Average Percentage Total Picture Bins Reduction for these sequences in same order are 5.92%, 3.22%, 4.72%.

Total coefficients Bin Reduction (significant Map, Last Coefficient Map and Residual Level and sign coding).

Average Percentage Total coefficients Bin Reduction at the above mentioned operating points for proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 2.21%, 2.11% and 3.53% respectively. Peak Percentage Bitrate gain for these sequences in same order are 9.86%, 3.86%, 5.67%.

| **Stream** | **Rate Points** | **PSNR** | **Curve fitting Based Bitrate (kb/sec)** | **H264 CABAC Bitrate (kb/sec)** | **% Bit Rate Gain** | **% Picture Bin Reduction** | **% Coeff Bin Reduction** |
|---|---|---|---|---|---|---|---|
| Cactus (1920x1080) | High Rate | 38.1 | 19679 | 19776 | 0.49 | 0.83 | 1.64 |
| | Mid Rate | 33.1 | 3174 | 3178 | 0.13 | 0.35 | 1.32 |
| | Low Rate | 31.1 | 1922 | 1919 | -0.16 | 0.26 | 2.46 |
| BlowingBubbles (416x240) | High Rate | 37.8 | 1533 | 1538 | 0.33 | 0.67 | 1.44 |
| | Mid Rate | 31.5 | 382.44 | 385.82 | 0.88 | 1.35 | 1.97 |
| | Low Rate | 28.2 | 171.98 | 172.46 | 0.28 | 0.61 | 3.09 |
| BQTerrace (1920x1080) | High Rate | 39.8 | 18982 | 19003 | 0.11 | 0.29 | 0.69 |
| | Mid Rate | 34.2 | 6161 | 6169 | 0.13 | 0.47 | 0.73 |
| | Low Rate | 28.8 | 2737 | 2741 | 0.15 | 0.14 | 0.46 |
| Kieba (832x480) | High Rate | 37.8 | 7282.51 | 7325.59 | 0.59 | 0.82 | 1.24 |
| | Mid Rate | 32.4 | 2646.17 | 2748.55 | 3.87 | 4.11 | 4.94 |
| | Low Rate | 27.4 | 880.9 | 887.21 | 0.72 | 1.60 | 4.17 |

Continued on next page

| Stream | Rate Points | PSNR | Curve fiting Based Bitrate (kb/sec) | H264 CABAC Bitrate (kb/sec) | % Bit Rate Gain | % Picture Bin Reduction | % Coeff Bin Reduction |
|---|---|---|---|---|---|---|---|
| Kimono (1920x1080) | High Rate | 41.2 | 14696 | 14723 | 0.18 | 0.35 | 0.53 |
| | Mid Rate | 35.6 | 3730 | 3765 | 0.94 | 1.16 | 3.66 |
| | Low Rate | 32.1 | 1748 | 1744 | -0.23 | 0.12 | 4.72 |
| RaceHorses (832x480) | High Rate | 39 | 8919 | 8946 | 0.30 | 0.59 | 0.90 |
| | Mid Rate | 33.2 | 2876 | 2932 | 1.95 | 2.13 | 3.58 |
| | Low Rate | 28.1 | 1003 | 1016 | 1.30 | 1.39 | 2.73 |
| ParkScene (1920x1080) | High Rate | 38.5 | 12115 | 12142 | 0.22 | 0.54 | 1.22 |
| | Mid Rate | 37.4 | 8958 | 9027 | 0.77 | 1.04 | 1.95 |
| | Low Rate | 31.9 | 2287 | 2356 | 3.02 | 2.65 | 5.67 |
| BasketBall (1920x1080) | High Rate | 38.6 | 9972 | 10034 | 0.62 | 0.28 | 1.67 |
| | Mid Rate | 34.2 | 2712 | 2710 | -0.07 | 0.06 | 2.85 |
| | Low Rate | 32.3 | 1804 | 1832 | 1.55 | 1.65 | 5.41 |
| **Average % Gain & Bit Reduction** | High Rate | | | | 0.36 | 0.55 | 1.17 |
| | Mid Rate | | | | 1.07 | 1.33 | 2.63 |
| | Low Rate | | | | 0.83 | 1.05 | 3.59 |
| **Peak % Gain & Bit Reduction** | High Rate | | | | 0.62 | 0.83 | 1.67 |
| | Mid Rate | | | | 3.87 | 4.11 | 4.94 |
| | Low Rate | | | | 3.02 | 2.65 | 5.67 |

Table 4.15: MSCF-CABAC vS H.264 CABAC

### 4.6.1 Test Conditions

H.264 JM 18.3 reference software was used for comparison analysis of all the test sequences. The reference PSNR's and Bits per frame with H.264 were generated at constant QP's with best quality configurations for range of bitrate's (high, medium, low).

The following are the key features/tools sets (Table 4.16) that were used to generate the reference streams.

| **H.264 Key Features / Tool Sets** | **Configuration Setting** |
|---|---|
| Profile | MAIN (esp only $4 \times 4$ transform) |
| NumberReferenceFrames | 5 |
| ME/MD Distortion | Hadamard |
| CABAC | ON |
| Loop Filter | ON |
| RDOptimization on Mode-decision | ON |
| RDOptimization on Quantization | OFF |
| AdaptiveRounding | OFF |
| Slices | OFF |
| Qmatrix | FLAT |

Table 4.16: Key tools / features used to generate the reference streams

| **Test Sequences** | **Spatial Resolution** |
|---|---|
| Cactus | 1920x1080 |
| BlowingBubbles | 416x240 |
| BQTerrace | 1920x1080 |
| Kieba | 832x480 |
| Kimono | 1920x1080 |
| RaceHorses | 832x480 |
| ParkScene | 1920x1080 |
| BasketBall | 1920x1080 |

Table 4.17: List of test sequences

A variety of video sequences with very different spatio-temporal characteristics, as well as content at different resolutions, were selected for this evaluation. The content tested included movie and sports segments, trailer like clips with fast scene cuts and other types of transitions, as well as standard and well known clips used by a variety of organizations for standardization purposes. The list of all the sequences and the target Qp's used can be seen in Table 4.17.

The Rate gain mainly depends on the distribution of quantized coefficients. From the

Quantized Coefficient data as shown in Appendix A it can be inferred that for typical real life sequences majority of the Quantized DCT coefficient's are not only distributed with absolute value of 1 and 2's but also exhibit statistical dependencies which can be exploited if the coefficients are coded jointly. In this case we exploit using simple zero order and first order curve fitting. H.264 CABAC encodes each coefficient separately thereby doesn't have the framework to exploit high order dependencies.

Also it should be noted that there might be hypothetical distributions of quantized coefficients for which MSCF-CABAC might not be efficient. For example, if the none of the coefficients pairs in the block cannot be fit as Zero-order or first-order then MSCF-CABAC is less efficient to H.264 CABAC. The possibility of such cases for the tested sequences have been found to 0.03% and the rest 99.77%.

Further to this it can be observed that the proposed Syntax elements operate on two or more coefficients at the same time and hence the amount of bins that needs to be encoded are less.

# Chapter 5

# Soft Decision Quantization Framework for MSCF-CABAC

In this chapter, we propose a Soft Decision Quantization (SDQ) design framework for the proposed MSCF-CABAC Model. Based on SDQ instead of conventional HDQ, the proposed framework allows us to optimize the significant coefficients by minimizing the actual RD cost based on the final reconstruction error and the entire coding rate. As seen in section 5, MSCF-CABAC particularly focus on efficiently coding the Significant coefficients, the SDQ for MSCF-CABAC is also designed for Significant coefficient's where all the coefficient rate of the block is concentrated.

In the following section, we first review RD optimization methods in the literature. Then, an SDQ scheme is introduced based on reviews of theoretical results on universal fixed-slope lossy coding. We then discuss briefly the design challenges of Lagrangian multiplier based RD optimization schemes. Further we propose a simplified graph based SDQ optimization to optimize the significant coefficients in the block based on MSCF-CABAC encoding rules. We conclude this section by discussion the optimality, complexity and comparing both objective and subjective comparison over H.264 JM.

## 5.1 Previous Rate-Distortion Optimization Work

Due to non-stationary nature of video content, it is extremely difficult to come up accurately with a theoretical R-D function by estimating a statistical model for video data[38, 39]. To avoid this model mismatch, typically operational RD functions are used, which

are computed based on the data to be compressed. This thesis is focused on developing operational RD methods. Below we discuss some of the previous work done related to Rate-Distortion optimization in literature.

Ramchandran et al. [61] developed an operational rate distortion framework for efficiently distributing bit budget among temporal and spatial coding methods for MPEG video compression. The rate distortion optimization problem was converted into a generalized bit allocation task. The exponential complexity issue was tackled by utilizing a monotonicity property of operational rate distortion curves for dependent blocks/frames. The monotonicity property was constructed based on an assumption that rate distortion performance for coding one frame was monotonic in the effectiveness of prediction, which depended on the reproduction quality of reference frames. A pruning rule was then developed to reduce search complexity based on the monotonicity property. Generally speaking, the above assumption implies a linear relationship between distortion and residual coding rate. In fact, the above assumption is valid to only some extent and consequently, it may not be able to find the optimal solution, either due to the approximation of the coding rate or because of the exponential complexity.

Wiegand et al. proposed a simple and effective operational RD method using the generalized Lagrangian multiplier method [15], for Hybrid Video coding [71, 75, 77]. The mode selection for motion estimation was conducted based on the actual RD cost on a macro-block basis. That is, for a given prediction mode, motion estimation is optimized based on approximate actual RD cost, as follows (Equation 5.1):

$$(f, v) = arg\, min_{f,v}\, d\left(x, p\left(m, f, v\right)\right) + \lambda \cdot \left(r\left(v\right) + r\left(f\right)\right) \tag{5.1}$$

where x stands for the original image block, p(m, f, v) is the prediction with given prediction mode m, reference index f, and motion vector v, $d(\cdot)$ is a distortion measure, r(v) is the number of bits for coding v, r(f) is the number of bits for coding f, and $\lambda$ is the Lagrangian multiplier.

Wen et. al [74] proposed an operational RD method for residual coding optimization in H.263+ using a trellis-based soft decision quantization design. In H.263+, residuals are coded with run-length codes followed by variable length coding (VLC). The VLC in H.263+ does not introduce any dependency among neighboring coefficients, while the dependency mainly comes from the run-length code. Therefore, a trellis structure is used to decouple the dependency so that a dynamic programming algorithm can be used to find the optimal path for quantization decisions.

Yang and Yu in [87] proposed an operational RD method for H.264 based on a soft

decision quantization (SDQ) mechanism, which has its root in a fundamental RD theoretic study on fixed-slope lossy data compression. Using SDQ instead of HDQ, a general framework was designed to jointly minimize the actual RD cost on a frame-by-frame basis including motion prediction, quantization, and entropy coding in a hybrid video coding scheme.

Further the following three RD methods were proposed

1. A graph-based algorithm was proposed for SDQ, given motion prediction and quantization step sizes,
2. An algorithm for residual coding optimization, given motion prediction, and
3. An iterative overall algorithm for jointly optimizing motion prediction, quantization, and entropy coding - with these embedded in the indicated order.

Among the three algorithms, the SDQ design was the core, which is developed based on a given entropy coding method, specifically, for context adaptive variable length coding (CAVLC) in H.264 baseline profile and the context adaptive binary arithmetic coding (CABAC) in H.264 main profile, respectively.

Further [67] proposed a more soft decision quantization algorithm that gives an optimized determination of transform coefficient levels by considering temporal dependencies as well. A linear model of inter-frame dependencies and a simplified rate model to formulate an optimization problem was assumed for computing the quantization outputs using a quadratic program, but neglecting specific factors related to the particular entropy coding method that is in use.

The SDQ design is primarily based on the Entropy coding technique that is being used, thus a new design criterion is needed to handle the complexities of specific entropy coding methods for designing SDQ in H.264. In this thesis we propose a simplified SDQ design for the proposed MSCF-CABAC Entropy coding method.

## 5.2 Soft Decision Quantization Overview

Video encoders primarily use rate-distortion optimization to improve quality where decisions have to be made that affect both file size and quality simultaneously. A wide range of RD optimization algorithms using the Lagrange multiplier technique have been proposed in the literature [1][2][3]. Conditions for optimizing the encoder operation are

derived within a rate-constrained product code framework using a Lagrangian formulation. Mathematically, the Rate Distortion minimizes, can be explained as the distortion between the original image X and the thresholded image $\tilde{X}$ (Quantized after prediction) given reconstructed image $\widehat{X}$ subject to a bit budget constraint, i.e (Equation 5.2)

$$Min[D(X,\tilde{X})|\widehat{X}] \text{ subject to } R(\tilde{X}) = R_{budget} \tag{5.2}$$

Alternative unconstrained problem is to minimize

$$J(\lambda) = D(X,\tilde{X}) + \lambda * R(\tilde{X}) \tag{5.3}$$

For a given value of $\lambda$, we can obtain the R($\tilde{X}$) or D(X,$\tilde{X}$) that minimizes J($\lambda$).

Now, we discuss how the above discussed may be used to conduct SDQ in hybrid video coding optimization. Consider a $4 \times 4$ block, with quantized transform coefficient u, prediction mode m, reference index f, motion vector v, and quantization step size q. Its reconstruction is computed by Equation 5.4.

$$\widehat{x} = p\,(m,f,v) + T^{-1}\,(u.q) \tag{5.4}$$

where $p\,(m,f,v)$ is the prediction corresponding to m, f, v and $T^{-1}\,(.)$ is the inverse transform.

Conventionally, the constraint of 5.4 is used to derive a deterministic quantization procedure, i.e. (Equation 5.5) ,

$$HDQ\,(T\,(z)) = round\,([T\,(z) + \delta \cdot q]\,/q) \tag{5.5}$$

which mainly minimizes the quantization distortion d(x , $\hat{x}$), where $z = xp(m,f,v)$. The factor $\delta$ is an offset parameter for adapting the quantization outputs to the source distribution to some extent. There are empirical studies on determining $\delta$ according to the signal statistics to improve the RD compression efficiency[79]. Still, this is an HDQ process.

Typically video coding standards describe only the bit-stream syntax for decoding process. The exact nature of the encoder design is generally left open to user specification. This allows different Rate Distortion optimization algorithms to be applied to generate a standardized video bit stream with a better RD performance. A search for $u$ to minimize the RD cost, i.e. (Equation 5.6),

$$\boldsymbol{u} = arg\, min_u d\left(\boldsymbol{z}, T^{-1}\left(\boldsymbol{u} \cdot q\right)\right) + \lambda . r_\gamma\left(\boldsymbol{u}\right) \tag{5.6}$$

The minimization in (Equation 5.6) is over all possible quantized values. In general, such a ***"u"*** will not be obtained by the hard decision process via (Equation 5.5), and the quantization by (Equation 5.6) is called SDQ [83].

Now, given (m, f, V, q), the problem of finding an optimal SDQ becomes to the minimization problem of Equation 5.7.

$$min_U \| \boldsymbol{Z} - T^{-1}\left(Q^{-1}\left(\boldsymbol{U}\right)\right) \|^2 + \lambda \cdot r\left(\boldsymbol{U}\right) \tag{5.7}$$

where Z is the residual corresponding to given (m, f , V , q). It is easy to see that the exact optimal SDQ solution to 5.7 depends on entropy coding, which determines the rate function $r(\cdot)$. Furthermore, the entropy coding method is application-dependent. Different applications have different entropy coding methods and hence different SDQ solutions. Some early work on practical (optimal or suboptimal) SDQ includes without limitation SDQ in JPEG image coding and H.263+ video coding [6, 17, 81, 34, 59]. In this thesis, we focus on designing SDQ algorithm for the proposed MSCF-CABAC optimization.

For a given value of $\lambda$, we can obtain the $R(\tilde{X})$ or $D(X, \tilde{X})$ that minimizes J($\lambda$). If the value of Lambda becomes small, the value of $R(\tilde{X})$ in J($\lambda$) tends to become large, and vice versa. The solution to the unconstrained Lagrangian cost function for any value of Lambda results in minimum distortion for some rate, the final rate cannot be specified a priori. Often it is desirable to find a particular value for lambda so that upon optimization of (2), the resulting rate closely matches a given rate constraint $R_{budget}$.

## 5.3 Design Challenges of Lagrangian multiplier Optimization

There are two popular ways to solve such R-D problem, i.e. Lagrange multiplier method and dynamic programming, among which the first one is widely used in today's video coding due to its lower cost in computation.

So, as we saw in previous section, the constrained optimization problem can be converted mathematically as, for a fixed quantizer scale, the distortion between the original image X and the thresholded image $\tilde{X}$ given reconstructed image $\hat{X}$ subject to a bit budget constraint, i.e

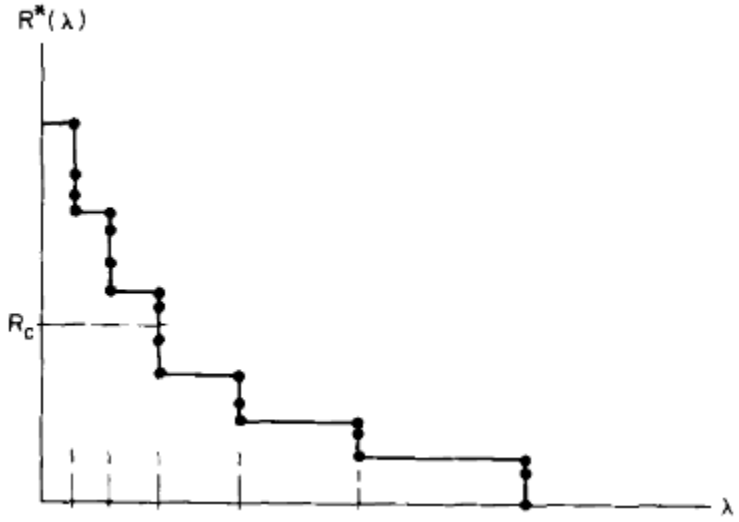


Figure 5.1: R vs Lambda variation

$$Min\left[D\left(X,\tilde{X}\right)|\hat{X}\right] \text{ subject to } R\left(\tilde{X}\right) \leq R_{budget} \tag{5.8}$$

Alternative unconstrained problem is to minimize

$$J\left(\lambda\right) = D\left(X,\tilde{X}\right) + \lambda * R\left(\tilde{X}\right) \tag{5.9}$$

The Figure 5.1 show the variation of R with respect to lambda [28].

It can be seen that as $\lambda$ increases the rate decreases. It shows a decreasing staircase curve where the discontinuities correspond to singular values of $\lambda$. The black dots at the singular points represent multiple values of constraints which correspond to multiple solutions. So by taking a sweep of $\lambda$ over all possible values (all values on the non-negative real line) we can find all the solutions to the unconstrained problem.

### 5.3.1 Empirical Fixed Lambda Method

In 2001, Wiegand and Girod proposed a Lagrange Multiplier selection method [3]. Assuming D to be differentiable everywhere, the minimum of the Lagrangian cost J is given by setting its derivative to zero, i.e.(Equation 5.8),

$$\frac{dJ}{dR} = \frac{dD}{dR} + \lambda = 0 \tag{5.10}$$

Yielding Equation 5.9:

$$\lambda = -\frac{dD}{dR} \tag{5.11}$$

In fact, Equation 5.9 indicates that $\lambda$ corresponds to the negative slope of Rate-Distortion function, that is $\lambda$ can be perfectly determined by the models of R and D. If a sufficiently high rate environment is supposed, the model can be derived as shown in Equation 5.9 according to the typical high-rate approximation curve for entropy-constrained scalar quantization [82] (Equation 5.10),

$$R(D) = alog_2\left(\frac{b}{D}\right) \tag{5.12}$$

where a and b are two constants. For the D model, preserving the same "high rate" assumption, the source probability distribution can be approximated as uniform within each quantization interval [82], leading to Equation 5.11.

$$D = \frac{(2Q)^2}{12} = \frac{Q^2}{3} \tag{5.13}$$

Note that here the quantizer value Q is half the distance of the quantizer reproduction levels. Putting (5.10) and (5.11) into (5.9), the final $\lambda$ can be determined by Equation 5.12.

$$\lambda = -\frac{dD}{dR} = c \cdot Q^2 \tag{5.14}$$

where c is a constant which is experimentally suggested to be 0.85. In summary, this method, or referenced as "HR- $\lambda$" for convenience, is practical and efficient. Therefore it was adopted into the reference software not only by H.263 [87][83], but also by H.264/AVC [84][85], the state-of-art video coding standard. However, its "high rate" assumption is not realistic all the time. Furthermore, $\lambda$ is only related to Q and no property of the input signal is considered, which results in that it can not adapt to different videos dynamically.

### 5.3.2 Bisection Search

The total data rate R is proportional to the Lagrangian multiplier $\lambda$, and thus R can be adjusted to the target rate $R_T$ by searching the optimal Lagrangian multiplier. The optimal Lagrangian multiplier $\lambda$ can be found by using a bisection method [34]. The bisection method uses two previously evaluated Lagrangian multipliers, $\lambda_l$ and $\lambda_h$. Corresponding to $\lambda_l$ and $\lambda_h$, the distortion and the number of bits allocated are denoted by $P_l$, $P_h$, $R_l$, $R_h$ respectively.

The bi-sectional search method lowers the gap between $\lambda_l$ and $\lambda_h$ by computing the following updated Lagrangian multiplier:

$$\lambda_{new} = (P_h - P_l) / (R_h - R_l) \tag{5.15}$$

Then, the total data rate $R_{new}$ corresponding to $\lambda_{new}$ is evaluated. If $R_{new}$ is greater than the rate requirement $R_T$, we update $\lambda_h$ with $R_{new}$ while keeping $\lambda_l$ the same. The opposite update is done if $R_{new}$ is less than $R_T$. An example of the bi-sectional search method is shown in Figure 5.2. These update procedures are repeated until $R_{new}$ equals to either $R_l$ or $R_h$ .

The Fixed lambda method is computationally very inexpensive but works good only for some specific sequences where bit rates are very high. The Bisection search method gives an optimal $\lambda$ but is highly computationally intensive and not feasible for online or real time applications.

For all our simulations the Fixed lambda method was used for its simplicity.

## 5.4 Simplified SDQ Design based on MSCF-CABAC

In this section, we present a Simplified graph-based SDQ algorithm based on MSCF-CABAC for the significant coefficients in the block to solve the SDQ problem of Equation 5.7.

The coefficients in a block can be categorized as Significant and non-significant based on their absolute values. All the non-zero coefficients fall into the category of significant coefficients and needs to be coded while the zero coefficients fall into the category of non-significant coefficients and are not coded. Whether the coefficient is significant or non-significant is indicated in the Significant Map of the block. Also the number of bits needed to encode a block of coefficients are largely concentrated on the significant

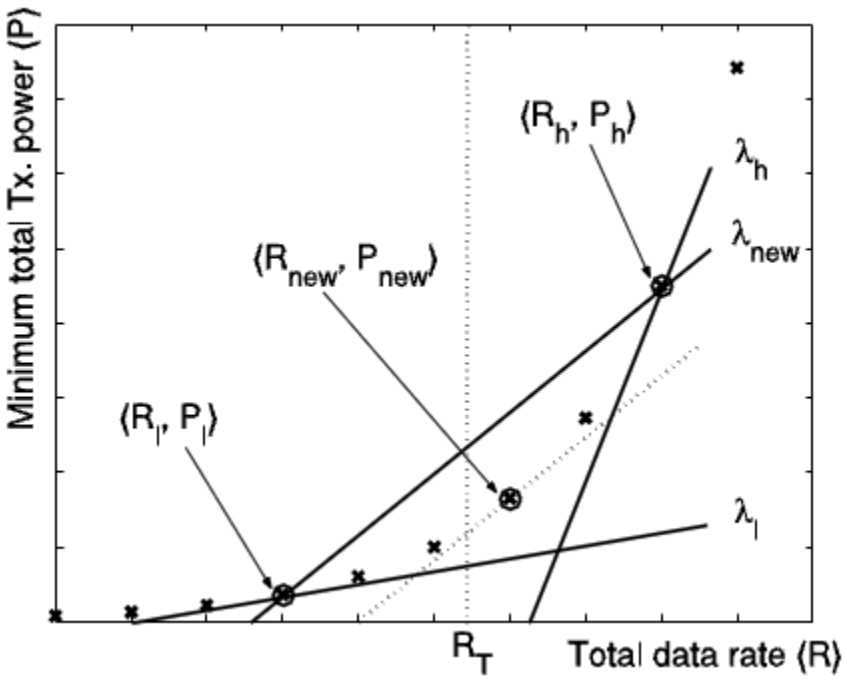


Figure 5.2: Lagrangian selection using Bisection Search

coefficients. MSCF-CABAC particularly focus only on efficiently coding the Significant coefficients, the SDQ for MSCF-CABAC is also designed for Significant coefficient's. Let $U = \{u1, \cdots, uK\}$, be set of significant coefficients in the block with K significant coefficients.

Clearly, for given residual U and quantizer (q), the distortion term in Equation 5.7 is additive within a block and across the block-wise. Also in H.264, encoding of each block depends not only on itself, but also on its two neighboring blocks due to context adaptivity across blocks. However, such dependency is very weak and doesn't affect much towards rate and are therefore, decoupled in the optimization for the whole frame thus enabling block by block optimization.

For all significant coefficients in the block, the optimization problem given in Equation 5.7 now reduces to,

$$\boldsymbol{u} = arg\, min_u\, d\left(\boldsymbol{z}, T^{-1}\left(\boldsymbol{u} \cdot q\right)\right) + \lambda . r_\gamma\left(\boldsymbol{u}\right) \tag{5.16}$$

where r(u) is the number of bits needed for encoding u using MSCF-CABAC given that its two neighboring blocks have been optimized.

In general, SDQ is a search in a vector space of quantization outputs for trade-off between rate and distortion. The efficiency of the search largely depends on how we may discover and utilize the structure of the vector space, which features the de-quantization syntax and the entropy coding method of MSCF-CABAC. In this study, we propose to use a dynamic programming technique to do the search, which requires an additive evaluation of the RD cost. In the following, we first show the additive distortion computation in the DCT domain based on the de-quantization syntax in H.264. Second, we design a Simplified graph for additive evaluation of the rate based on analysis of MSCF-CABAC, with states being defined according to pair coding syntax and connections being specified according to context transitions rules. Finally, we discuss the optimality and complexity of the graph-based algorithm, showing that the graph design helps to solve the minimization problem of Equation 5.16.

### 5.4.1 Distortion Computation in the DCT domain

The distortion term in Equation 5.16 is defined in the pixel domain. It contains inverse DCT, which is not only time consuming, but also makes the optimization problem intractable. Consider that DCT is a unitary transform, which maintains the Euclidean distance. We choose the Euclidean distance for $d(\cdot)$. Then, the distortion can be computed in the transform domain in an element-wise additive manner.

As reviewed in Section 3.2.1, the transform and quantization in H.264 are combined together. Specifically, the residual reconstruction process is

$$T^{-1}\left(\boldsymbol{u}\cdot q\right)=\boldsymbol{w}^{T}\cdot\left(u\otimes\boldsymbol{dq}\left[p_{rem}\right]\cdot 2^{P_{quo}}/64\right)\cdot\boldsymbol{w} \tag{5.17}$$

Since $\hat{w}$ defines a unitary transform, we have

$$\|\hat{\boldsymbol{w}}^{T}\cdot\boldsymbol{Y}\cdot\hat{\boldsymbol{w}}\|^{2}=\|\boldsymbol{Y}\|^{2} \tag{5.18}$$

Equivalently, that is,

$$\|\hat{\boldsymbol{w}}^{T}\cdot\boldsymbol{Y}\cdot\hat{\boldsymbol{w}}\|^{2}=\|\boldsymbol{Y}\otimes\boldsymbol{B}\|^{2} \tag{5.19}$$

where Y is any $4\times 4$ matrix, and

$$B = \begin{bmatrix} 4 & \sqrt{10} & 4 & \sqrt{10} \\ \sqrt{10} & \frac{5}{2} & \sqrt{10} & \frac{5}{2} \\ 4 & \sqrt{10} & 4 & \sqrt{10} \\ \sqrt{10} & \frac{5}{2} & \sqrt{10} & \frac{5}{2} \end{bmatrix}$$

Note that $\boldsymbol{B}$ is obtained based on the given matrices of w and $\hat{w}$ as shown in Section 3.2.1. Consider $z = \hat{w}^T \left(\hat{w} \cdot z \cdot \hat{w}^T\right) \hat{w}$. Applying Equation 5.19, we compute the the distortion term in Equation 5.16 with the Euclidean measure by

$$\begin{aligned} D &= \|z - w^T \cdot \left(u \otimes dq\left[p_{rem}\right] \cdot 2^{P_{quo}}/64\right) \cdot w\|^2 \\ &= \|w^T \left(\left(\hat{w} \cdot z \cdot \hat{w}^T\right) \otimes f - u \otimes dq\left[p_{rem}\right] \cdot 2^{P_{quo}}/64\right) \cdot w\|^2 \\ &= \|c - u \otimes dq\left[p_{rem}\right] \cdot 2^{P_{quo}}/64 \otimes B\|^2 \end{aligned} \tag{5.20}$$

where $c = \left(w \cdot z \cdot w^T\right) \otimes f$. The equation of 5.20 brings to us two advantages. The first is the high efficiency for computing distortion. Note that B and dq are constant matrices defined in the standard. c is computed before soft decision quantization for given z. Thus, the evaluation of D consumes only two integer multiplications together with some shifts and additions per coefficient. More importantly, the second advantage is the resulted element-wise additive computation of distortion, which enables us to solve the soft decision quantization problem using the Viterbi algorithm to be presented later.

After applying the result of Equation 5.20 to 5.16, the soft decision quantization problem becomes

$$u = arg\, min_u \|c - u \otimes dq\left[p_{rem}\right] \cdot 2^{P_{quo}}/64 \otimes B\|^2 + \lambda \cdot r\left(u\right) \tag{5.21}$$

Note that every bold symbol here, e.g., **u**, represents a 4x4 matrix. For entropy coding, the 4x4 matrix of u will be zig-zag ordered into a 1x16 sequence. To facilitate our following discussion of algorithm design based on CAVLC, we introduce a new denotation, i.e., to add a bar on the top of a bold symbol to indicate the zig-zag ordered sequence of the corresponding matrix. E.g., $\overline{u}$ represents the 1x16 vector obtained by ordering u. Then, the equation of 5.21 is rewritten as follows,

$$\overline{u} = arg\, min_{\overline{u}} \|\overline{c} - \overline{u} \otimes \overline{dq}\left[p_{rem}\right] \cdot 2^{P_{quo}}/64 \otimes \overline{B}\|^2 + \lambda \cdot r\left(\overline{u}\right) \tag{5.22}$$

where we still use the symbol $\otimes$ to indicate the element-wise multiplication between two vectors. Finally, for simplicity, we denote $\overline{b}(p) = \overline{dq}[p_{rem}] \cdot 2^{p_{quo}}/64 \otimes \overline{B}$ and obtain the following SDQ problem:

$$\overline{u} = arg\,min\overline{u}\|\overline{c} - \overline{u} \otimes \overline{b}(p)\|^2 + \lambda \cdot r_{CAVLC}(\overline{u}) \tag{5.23}$$

Note that the rate function $r(\cdot)$ is further clarified to be related to $CAVLC^2$.

### 5.4.2 Graph Design for SDQ based on MSCF-CABAC

MSCF-CABAC employs an adaptive context updating scheme besides the adaptive context selection scheme for all significant coefficients in the block. The context states (or probabilities) in a context model for coding a given pair/level, are dependent on all previously encoded pairs. Hence the problem to tackle becomes a two dimensional problem. For a given context states the first problem to tackle can be expressed as

$$\overline{u} = arg\,min_{\hat{u}}\|\overline{c} - u \otimes b(p)\|^2 + \lambda \cdot r(\overline{u} \mid \Omega) \tag{5.24}$$

where represents context states, or the probabilities in all context models used for coding non-zero transform coefficient levels $\overline{u}$.

The second problem is to update context states based on the obtained quantization outputs $\overline{u}$ which is relatively simple problem compared to the first problem.

To solve the problem of 5.23, we develop a graph structure, in which the rate function $r(\overline{u}\,|)$ with given is computed additively. As shown in Figure 5.3, a graph is constructed based on coding features of MSCF-CABAC. Basically, states are defined based on the context model selection, which depends on characteristics of significant coefficients in the block/pairs which is the key difference compared to the H.264 CABAC where the transition depend only on the absolute value of the coefficient if it is greater than 1 or not.

For a $4 \times 4$ luma block, there could be up to a maximum 8 or less columns with each of them corresponding to two adjacent significant coefficients in the encode scanning order. In each column there are up to 4 states. Every state corresponds to a pair of coefficients. Transitions are established between states according to the characteristics of the coefficients in that pair. For example from state (S) there could be transition to states (0_0) if the Quantized coefficient indices in the pair (C14, C15) are (1,1) and can transition to state (1_0) if (C14, C15) form a primitive model. It could also transition to (2_0) if (C14, C15)

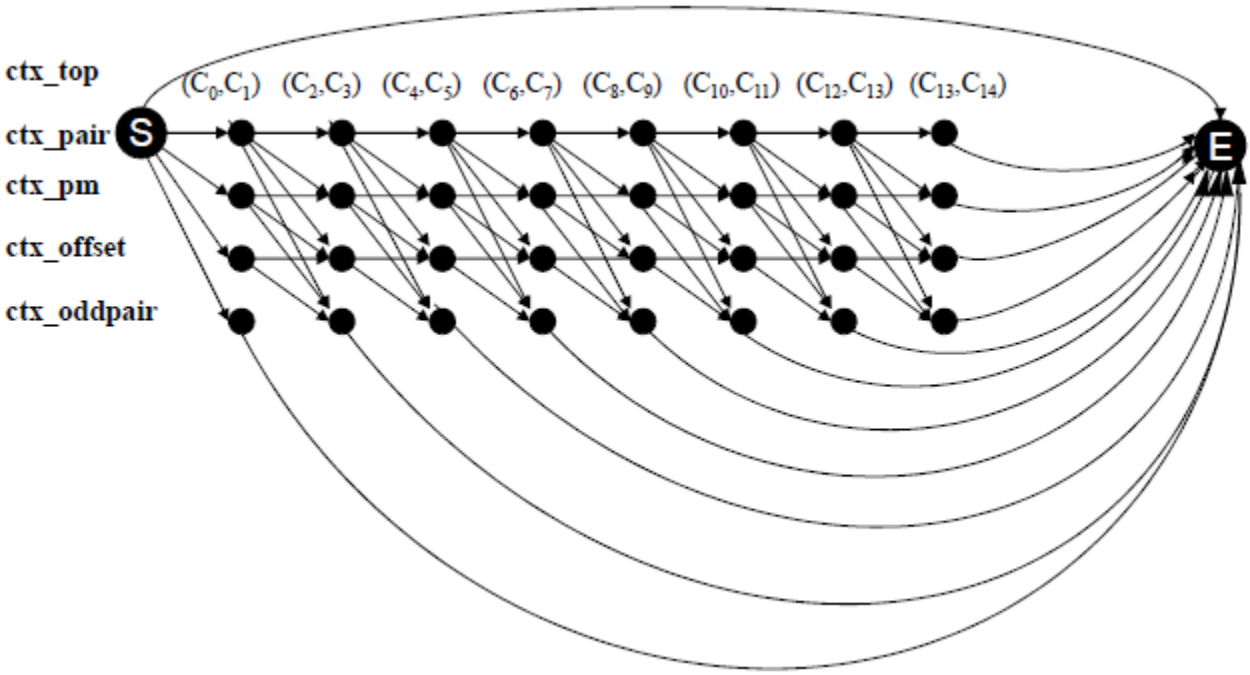


Figure 5.3: Parallel Transitions

does not form a primitive model and can transition to (3_0) if (C14, C15) has odd last significant coefficient.

As we can seen that each state may accord to multiple quantization outputs. These result in multiple transitions between two connected states. There can be as many parallel transitions as the range of coefficients with each according to its unique quantization output. Clearly, the only difference between these parallel transitions and the transitions based on context rules is that former is only specific to the quantization output and will not affect any other connections. Therefore, they are named parallel transitions.

In MSCF-CABAC, a state is defined for a pair of coefficient. So a parallel transition also defined for a pair of coefficients. In practice, because the distortion is a quadratic function with respect to the quantization output, it is sufficient to investigate only a few parallel transitions. Thus the complexity is greatly reduced without sacrificing much of the RD performance.

### 5.4.3 Rate Distortion Metric Computation in the Graph

Let us denote $r_s(\cdot\,|)$, $r_l(\cdot\,|)$ and $r_c(\cdot\,|)$ as the coding rate for a significant-coefficient-flag bit, a last-coefficient-flag bit, and a quantized coefficient $u_i$, respectively.

The RD metric for a typical transition can be defined as

$$g_{m,i} = (c_i - b_i \cdot u_i\,|)^2 + \lambda \cdot (r_s(l \mid \Omega) + r_l(l \mid \Omega) + r_c(u_i \mid \Omega)) \tag{5.25}$$

Below we details the RD metric for multiple parallel transitions. Also since the encoding starts only from the First non-zero coefficient transition from S must go to last significant coefficient which is uK.

Transition 1 :

Consider a transition from the state S to the state E. This transition implies that all the coefficient in the block are 1 and the significant and last coefficient map is same as that from the HDQ.

Transition 2 :

Consider a transition from the state S to the $m^{th}$ state (0¡=m¡=2) at the coefficient uk and denote it as sk,m.This transaction implies that not all the coefficients in the block are one and the coefficients ate pair-wise coded.

Transition 3 :

Consider a transition from the state S to the $m^{th}$ state (m=3) at the coefficient uk. This transition implies that the this coefficient is the last significant coefficient in the encode scanning order. Note this may not be same as the last significant coefficient.

### 5.4.4 Optimality

Based on the graph design and the metric computation discussed above, the solution to 5.23 now becomes a problem of searching for a path in the graph for the minimal RD cost. It is not hard to see that the proposed graph design would allow an element-wise additive computation of the RD cost in 5.23 with given . In this case, the Viterbi algorithm can be used to do the search. By examining the details of MSCF-CABAC, it is not hard to see that for any given path and its corresponding coefficient sequence, the accumulated metric along the path can be easily computed by 5.24. Thus, applying Viterbi algorithm to search the graph leads to the solution of the problem in 5.23.

In general, the optimality of the above SDQ algorithm for 5.3 is not guaranteed due to following reasons

1. Limited search space in a vector space of quantization outputs.
2. Multi-dimensional dependencies involved in the search for optimal path.

Nevertheless, it can be shown that the proposed graph design leads to the optimal solution to 5.23. Thus the SDQ algorithm is referred to as being near-optimal for solving 5.3.

### 5.4.5 Complexity

The complexity of the proposed graph-based SDQ algorithm (i.e., dynamic programming applied to Graph 5.3) depends on three factors, i.e.,

1. Number of columns
2. Number of states in each column
3. Number of parallel transitions for each connection.

## 5.5 Comparison

### 5.5.1 Objective Analysis

The proposed MSCF-CABAC compression scheme has been implemented in H.264 JM reference software. In this section we provide a review of analysis between H.264 with MSCF-CABAC with SDQ with H.264 JM CABAC with RDOQ. Comparative studies of the coding performance are shown by RD curves, with the distortion being measured by PSNR. Figures 5.4 shows the RD curves for Basket-Ball drive sequence measured over 50 frames covering all picture types. Comparisons are conducted among four encoders, i.e., a Main profile encoder with the proposed MSCF-CABAC, a main profile reference encoder with CABAC and RDOQ OFF, a Main profile encoder with the proposed SDQ for MSCF-CABAC and a Main profile encoder with the CABAC and RDOQ enabled. Also Fixed lambda method discussed in section 5.3.1 was used for both proposed SDQ and RDOQ method in reference software implementation.

Experiments over a set of five video sequences (i.e., BasketballDrive, BlowingBubbles, BQMall, BQSquare, ChinaSpeed, ParkScene, Cactus, show the proposed SDQ based on MSCF-CABAC achieves an average 10% rate reduction while preserving the same PSNR CABAC and 3% over the RD optimization in [75] with the Main profile profile.

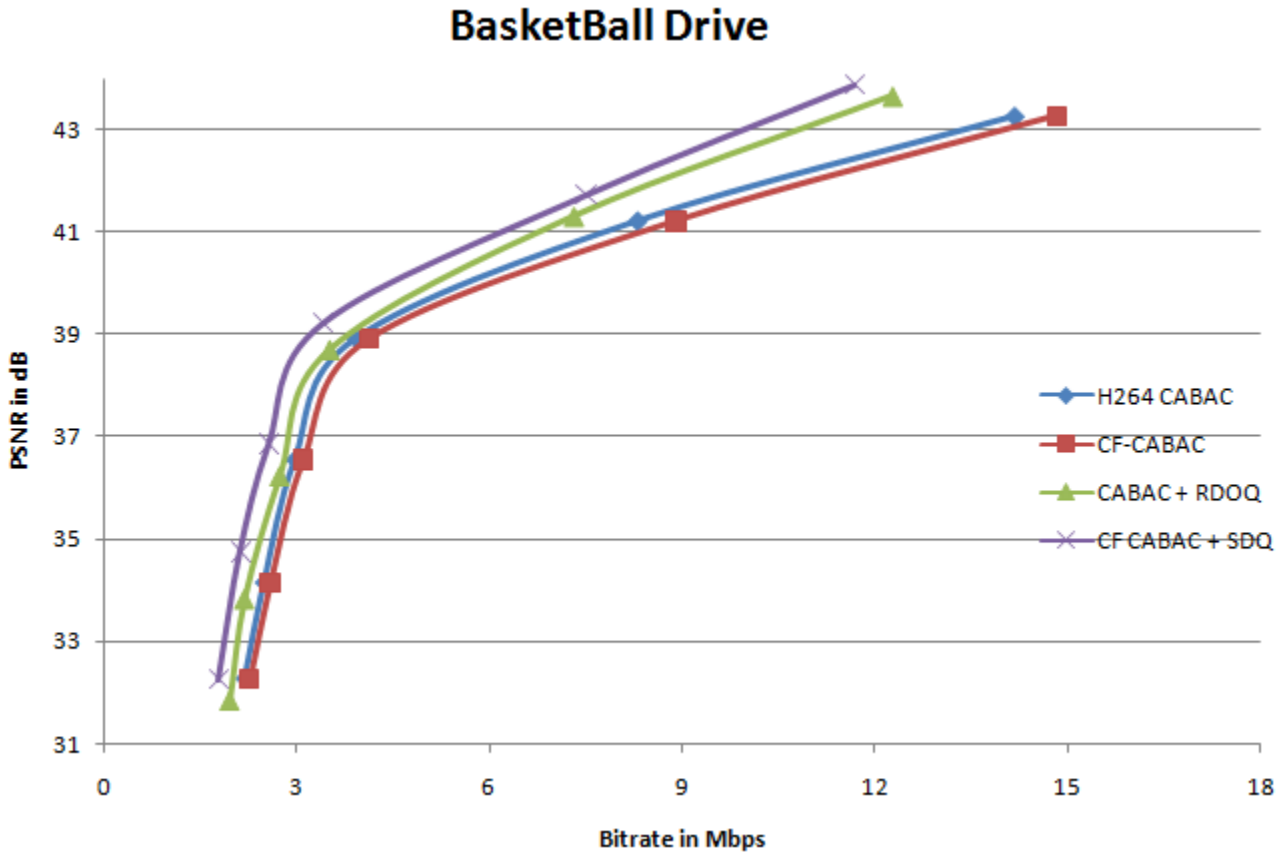


Figure 5.4: Basket Ball Drive Graph

### 5.5.2 Subjective Analysis

The proposed SDQ for MSCF-CABAC is also subjectively better than over the RD optimization in [75] as it preserves more details. One of the key drawbacks is that the PSNR or R-D metric is not optimal from visual perspective. Typically SDQ algorithm tries to trade-off some distortion for gaining rate, especially, tries to cancel the Significant coefficients

with absolute value of 1. This is not desirable from the perceptual quality perspective though. Since SDQ based on MSCF-CABAC efficiently represents the significant coefficients with less scope of cancellation, it is observed to preserve more details in textured areas as shown in figure below.

# Chapter 6

# Conclusion and Future Work

This chapter concludes the thesis with a summary of contributions and presents a few thoughts on future research.

## 6.1 Conclusions

In this thesis, we first briefly studied the existing Image/video compression schemes with main focus on Surface/curve fitting. We discussed the techniques traditionally employed by Hybrid video coding standards. Then we proposed a curve fitting based Context adaptive Binary Arithmetic coding to efficiently optimize the significant coefficients in the block. In the process we defined a set of new syntax element with their motivation.

The Proposed Multi-symbol Curve-fit CABAC was incorporated into the encoder and decoder of the JM 18.3 software. It was applied to the coding of significant coefficients. Experiments were performed using common conditions specified in [77] for nine different set of sequences. Table 4.15 lists the comparison results of proposed MSCF-CABAC vS H264 CABAC for three different operating points (High Rate, Medium Rate and Low Rate). The QP values used for High rate range from 22-26, for Medium rate from 28-32 and for low Rate from 34-38.

- **Compression efficiency**

  Average Percentage Bitrate gain for same PSNR of proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 0.36%, 1.07% and

0.83% respectively. Peak Percentage Bitrate gain for these sequences in same order are 0.62%, 3.87%, 3.02%.

- **Throughput improvement/Bin Reduction** Average Percentage Total Picture Bins Reduction at the above mentioned operating points for proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 0.55%, 1.33% and 1.05% respectively. Average Percentage Total Picture Bins Reduction for these sequences in same order are 0.83%, 4.11%, 2.65%.

Total coefficients Bin Reduction (significant Map, Last Coefficient Map and Residual Level and sign coding).

Average Percentage Total coefficients Bin Reduction at the above mentioned operating points for proposed MSCF-CABAC compared to H264 CABAC for High rate, Mid Rate and Low Rate are 1.17%, 2.63% and 3.59% respectively. Peak Percentage Bitrate gain for these sequences in same order are 1.67%, 4.94%, 5.67%.

Further, In Chapter 6 we proposed a Simplified graph-based SDQ algorithm based on MSCF-CABAC entropy coding for optimizing the significant coefficients in the block. With a similar comparative setting, experiments in Chapter 6 have showed that the proposed graph-based SDQ algorithm based on MSCF-CABAC achieves on average of 10% rate reduction over the JM reference software for main profile H.264 codec with CABAC and RDOQ off., and an average of 2% gain consistently when RDOQ is enabled. Also we demonstrated the complexity of proposed SDQ is dependent on the characteristic of the coefficient in the block and based on statistics it is much less or comparable than the complexity of the RDOQ in JM H.264 Standard.

## 6.2 Future Research

Needless to say, there are many topics left for future work. Yet, in the following, we discuss a few of them.

### 6.2.1 Higher Order Curve fitting

In this thesis, from the idea of curve fitting, we proposed an technique to exploit the first order dependency of the quantized transform coefficients. However the coefficient's do exhibit higher order dependencies in a block. By exploiting this higher order dependencies

and finding efficient representations of an image using polynomial fitting can result in a superior value of PSNR and subjectively quality. Though it comes with own challenges like finding the best high-order fitting scheme for different contents and computationally requirement. Also more complex fitting schemes require more data to convey to the decode. If the above challenges can be addressed, it is only prudent to apply high-order curve fitting techniques to achieve better compression.

### 6.2.2 JOINT R-D optimizations Framework

Video compression generally assumes four types (temporal, spatial, psycho-visual, and statistical) of redundancy, leading to a hybrid coding structure [30], as shown in Figure 1.2. The hybrid structure employs four coding parts, i.e., motion compensation, transform, quantization, and entropy coding to reduce each of those redundancies. H.264, the newest hybrid video compression standard [75], has proved its superiority in coding efficiency over its precedents, largely due to its improved coding technologies for each individual coding part from efficient prediction modes to complex binary arithmetic coding. It can be Intuitively demonstrated that each coding part is not entirely independent and they affect the overall R-D performance of other coding parts. For example, From the correlation point the more effective is motion compensation, the less correlated are the residuals, thus the less interesting for transforming the residual to the frequency domain.

This thesis only focuses on one of the coding part, the Entropy coding design. It would be interesting to see how much RD theoretic studies can help to further improve the coding performance for H.264 by jointly designing the whole hybrid coding structure with the proposed entropy coding.

# APPENDICES

# Appendix A

# AppendixA

Below table Summarizes in great detail the different probabilities of occurrence of coefficient Indices for all range of QPs and all picture types.

BasketBall Drive

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 93.7 | 5.938 | 0.3 | 0.04 |
| 1 | I | 32 | 98.3 | 1.622 | 0.1 | 0.02 |
| 1 | I | 37 | 98.8 | 1.152 | 0.1 | 0.02 |
| 1 | I | 42 | 99.1 | 0.887 | 0 | 0.01 |
| 1 | I | 47 | 99.3 | 0.634 | 0.1 | 0.01 |
| 1 | P | 22 | 94 | 5.457 | 0.4 | 0.08 |
| 1 | P | 32 | 97.5 | 2.359 | 0.1 | 0.03 |
| 1 | P | 37 | 98.1 | 1.751 | 0.1 | 0.02 |
| 1 | P | 42 | 98.7 | 1.247 | 0.1 | 0.01 |
| 1 | P | 47 | 99.2 | 0.791 | 0 | 0.01 |
| 1 | B | 22 | 93.3 | 5.979 | 0.6 | 0.12 |
| 1 | B | 32 | 97.9 | 1.958 | 0.1 | 0.01 |
| 1 | B | 37 | 98.5 | 1.372 | 0.1 | 0.01 |
| 1 | B | 42 | 99 | 0.958 | 0.1 | 0 |
| 1 | B | 47 | 99.6 | 0.358 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 85 | 13.1 | 0.97 | 0.33 | 0.03 | 0.42 |
| 2 | I | 32 | 88 | 10.3 | 1.51 | 0.2 | 0.02 | 0.31 |
| 2 | I | 37 | 90 | 8.36 | 0.89 | 0.17 | 0.07 | 0.13 |
| 2 | I | 42 | 93 | 5.3 | 1.11 | 0.14 | 0 | 0.07 |
| 2 | I | 47 | 92 | 7.25 | 0.78 | 0 | 0 | 0 |
| 2 | P | 22 | 85 | 12.9 | 1.34 | 0.54 | 0.06 | 0.6 |
| 2 | P | 32 | 88 | 10 | 1.16 | 0.36 | 0.03 | 0.27 |
| 2 | P | 37 | 88 | 10 | 1.37 | 0.3 | 0.02 | 0.16 |
| 2 | P | 42 | 92 | 7.3 | 0.73 | 0.13 | 0 | 0.21 |
| 2 | P | 47 | 95 | 4.74 | 0.35 | 0.18 | 0 | 0 |
| 2 | B | 22 | 86 | 11.6 | 1.47 | 0.52 | 0.06 | 0.58 |
| 2 | B | 32 | 89 | 9.33 | 1.15 | 0.2 | 0.02 | 0.15 |
| 2 | B | 37 | 92 | 6.68 | 0.85 | 0.08 | 0 | 0.03 |
| 2 | B | 42 | 94 | 5.69 | 0.23 | 0.11 | 0 | 0 |
| 2 | B | 47 | 96 | 3.57 | 0 | 0 | 0 | 0 |

| Block coeffs > 0 | Picture Type | QP | 1% | 1-2% | 2-3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 94.02 | 4.2 | 1.79 |
| 3 | I | 32 | 0 | 94.88 | 3.8 | 1.33 |
| 3 | I | 37 | 0 | 96.59 | 2.5 | 0.91 |
| 3 | I | 42 | 0 | 95.4 | 3 | 1.61 |
| 3 | I | 47 | 0 | 95.59 | 3.5 | 0.88 |
| 3 | P | 22 | 0 | 94.23 | 4 | 1.75 |
| 3 | P | 32 | 0 | 93.22 | 5.1 | 1.69 |
| 3 | P | 37 | 0 | 93.44 | 5 | 1.58 |
| 3 | P | 42 | 0 | 95.35 | 3.6 | 1.03 |
| 3 | P | 47 | 0 | 95.35 | 4.7 | 0 |
| 3 | B | 22 | 0 | 95.74 | 2.9 | 1.32 |
| 3 | B | 32 | 0 | 96.12 | 3.2 | 0.66 |
| 3 | B | 37 | 0 | 97.15 | 2.4 | 0.47 |
| 3 | B | 42 | 0 | 96 | 2.9 | 1.14 |
| 3 | B | 47 | 0 | 94.74 | 5.3 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2-3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 45.9 | 37.59 | 11 | 5.48 |
| 4 | I | 32 | 45 | 40 | 11 | 4.16 |
| 4 | I | 37 | 50.9 | 37.91 | 7.1 | 4.03 |
| 4 | I | 42 | 64.7 | 29.41 | 3.5 | 2.35 |
| 4 | I | 47 | 56.7 | 40 | 3.3 | 0 |
| 4 | P | 22 | 48.5 | 35.05 | 10 | 5.96 |
| 4 | P | 32 | 43.1 | 39.57 | 12 | 5 |
| 4 | P | 37 | 43.5 | 42.51 | 11 | 3.12 |
| 4 | P | 42 | 61.2 | 30.22 | 7.2 | 1.44 |
| 4 | P | 47 | 75 | 16.67 | 8.3 | 0 |
| 4 | B | 22 | 55.3 | 31.54 | 8.7 | 4.39 |
| 4 | B | 32 | 50.5 | 39.28 | 7.6 | 2.58 |
| 4 | B | 37 | 53.6 | 37.5 | 6 | 2.98 |
| 4 | B | 42 | 69.2 | 19.23 | 12 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2-3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 31.5 | 37.09 | 18 | 13.2 |
| 5 | I | 32 | 20.5 | 47.62 | 21 | 10.6 |
| 5 | I | 37 | 27.6 | 49.66 | 14 | 8.97 |
| 5 | I | 42 | 48.1 | 25.93 | 26 | 0 |
| 5 | I | 47 | 57.1 | 28.57 | 0 | 14.3 |
| 5 | P | 22 | 37.6 | 36.4 | 15 | 10.7 |
| 5 | P | 32 | 23.7 | 42.42 | 22 | 11.8 |
| 5 | P | 37 | 27.2 | 52.63 | 14 | 6.41 |
| 5 | P | 42 | 50 | 35.29 | 8.8 | 5.88 |
| 5 | B | 22 | 42.3 | 35.36 | 13 | 8.93 |
| 5 | B | 32 | 33.1 | 45.09 | 18 | 4.28 |
| 5 | B | 37 | 42.9 | 41.76 | 9.9 | 5.49 |
| 5 | B | 42 | 50 | 50 | 0 | 0 |

Blowing Bubbles

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 90.4 | 8.59 | 0.8 | 0.2 |
| 1 | I | 27 | 92.9 | 6.75 | 0.3 | 0.1 |
| 1 | I | 32 | 95.9 | 3.75 | 0.34 | 0 |
| 1 | I | 37 | 97.1 | 2.56 | 0.28 | 0.1 |
| 1 | I | 42 | 97.6 | 2.11 | 0.29 | 0 |
| 1 | I | 47 | 97.9 | 1.93 | 0.11 | 0.1 |
| 1 | P | 22 | 88 | 9.37 | 1.86 | 0.8 |
| 1 | P | 27 | 91.7 | 7.19 | 0.86 | 0.2 |
| 1 | P | 32 | 96 | 3.66 | 0.27 | 0.1 |
| 1 | P | 37 | 97.4 | 2.25 | 0.23 | 0.1 |
| 1 | P | 42 | 98.3 | 1.5 | 0.12 | 0.1 |
| 1 | P | 47 | 98.4 | 1.6 | 0 | 0 |
| 1 | B | 22 | 91.2 | 7.3 | 1.16 | 0.3 |
| 1 | B | 27 | 93.5 | 5.8 | 0.56 | 0.1 |
| 1 | B | 32 | 96.4 | 3.33 | 0.18 | 0 |
| 1 | B | 37 | 98.1 | 1.75 | 0.19 | 0 |
| 1 | B | 42 | 97.1 | 2.73 | 0.18 | 0 |
| 1 | B | 47 | 97.8 | 2.2 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 81.5 | 15 | 1.52 | 0.69 | 0 | 0.83 |
| 2 | I | 32 | 86.2 | 11 | 1.35 | 0.42 | 0.17 | 0.42 |
| 2 | I | 27 | 84.4 | 13 | 0.85 | 0.56 | 0.14 | 0.77 |
| 2 | I | 37 | 85.4 | 13 | 0.8 | 0.13 | 0 | 0.27 |
| 2 | I | 42 | 93 | 6.1 | 0.84 | 0 | 0 | 0 |
| 2 | I | 47 | 95.9 | 3.2 | 0.91 | 0 | 0 | 0 |
| 2 | P | 22 | 80.7 | 15 | 2.01 | 1.06 | 0.11 | 1.6 |
| 2 | P | 27 | 84.8 | 12 | 1.71 | 0.81 | 0.07 | 0.9 |
| 2 | P | 32 | 88 | 10 | 1.45 | 0.07 | 0.04 | 0.32 |
| 2 | P | 37 | 92.7 | 5.3 | 1.3 | 0.08 | 0.08 | 0.46 |
| 2 | P | 42 | 92.8 | 6.9 | 0.28 | 0 | 0 | 0 |
| 2 | P | 47 | 93.9 | 6.1 | 0 | 0 | 0 | 0 |
| 2 | B | 22 | 81.9 | 14 | 1.63 | 0.78 | 0.07 | 1.75 |
| 2 | B | 27 | 84.9 | 12 | 1.59 | 0.73 | 0.06 | 0.67 |
| 2 | B | 32 | 87.9 | 10 | 1.12 | 0.19 | 0 | 0.42 |
| 2 | B | 37 | 86.5 | 13 | 0.46 | 0 | 0 | 0.46 |
| 2 | B | 42 | 89.7 | 8.4 | 1.87 | 0 | 0 | 0 |
| 2 | B | 47 | 88.9 | 11 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 93.6 | 4.02 | 2.3 |
| 3 | I | 27 | 0 | 94.3 | 3.84 | 1.8 |
| 3 | I | 32 | 0 | 96.3 | 2.68 | 1 |
| 3 | I | 37 | 0 | 97.6 | 1.69 | 0.7 |
| 3 | I | 42 | 0 | 99.3 | 0.69 | 0 |
| 3 | I | 47 | 0 | 100 | 0 | 0 |
| 3 | P | 22 | 0 | 93 | 4.31 | 2.7 |
| 3 | P | 27 | 0 | 94.5 | 3.6 | 1.9 |
| 3 | P | 32 | 0 | 96.3 | 2.48 | 1.2 |
| 3 | P | 37 | 0 | 94.9 | 3.1 | 2 |
| 3 | P | 42 | 0 | 93.7 | 2.53 | 3.8 |
| 3 | P | 47 | 0 | 100 | 0 | 0 |
| 3 | B | 22 | 0 | 94.4 | 3.41 | 2.2 |
| 3 | B | 27 | 0 | 94.9 | 3.29 | 1.8 |
| 3 | B | 32 | 0 | 94.9 | 2.85 | 2.2 |
| 3 | B | 37 | 0 | 92.8 | 4.31 | 2.9 |
| 3 | B | 42 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 50.1 | 35.7 | 9.82 | 4.4 |
| 4 | I | 27 | 48.9 | 37.8 | 9.09 | 4.3 |
| 4 | I | 32 | 56.7 | 34.7 | 6.75 | 1.9 |
| 4 | I | 37 | 59.8 | 34.7 | 4.63 | 0.8 |
| 4 | I | 42 | 82.4 | 16.2 | 1.35 | 0 |
| 4 | I | 47 | 84.2 | 15.8 | 0 | 0 |
| 4 | P | 22 | 55 | 31.1 | 9.05 | 4.9 |
| 4 | P | 27 | 58.2 | 30.2 | 7.55 | 4 |
| 4 | P | 32 | 61 | 32 | 4.77 | 2.3 |
| 4 | P | 37 | 64.9 | 28.9 | 5.06 | 1.2 |
| 4 | P | 42 | 57.1 | 28.6 | 14.3 | 0 |
| 4 | B | 47 | 65.7 | 23.6 | 6.63 | 4.1 |
| 4 | B | 22 | 59.3 | 29.2 | 7.64 | 3.8 |
| 4 | B | 27 | 63.9 | 27 | 6.52 | 2.6 |
| 4 | B | 32 | 62.1 | 29.3 | 8.62 | 0 |
| 4 | B | 37 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 34.2 | 40.2 | 14.6 | 11 |
| 5 | I | 27 | 39.5 | 40.6 | 14 | 5.9 |
| 5 | I | 32 | 53.6 | 36.3 | 7.77 | 2.3 |
| 5 | I | 37 | 48.6 | 44.6 | 4.05 | 2.7 |
| 5 | I | 42 | 79.4 | 20.6 | 0 | 0 |
| 5 | I | 47 | 90 | 10 | 0 | 0 |
| 5 | P | 22 | 50.1 | 30.5 | 10.7 | 8.7 |
| 5 | P | 27 | 49.2 | 33.9 | 10.9 | 6 |
| 5 | P | 32 | 51.1 | 36.4 | 8.79 | 3.7 |
| 5 | P | 37 | 62.8 | 28.8 | 7.05 | 1.3 |
| 5 | P | 42 | 37.5 | 62.5 | 0 | 0 |
| 5 | B | 47 | 59.2 | 24.6 | 9.28 | 6.9 |
| 5 | B | 22 | 46.8 | 37.5 | 10.7 | 5 |
| 5 | B | 27 | 54 | 36.8 | 7.28 | 2 |
| 5 | B | 32 | 76.9 | 15.4 | 7.69 | 0 |

BQMall

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 91.9 | 7.35 | 0.57 | 0.15 |
| 1 | I | 27 | 94.5 | 4.98 | 0.29 | 0.23 |
| 1 | I | 32 | 96.5 | 2.98 | 0.36 | 0.12 |
| 1 | I | 37 | 97.1 | 2.54 | 0.2 | 0.15 |
| 1 | I | 42 | 97.3 | 2.33 | 0.34 | 0.02 |
| 1 | I | 47 | 97.7 | 2.03 | 0.21 | 0.08 |
| 1 | P | 22 | 91.3 | 6.88 | 1.25 | 0.61 |
| 1 | P | 27 | 93.1 | 5.65 | 0.85 | 0.45 |
| 1 | P | 32 | 95.7 | 3.72 | 0.43 | 0.13 |
| 1 | P | 37 | 96.6 | 2.98 | 0.29 | 0.09 |
| 1 | P | 42 | 98.2 | 1.61 | 0.13 | 0.05 |
| 1 | P | 47 | 98.3 | 1.69 | 0.04 | 0 |
| 1 | B | 22 | 93.2 | 5.59 | 0.84 | 0.33 |
| 1 | B | 27 | 94.8 | 4.3 | 0.68 | 0.18 |
| 1 | B | 32 | 97.1 | 2.57 | 0.26 | 0.06 |
| 1 | B | 37 | 97.9 | 1.88 | 0.18 | 0 |
| 1 | B | 42 | 98.9 | 1.1 | 0 | 0 |
| 1 | B | 47 | 98.8 | 1.18 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 80.5 | 16.2 | 1.43 | 0.79 | 0.07 | 0.98 |
| 2 | I | 32 | 88.3 | 9.41 | 1.7 | 0.39 | 0 | 0.24 |
| 2 | I | 27 | 84 | 13 | 1.95 | 0.43 | 0.13 | 0.48 |
| 2 | I | 37 | 87.9 | 10.3 | 1.23 | 0.25 | 0.11 | 0.18 |
| 2 | I | 42 | 92 | 6.74 | 0.99 | 0.11 | 0.06 | 0.11 |
| 2 | I | 47 | 93.8 | 5.66 | 0.42 | 0.14 | 0 | 0 |
| 2 | P | 22 | 81 | 14 | 2.64 | 0.96 | 0.19 | 1.26 |
| 2 | P | 27 | 83.9 | 13.1 | 1.63 | 0.46 | 0.11 | 0.82 |
| 2 | P | 32 | 87.9 | 10.1 | 1.11 | 0.37 | 0.04 | 0.51 |
| 2 | P | 37 | 90.6 | 7.92 | 1.16 | 0.17 | 0 | 0.12 |
| 2 | P | 42 | 93.5 | 6.12 | 0.34 | 0.06 | 0 | 0 |
| 2 | P | 47 | 94.2 | 5.81 | 0 | 0 | 0 | 0 |
| 2 | B | 22 | 85.2 | 10.4 | 2.28 | 0.91 | 0.13 | 1.01 |
| 2 | B | 27 | 88.4 | 8.52 | 1.87 | 0.64 | 0.09 | 0.49 |
| 2 | B | 32 | 90.6 | 7.6 | 1.2 | 0.34 | 0.02 | 0.19 |
| 2 | B | 37 | 92.6 | 6.51 | 0.62 | 0.28 | 0 | 0 |
| 2 | B | 42 | 95.1 | 4.18 | 0.7 | 0 | 0 | 0 |
| 2 | B | 47 | 100 | 0 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 93.8 | 3.93 | 2.28 |
| 3 | I | 27 | 0 | 94 | 3.84 | 2.11 |
| 3 | I | 32 | 0 | 96.1 | 2.5 | 1.36 |
| 3 | I | 37 | 0 | 96.3 | 2.38 | 1.32 |
| 3 | I | 42 | 0 | 96.3 | 2.91 | 0.75 |
| 3 | I | 47 | 0 | 97.7 | 1.85 | 0.5 |
| 3 | P | 22 | 0 | 95.1 | 2.89 | 2.01 |
| 3 | P | 27 | 0 | 95.1 | 2.9 | 2.01 |
| 3 | P | 32 | 0 | 96.7 | 2.47 | 0.83 |
| 3 | P | 37 | 0 | 96.6 | 2.66 | 0.77 |
| 3 | P | 42 | 0 | 97.6 | 2.09 | 0.35 |
| 3 | P | 47 | 0 | 96.8 | 3.23 | 0 |
| 3 | B | 22 | 0 | 95.5 | 2.87 | 1.65 |
| 3 | B | 27 | 0 | 96.2 | 2.55 | 1.22 |
| 3 | B | 32 | 0 | 97.1 | 2.4 | 0.55 |
| 3 | B | 37 | 0 | 97.9 | 1.87 | 0.19 |
| 3 | B | 42 | 0 | 97.4 | 2.56 | 0 |
| 3 | B | 47 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 41.2 | 38 | 12.4 | 8.43 |
| 4 | I | 27 | 42 | 39.3 | 12.7 | 5.89 |
| 4 | I | 32 | 50.8 | 36 | 10.3 | 2.8 |
| 4 | I | 37 | 57.8 | 32.1 | 7.18 | 2.93 |
| 4 | I | 42 | 63.5 | 30.1 | 4.34 | 2.04 |
| 4 | I | 47 | 64.9 | 29.4 | 4.12 | 1.55 |
| 4 | P | 22 | 47.7 | 32.8 | 12.4 | 7.1 |
| 4 | P | 27 | 48.7 | 35.8 | 10.1 | 5.29 |
| 4 | P | 32 | 53 | 35.8 | 8.03 | 3.17 |
| 4 | P | 37 | 56.5 | 34.2 | 7.69 | 1.62 |
| 4 | P | 42 | 70.6 | 24.6 | 4.28 | 0.53 |
| 4 | P | 47 | 66.7 | 33.3 | 0 | 0 |
| 4 | B | 22 | 59.5 | 27.4 | 8.1 | 5.01 |
| 4 | B | 27 | 57.5 | 31 | 8.16 | 3.27 |
| 4 | B | 32 | 59.7 | 32.1 | 6.55 | 1.65 |
| 4 | B | 37 | 62.5 | 34.9 | 2.63 | 0 |
| 4 | B | 42 | 75 | 25 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 27.7 | 39.8 | 18.6 | 13.8 |
| 5 | I | 27 | 31 | 38.8 | 17.6 | 12.6 |
| 5 | I | 32 | 36.5 | 43.6 | 12.9 | 7 |
| 5 | I | 37 | 46 | 38.9 | 10.8 | 4.32 |
| 5 | I | 42 | 53.4 | 35.2 | 10.2 | 1.14 |
| 5 | I | 47 | 57.1 | 36.5 | 6.35 | 0 |
| 5 | P | 22 | 36.4 | 35.2 | 16.2 | 12.2 |
| 5 | P | 27 | 34.4 | 39.8 | 15.7 | 10 |
| 5 | P | 32 | 39.3 | 41.1 | 13.8 | 5.74 |
| 5 | P | 37 | 41.3 | 41.4 | 14.3 | 3.02 |
| 5 | P | 42 | 69.2 | 23.1 | 2.56 | 5.13 |
| 5 | P | 47 | 100 | 0 | 0 | 0 |
| 5 | B | 22 | 49.1 | 31 | 11.3 | 8.57 |
| 5 | B | 27 | 44.3 | 37.1 | 12.4 | 6.14 |
| 5 | B | 32 | 45.6 | 41 | 10.7 | 2.72 |
| 5 | B | 37 | 59.5 | 33.3 | 7.14 | 0 |

BQSquare

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 93.02 | 6.36 | 0.31 | 0.31 |
| 1 | I | 27 | 95.83 | 3.25 | 0.61 | 0.31 |
| 1 | I | 32 | 95.11 | 4.13 | 0.25 | 0.51 |
| 1 | I | 37 | 94.75 | 3.87 | 0.83 | 0.55 |
| 1 | I | 42 | 95.78 | 3.23 | 0.72 | 0.27 |
| 1 | I | 47 | 96.23 | 2.81 | 0.7 | 0.26 |
| 1 | P | 22 | 96.33 | 3.35 | 0.32 | 0 |
| 1 | P | 27 | 94.82 | 4.69 | 0.49 | 0 |
| 1 | P | 32 | 94.03 | 5.26 | 0.71 | 0 |
| 1 | P | 37 | 98.14 | 1.86 | 0 | 0 |
| 1 | P | 42 | 99.78 | 0.22 | 0 | 0 |
| 1 | P | 47 | 97.56 | 2.44 | 0 | 0 |
| 1 | B | 22 | 91.11 | 8.11 | 0.66 | 0.12 |
| 1 | B | 27 | 95.86 | 3.97 | 0.14 | 0.03 |
| 1 | B | 32 | 98.4 | 1.6 | 0 | 0 |
| 1 | B | 37 | 100 | 0 | 0 | 0 |
| 1 | B | 42 | 98.53 | 1.47 | 0 | 0 |
| 1 | B | 47 | 100 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 86.2 | 10.8 | 1.5 | 0.3 | 0.15 | 1.05 |
| 2 | I | 32 | 83.3 | 14.2 | 1.37 | 0.27 | 0 | 0.82 |
| 2 | I | 27 | 87.5 | 8.93 | 2.01 | 0.89 | 0 | 0.67 |
| 2 | I | 37 | 87.6 | 9.8 | 1.09 | 0.65 | 0 | 0.87 |
| 2 | I | 42 | 89.6 | 7.94 | 1.06 | 0.53 | 0.18 | 0.71 |
| 2 | I | 47 | 87.8 | 8.68 | 2.67 | 0.67 | 0 | 0.17 |
| 2 | P | 22 | 86.3 | 11.5 | 1.51 | 0.47 | 0 | 0.19 |
| 2 | P | 27 | 89.4 | 9.09 | 0.93 | 0.4 | 0.07 | 0.13 |
| 2 | P | 32 | 92.2 | 6.67 | 1.03 | 0.11 | 0 | 0 |
| 2 | P | 37 | 98.8 | 0.82 | 0.35 | 0 | 0 | 0 |
| 2 | P | 42 | 92.6 | 6.67 | 0.37 | 0.37 | 0 | 0 |
| 2 | P | 47 | 94.1 | 4.24 | 0.85 | 0.85 | 0 | 0 |
| 2 | B | 22 | 88.5 | 10.4 | 0.78 | 0.21 | 0.02 | 0.04 |
| 2 | B | 27 | 94.9 | 4.72 | 0.32 | 0.07 | 0.02 | 0.02 |
| 2 | B | 32 | 98.1 | 1.58 | 0.18 | 0.18 | 0 | 0 |
| 2 | B | 37 | 93.7 | 6.34 | 0 | 0 | 0 | 0 |
| 2 | B | 42 | 100 | 0 | 0 | 0 | 0 | 0 |
| 2 | B | 47 | 83.3 | 16.7 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 97.1 | 2.11 | 0.81 |
| 3 | I | 27 | 0 | 94.8 | 2.86 | 2.34 |
| 3 | I | 32 | 0 | 97.1 | 1.75 | 1.17 |
| 3 | I | 37 | 0 | 96.6 | 2 | 1.4 |
| 3 | I | 42 | 0 | 96.3 | 2.43 | 1.22 |
| 3 | I | 47 | 0 | 96.8 | 2.08 | 1.12 |
| 3 | P | 22 | 0 | 99.3 | 0.65 | 0 |
| 3 | P | 27 | 0 | 99.4 | 0.53 | 0.05 |
| 3 | P | 32 | 0 | 99.7 | 0.24 | 0.06 |
| 3 | P | 37 | 0 | 99.5 | 0.35 | 0.18 |
| 3 | P | 42 | 0 | 98.8 | 0.61 | 0.61 |
| 3 | P | 47 | 0 | 100 | 0 | 0 |
| 3 | B | 22 | 0 | 99.6 | 0.36 | 0.04 |
| 3 | B | 27 | 0 | 99.7 | 0.29 | 0.03 |
| 3 | B | 32 | 0 | 99.8 | 0.22 | 0 |
| 3 | B | 37 | 0 | 98.7 | 0 | 1.32 |
| 3 | B | 42 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 71.51 | 22 | 3.76 | 2.69 |
| 4 | I | 27 | 63.33 | 26.4 | 5.64 | 4.62 |
| 4 | I | 32 | 50.77 | 35.5 | 7.14 | 6.63 |
| 4 | I | 37 | 60.16 | 29.3 | 6.25 | 4.3 |
| 4 | I | 42 | 58.6 | 32.9 | 5.68 | 2.84 |
| 4 | I | 47 | 57.69 | 35.2 | 5 | 2.12 |
| 4 | P | 22 | 72.83 | 19.5 | 5.59 | 2.09 |
| 4 | P | 27 | 76.3 | 19.2 | 3.42 | 1.11 |
| 4 | P | 32 | 89.14 | 9.01 | 1.37 | 0.48 |
| 4 | P | 37 | 77.18 | 17.7 | 3.3 | 1.8 |
| 4 | P | 42 | 74.19 | 23.7 | 2.15 | 0 |
| 4 | P | 47 | 100 | 0 | 0 | 0 |
| 4 | B | 22 | 77.78 | 18.1 | 3.3 | 0.8 |
| 4 | B | 27 | 81.84 | 15.9 | 2.03 | 0.22 |
| 4 | B | 32 | 76 | 18.9 | 3.43 | 1.71 |
| 4 | B | 37 | 67.74 | 22.6 | 9.68 | 0 |
| 4 | B | 42 | 100 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 62.3 | 24 | 8.81 | 4.92 |
| 5 | I | 27 | 45.45 | 33.6 | 11.2 | 9.7 |
| 5 | I | 32 | 48.15 | 32.6 | 14.3 | 4.94 |
| 5 | I | 37 | 45.61 | 36.8 | 11.4 | 6.14 |
| 5 | I | 42 | 47.29 | 41.6 | 8.24 | 2.82 |
| 5 | I | 47 | 55.42 | 34.9 | 6.43 | 3.21 |
| 5 | P | 22 | 65.17 | 24.4 | 7.27 | 3.19 |
| 5 | P | 27 | 72.9 | 20.2 | 5.08 | 1.78 |
| 5 | P | 32 | 83.71 | 12.8 | 2.86 | 0.62 |
| 5 | P | 37 | 53.62 | 31.4 | 10.1 | 4.83 |
| 5 | P | 42 | 56.67 | 40 | 3.33 | 0 |
| 4 | P | 47 | 100 | 0 | 0 | 0 |
| 5 | B | 22 | 72.21 | 22.5 | 3.96 | 1.3 |
| 5 | B | 27 | 77.16 | 18.8 | 3.25 | 0.84 |
| 5 | B | 32 | 58.26 | 29.6 | 8.7 | 3.48 |
| 5 | B | 37 | 53.85 | 46.2 | 0 | 0 |

BQTerrace

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 95.94 | 3.76 | 0.23 | 0.07 |
| 1 | I | 27 | 96.68 | 3 | 0.26 | 0.06 |
| 1 | I | 32 | 96.94 | 2.77 | 0.22 | 0.08 |
| 1 | I | 37 | 96.83 | 2.76 | 0.3 | 0.1 |
| 1 | I | 42 | 97.17 | 2.52 | 0.24 | 0.07 |
| 1 | I | 47 | 97.52 | 2.24 | 0.17 | 0.07 |
| 1 | P | 22 | 98.38 | 1.52 | 0.08 | 0.02 |
| 1 | P | 27 | 98.03 | 1.78 | 0.17 | 0.02 |
| 1 | P | 32 | 98.42 | 1.47 | 0.1 | 0.01 |
| 1 | P | 37 | 99 | 0.91 | 0.09 | 0 |
| 1 | P | 42 | 99 | 0.96 | 0.03 | 0.01 |
| 1 | P | 47 | 98.96 | 0.95 | 0.09 | 0 |
| 1 | B | 22 | 98.82 | 1.12 | 0.05 | 0.01 |
| 1 | B | 27 | 98.63 | 1.29 | 0.07 | 0.01 |
| 1 | B | 32 | 99.15 | 0.8 | 0.05 | 0 |
| 1 | B | 37 | 99.33 | 0.65 | 0.01 | 0 |
| 1 | B | 42 | 99.35 | 0.65 | 0 | 0 |
| 1 | B | 47 | 99.52 | 0.48 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 91 | 7.54 | 0.78 | 0.24 | 0.05 | 0.39 |
| 2 | I | 32 | 88.1 | 9.65 | 1.26 | 0.51 | 0.08 | 0.39 |
| 2 | I | 27 | 88.1 | 9.65 | 1.24 | 0.42 | 0.05 | 0.59 |
| 2 | I | 37 | 88.8 | 9.09 | 1.32 | 0.36 | 0.07 | 0.4 |
| 2 | I | 42 | 91.2 | 6.81 | 1.49 | 0.35 | 0.02 | 0.12 |
| 2 | I | 47 | 90.9 | 6.6 | 2.16 | 0.23 | 0.03 | 0.07 |
| 2 | P | 22 | 94.9 | 4.33 | 0.53 | 0.11 | 0.02 | 0.08 |
| 2 | P | 27 | 93.8 | 5.26 | 0.72 | 0.16 | 0.01 | 0.1 |
| 2 | P | 32 | 93.5 | 5.27 | 0.88 | 0.22 | 0.01 | 0.08 |
| 2 | P | 37 | 94.4 | 4.43 | 0.98 | 0.1 | 0 | 0.12 |
| 2 | P | 42 | 94.4 | 4.54 | 0.87 | 0.17 | 0 | 0 |
| 2 | P | 47 | 92.8 | 6.4 | 0.8 | 0 | 0 | 0 |
| 2 | B | 22 | 96.8 | 2.86 | 0.28 | 0.07 | 0.01 | 0.01 |
| 2 | B | 27 | 96.2 | 3.17 | 0.53 | 0.08 | 0 | 0.01 |
| 2 | B | 32 | 95.6 | 3.5 | 0.68 | 0.15 | 0.01 | 0.03 |
| 2 | B | 37 | 95.6 | 3.96 | 0.41 | 0 | 0 | 0.06 |
| 2 | B | 42 | 95.1 | 4.43 | 0.49 | 0 | 0 | 0 |
| 2 | B | 47 | 100 | 0 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 97.1 | 1.57 | 1.36 |
| 3 | I | 27 | 0 | 95.4 | 2.99 | 1.62 |
| 3 | I | 32 | 0 | 96.3 | 2.52 | 1.19 |
| 3 | I | 37 | 0 | 96.5 | 2.39 | 1.15 |
| 3 | I | 42 | 0 | 96.8 | 2.46 | 0.69 |
| 3 | I | 47 | 0 | 97.5 | 1.87 | 0.65 |
| 3 | P | 22 | 0 | 99.4 | 0.49 | 0.16 |
| 3 | P | 27 | 0 | 99 | 0.8 | 0.24 |
| 3 | P | 32 | 0 | 98.8 | 0.93 | 0.28 |
| 3 | P | 37 | 0 | 98.1 | 1.65 | 0.23 |
| 3 | P | 42 | 0 | 98.3 | 1.72 | 0 |
| 3 | P | 47 | 0 | 100 | 0 | 0 |
| 3 | B | 22 | 0 | 99.9 | 0.12 | 0.02 |
| 3 | B | 27 | 0 | 99.5 | 0.34 | 0.13 |
| 3 | B | 32 | 0 | 99.4 | 0.48 | 0.16 |
| 3 | B | 37 | 0 | 98.5 | 1.49 | 0 |
| 3 | B | 42 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 59.74 | 27.9 | 8.11 | 4.27 |
| 4 | I | 27 | 48.09 | 37.6 | 9.44 | 4.86 |
| 4 | I | 32 | 48.86 | 38.2 | 9.19 | 3.74 |
| 4 | I | 37 | 51.92 | 37.1 | 8.03 | 2.92 |
| 4 | I | 42 | 56.95 | 35.6 | 5.84 | 1.56 |
| 4 | I | 47 | 55.33 | 38.3 | 5.36 | 1.04 |
| 4 | P | 22 | 81.35 | 15.9 | 2.1 | 0.64 |
| 4 | P | 27 | 77.71 | 18.4 | 3.13 | 0.8 |
| 4 | P | 32 | 69.58 | 25.1 | 4.24 | 1.05 |
| 4 | P | 37 | 62.08 | 32.6 | 3.66 | 1.64 |
| 4 | P | 42 | 62.89 | 29.9 | 6.19 | 1.03 |
| 4 | P | 47 | 50 | 50 | 0 | 0 |
| 4 | B | 22 | 89.73 | 9.16 | 0.91 | 0.2 |
| 4 | B | 27 | 81.57 | 15.3 | 2.39 | 0.76 |
| 4 | B | 32 | 68.26 | 26.2 | 4.46 | 1.06 |
| 4 | B | 37 | 65 | 31.4 | 2.73 | 0.91 |
| 4 | B | 42 | 77.78 | 11.1 | 11.1 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 45.63 | 34.4 | 12.5 | 7.52 |
| 5 | I | 27 | 33.57 | 41.7 | 15.5 | 9.28 |
| 5 | I | 32 | 33.7 | 44.4 | 14.3 | 7.68 |
| 5 | I | 37 | 37.16 | 43.7 | 13.9 | 5.24 |
| 5 | I | 42 | 40.87 | 45.9 | 9.62 | 3.58 |
| 5 | I | 47 | 43.34 | 47.1 | 6.98 | 2.6 |
| 5 | P | 22 | 73.09 | 22 | 3.55 | 1.34 |
| 5 | P | 27 | 68.84 | 23.8 | 5.38 | 1.99 |
| 5 | P | 32 | 55.86 | 33 | 8.1 | 3.07 |
| 5 | P | 37 | 43.65 | 44.3 | 9.63 | 2.46 |
| 5 | P | 42 | 39.29 | 57.1 | 0 | 3.57 |
| 5 | B | 22 | 84.36 | 13.6 | 1.62 | 0.43 |
| 5 | B | 27 | 67.84 | 24.6 | 5.65 | 1.91 |
| 5 | B | 32 | 49.3 | 37.3 | 11.3 | 2.1 |
| 5 | B | 37 | 52.54 | 44.1 | 1.69 | 1.69 |

Cactus

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 93.1 | 6.4 | 0.44 | 0.05 |
| 1 | I | 32 | 97 | 2.8 | 0.16 | 0.05 |
| 1 | I | 37 | 97.6 | 2.18 | 0.16 | 0.05 |
| 1 | I | 42 | 98.2 | 1.63 | 0.13 | 0.04 |
| 1 | I | 47 | 98.2 | 1.58 | 0.17 | 0.04 |
| 1 | P | 22 | 90.1 | 7.82 | 1.38 | 0.73 |
| 1 | P | 32 | 95.4 | 4.16 | 0.4 | 0.06 |
| 1 | P | 37 | 97.8 | 2.03 | 0.14 | 0.01 |
| 1 | P | 42 | 98.8 | 1.13 | 0.03 | 0 |
| 1 | P | 47 | 99.3 | 0.66 | 0.03 | 0 |
| 1 | B | 22 | 91.9 | 6.83 | 0.98 | 0.31 |
| 1 | B | 32 | 98.1 | 1.75 | 0.17 | 0.02 |
| 1 | B | 37 | 99.1 | 0.82 | 0.07 | 0.01 |
| 1 | B | 42 | 99.5 | 0.5 | 0 | 0.02 |
| 1 | B | 47 | 99.7 | 0.28 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 81.6 | 16.1 | 1.2 | 0.46 | 0.05 | 0.56 |
| 2 | I | 37 | 90 | 7.78 | 1.5 | 0.42 | 0.07 | 0.22 |
| 2 | I | 32 | 88.5 | 9.62 | 1.19 | 0.34 | 0.03 | 0.29 |
| 2 | I | 42 | 90.4 | 8.06 | 1.03 | 0.24 | 0.04 | 0.2 |
| 2 | I | 47 | 92 | 6.8 | 0.77 | 0.28 | 0.03 | 0.15 |
| 2 | P | 22 | 78.8 | 17 | 1.66 | 0.77 | 0.07 | 1.66 |
| 2 | P | 32 | 89.6 | 8.69 | 1.12 | 0.35 | 0.04 | 0.22 |
| 2 | P | 37 | 90.3 | 8.14 | 1.13 | 0.24 | 0.02 | 0.15 |
| 2 | P | 42 | 92.8 | 6.59 | 0.43 | 0.05 | 0.05 | 0.05 |
| 2 | P | 47 | 94 | 3.13 | 2.82 | 0 | 0 | 0 |
| 2 | B | 22 | 86.2 | 10.9 | 1.34 | 0.6 | 0.06 | 0.95 |
| 2 | B | 32 | 91.8 | 6.58 | 0.95 | 0.41 | 0.03 | 0.22 |
| 2 | B | 37 | 91.8 | 6.79 | 1 | 0.36 | 0 | 0.09 |
| 2 | B | 42 | 92.2 | 7.34 | 0 | 0.46 | 0 | 0 |
| 2 | B | 47 | 97.6 | 0 | 2.38 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 93.6 | 4.65 | 1.79 |
| 3 | I | 32 | 0 | 95.4 | 3.4 | 1.16 |
| 3 | I | 37 | 0 | 95.1 | 3.31 | 1.61 |
| 3 | I | 42 | 0 | 95.1 | 3.99 | 0.87 |
| 3 | I | 47 | 0 | 94.5 | 4.59 | 0.94 |
| 3 | P | 22 | 0 | 92.3 | 4.79 | 2.96 |
| 3 | P | 32 | 0 | 96.3 | 2.81 | 0.91 |
| 3 | P | 37 | 0 | 96.3 | 3.15 | 0.59 |
| 3 | P | 42 | 0 | 97.1 | 2.54 | 0.36 |
| 3 | P | 47 | 0 | 91.5 | 6.38 | 2.13 |
| 3 | B | 22 | 0 | 95.7 | 2.82 | 1.44 |
| 3 | B | 32 | 0 | 96.3 | 2.79 | 0.87 |
| 3 | B | 37 | 0 | 95.6 | 3.61 | 0.83 |
| 3 | B | 42 | 0 | 96.6 | 3.45 | 0 |
| 3 | B | 47 | 0 | 77.8 | 22.2 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 42.8 | 40.4 | 11.4 | 5.4 |
| 4 | I | 32 | 46.2 | 39.6 | 9.64 | 4.53 |
| 4 | I | 37 | 47.5 | 40.7 | 8.64 | 3.12 |
| 4 | I | 42 | 51.2 | 37.5 | 6.77 | 4.62 |
| 4 | I | 47 | 53.1 | 35.4 | 7.18 | 4.31 |
| 4 | P | 22 | 42.6 | 38.4 | 11.8 | 7.22 |
| 4 | P | 32 | 55.5 | 34.6 | 7.22 | 2.66 |
| 4 | P | 37 | 48 | 43.8 | 6.3 | 1.89 |
| 4 | P | 42 | 58.7 | 30.3 | 6.42 | 4.59 |
| 4 | P | 47 | 77.8 | 11.1 | 11.1 | 0 |
| 4 | B | 22 | 63.6 | 27.1 | 6.28 | 3.05 |
| 4 | B | 32 | 55 | 36 | 6.93 | 2.11 |
| 4 | B | 37 | 52 | 39 | 7.32 | 1.63 |
| 4 | B | 42 | 63.6 | 36.4 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 29.1 | 42.2 | 17.2 | 11.4 |
| 5 | I | 32 | 28.2 | 45.5 | 17.2 | 9.07 |
| 5 | I | 37 | 35 | 45.1 | 11.7 | 8.15 |
| 5 | I | 42 | 40.1 | 43 | 11.6 | 5.23 |
| 5 | I | 47 | 37.5 | 51.8 | 7.14 | 3.57 |
| 5 | P | 22 | 33.9 | 39.9 | 15.6 | 10.7 |
| 5 | P | 32 | 40.6 | 40.8 | 12.7 | 5.82 |
| 5 | P | 37 | 34.6 | 48 | 13.4 | 4.04 |
| 5 | P | 42 | 21.7 | 56.5 | 4.35 | 17.4 |
| 5 | B | 22 | 54.9 | 30.7 | 9.15 | 5.31 |
| 5 | B | 32 | 39 | 42.4 | 12.6 | 6.05 |
| 5 | B | 37 | 25 | 55.6 | 8.33 | 11.1 |
| 5 | B | 42 | 100 | 0 | 0 | 0 |

ChinaSpeed

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 90.2 | 9.18 | 0.51 | 0.09 |
| 1 | I | 27 | 94.6 | 5.1 | 0.25 | 0.06 |
| 1 | I | 32 | 97.8 | 1.97 | 0.16 | 0.06 |
| 1 | I | 37 | 98 | 1.59 | 0.21 | 0.17 |
| 1 | I | 42 | 97.8 | 1.72 | 0.29 | 0.18 |
| 1 | I | 47 | 97.5 | 2.06 | 0.22 | 0.18 |
| 1 | P | 22 | 86.2 | 9.22 | 1.82 | 2.74 |
| 1 | P | 27 | 88.2 | 7.06 | 1.6 | 3.11 |
| 1 | P | 32 | 87.8 | 7.26 | 3.2 | 1.71 |
| 1 | P | 37 | 88.5 | 10.6 | 0.74 | 0.14 |
| 1 | P | 42 | 98.7 | 1.13 | 0.05 | 0.08 |
| 1 | P | 47 | 98.9 | 0.95 | 0.09 | 0.06 |
| 1 | B | 22 | 88.6 | 8.96 | 1.5 | 0.9 |
| 1 | B | 27 | 94.7 | 4.29 | 0.69 | 0.31 |
| 1 | B | 32 | 96.8 | 2.72 | 0.32 | 0.17 |
| 1 | B | 37 | 98 | 1.78 | 0.16 | 0.06 |
| 1 | B | 42 | 98.9 | 0.84 | 0.2 | 0.05 |
| 1 | B | 47 | 98.4 | 1.63 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 84.5 | 13 | 1.4 | 0.3 | 0.1 | 0.72 |
| 2 | I | 32 | 89.5 | 7.75 | 1.6 | 0.5 | 0.1 | 0.5 |
| 2 | I | 27 | 87.8 | 9.68 | 1.5 | 0.3 | 0 | 0.7 |
| 2 | I | 37 | 88.6 | 8.72 | 1.3 | 0.4 | 0.3 | 0.64 |
| 2 | I | 42 | 88.9 | 8.39 | 1.6 | 0.4 | 0.1 | 0.53 |
| 2 | I | 47 | 91 | 6.56 | 1.1 | 0.6 | 0.3 | 0.39 |
| 2 | P | 22 | 78 | 14.7 | 2 | 0.9 | 0.1 | 4.32 |
| 2 | P | 27 | 84.1 | 9.35 | 2 | 0.8 | 0.2 | 3.43 |
| 2 | P | 32 | 86.6 | 8.78 | 2.8 | 0.8 | 0.1 | 0.91 |
| 2 | P | 37 | 89.3 | 7.48 | 2.2 | 0.8 | 0 | 0.25 |
| 2 | P | 42 | 93.2 | 5.27 | 1.3 | 0.2 | 0 | 0 |
| 2 | P | 47 | 95.5 | 3.2 | 0.9 | 0.4 | 0 | 0 |
| 2 | B | 22 | 83.2 | 12.7 | 2 | 0.7 | 0.1 | 1.3 |
| 2 | B | 27 | 90.6 | 6.39 | 1.9 | 0.4 | 0.1 | 0.65 |
| 2 | B | 32 | 92.1 | 5.51 | 1.7 | 0.2 | 0.1 | 0.46 |
| 2 | B | 37 | 94 | 4.06 | 1.4 | 0.2 | 0 | 0.27 |
| 2 | B | 42 | 96.5 | 2.94 | 0.5 | 0.1 | 0 | 0.07 |
| 2 | B | 47 | 94.6 | 5.36 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 94.2 | 3.37 | 2.44 |
| 3 | I | 27 | 0 | 93.9 | 3.74 | 2.33 |
| 3 | I | 32 | 0 | 94.7 | 3 | 2.28 |
| 3 | I | 37 | 0 | 96.1 | 2.29 | 1.57 |
| 3 | I | 42 | 0 | 97.4 | 2.1 | 0.54 |
| 3 | I | 47 | 0 | 95.6 | 3.42 | 0.96 |
| 3 | P | 22 | 0 | 90.9 | 4.04 | 5.02 |
| 3 | P | 27 | 0 | 92.7 | 4 | 3.29 |
| 3 | P | 32 | 0 | 93.9 | 3.8 | 2.28 |
| 3 | P | 37 | 0 | 96.3 | 2.57 | 1.13 |
| 3 | P | 42 | 0 | 97.1 | 2.18 | 0.69 |
| 3 | P | 47 | 0 | 98.9 | 1.08 | 0 |
| 3 | B | 22 | 0 | 94.9 | 3.18 | 1.87 |
| 3 | B | 27 | 0 | 96.8 | 2.23 | 1 |
| 3 | B | 32 | 0 | 98 | 1.37 | 0.63 |
| 3 | B | 37 | 0 | 98.8 | 1.08 | 0.11 |
| 3 | B | 42 | 0 | 99.5 | 0.43 | 0.11 |
| 3 | B | 47 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 51.4 | 29.8 | 8.97 | 9.83 |
| 4 | I | 27 | 54.7 | 25.6 | 7.73 | 12 |
| 4 | I | 32 | 53 | 29 | 8.51 | 9.49 |
| 4 | I | 37 | 53.4 | 32.6 | 7.07 | 6.91 |
| 4 | I | 42 | 60.7 | 27 | 8.22 | 4.01 |
| 4 | I | 47 | 62.4 | 26.2 | 8.46 | 2.99 |
| 4 | P | 22 | 51.9 | 28.3 | 9.54 | 10.2 |
| 4 | P | 27 | 51.3 | 23.9 | 10.2 | 14.6 |
| 4 | P | 32 | 57.3 | 26.3 | 10.7 | 5.64 |
| 4 | P | 37 | 71.6 | 20.7 | 4.44 | 3.24 |
| 4 | P | 42 | 82 | 14.6 | 2.34 | 0.98 |
| 4 | P | 47 | 84.2 | 12.8 | 2.46 | 0.49 |
| 4 | B | 22 | 61.3 | 25.2 | 7.7 | 5.81 |
| 4 | B | 27 | 68.7 | 20.4 | 6.21 | 4.71 |
| 4 | B | 32 | 74.5 | 18.5 | 4.72 | 2.26 |
| 4 | B | 37 | 83.9 | 13.6 | 2.02 | 0.55 |
| 4 | B | 42 | 88.8 | 10.1 | 0.95 | 0.16 |
| 4 | B | 47 | 81.8 | 18.2 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 44 | 33.8 | 12.6 | 9.55 |
| 5 | I | 27 | 45.6 | 32.3 | 11.7 | 10.3 |
| 5 | I | 32 | 42.2 | 36.8 | 13.1 | 7.89 |
| 5 | I | 37 | 45.2 | 36.1 | 12 | 6.63 |
| 5 | I | 42 | 51.5 | 35.5 | 9.06 | 3.99 |
| 5 | I | 47 | 57.8 | 29.3 | 9.15 | 3.74 |
| 5 | P | 22 | 48 | 28 | 11.8 | 12.2 |
| 5 | P | 27 | 50.9 | 24.6 | 11.5 | 13 |
| 5 | P | 32 | 60.3 | 26.1 | 7.58 | 6.04 |
| 5 | P | 37 | 68.2 | 23 | 6.87 | 1.92 |
| 5 | P | 42 | 80.5 | 16.5 | 2.66 | 0.38 |
| 5 | P | 47 | 75 | 20.3 | 4.69 | 0 |
| 5 | B | 22 | 60.6 | 23.8 | 9.13 | 6.52 |
| 5 | B | 27 | 69.1 | 21 | 6.29 | 3.59 |
| 5 | B | 32 | 75.8 | 18.3 | 4.13 | 1.7 |
| 5 | B | 37 | 81.5 | 15.3 | 2.63 | 0.55 |
| 5 | B | 42 | 82.7 | 15.3 | 1 | 1 |

Kimono1

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 91.4 | 8.11 | 0.46 | 0.05 |
| 1 | I | 27 | 92.3 | 7.38 | 0.26 | 0.02 |
| 1 | I | 32 | 96.8 | 3.03 | 0.11 | 0.02 |
| 1 | I | 37 | 98.4 | 1.55 | 0.07 | 0.01 |
| 1 | I | 42 | 99.3 | 0.66 | 0.04 | 0.01 |
| 1 | I | 47 | 99.4 | 0.58 | 0.03 | 0 |
| 1 | P | 22 | 90.2 | 8.52 | 1.09 | 0.2 |
| 1 | P | 27 | 92.7 | 6.77 | 0.51 | 0.07 |
| 1 | P | 32 | 96.9 | 2.93 | 0.16 | 0.02 |
| 1 | P | 37 | 98.3 | 1.61 | 0.09 | 0 |
| 1 | P | 42 | 98.7 | 1.24 | 0.07 | 0 |
| 1 | P | 47 | 99.4 | 0.62 | 0.02 | 0 |
| 1 | B | 22 | 92.7 | 6.47 | 0.72 | 0.13 |
| 1 | B | 27 | 96.8 | 2.94 | 0.2 | 0.03 |
| 1 | B | 32 | 98.6 | 1.31 | 0.07 | 0 |
| 1 | B | 37 | 99 | 0.95 | 0.06 | 0 |
| 1 | B | 42 | 99.4 | 0.54 | 0.03 | 0 |
| 1 | B | 47 | 99.1 | 0.89 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 80.3 | 17.5 | 1.1 | 0.4 | 0.04 | 0.5 |
| 2 | I | 32 | 86 | 12.3 | 1 | 0.3 | 0 | 0.3 |
| 2 | I | 27 | 83.4 | 14.5 | 1.2 | 0.4 | 0.04 | 0.5 |
| 2 | I | 37 | 82.3 | 16.3 | 0.5 | 0.2 | 0.07 | 0.6 |
| 2 | I | 42 | 84.5 | 15.1 | 0.3 | 0 | 0 | 0 |
| 2 | I | 47 | 82.3 | 17.7 | 0 | 0 | 0 | 0 |
| 2 | P | 22 | 77.5 | 18.7 | 1.7 | 0.8 | 0.07 | 1.3 |
| 2 | P | 27 | 80.8 | 16.5 | 1.4 | 0.6 | 0.05 | 0.7 |
| 2 | P | 32 | 85.2 | 13.1 | 1 | 0.4 | 0.03 | 0.4 |
| 2 | P | 37 | 78.4 | 20.2 | 0.6 | 0.3 | 0 | 0.5 |
| 2 | P | 42 | 78 | 22 | 0 | 0 | 0 | 0 |
| 2 | P | 47 | 100 | 0 | 0 | 0 | 0 | 0 |
| 2 | B | 22 | 77.9 | 18 | 1.9 | 1 | 0.08 | 1.1 |
| 2 | B | 27 | 81.1 | 16.7 | 1.2 | 0.5 | 0.04 | 0.5 |
| 2 | B | 32 | 83.6 | 15 | 0.9 | 0.3 | 0.03 | 0.2 |
| 2 | B | 37 | 79.3 | 20.7 | 0 | 0 | 0 | 0 |
| 2 | B | 42 | 86.2 | 13.8 | 0 | 0 | 0 | 0 |
| 2 | B | 47 | 100 | 0 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 84.1 | 10.4 | 5.48 |
| 3 | I | 27 | 0 | 79.8 | 13.4 | 6.83 |
| 3 | I | 32 | 0 | 78.8 | 14.6 | 6.67 |
| 3 | I | 37 | 0 | 78.7 | 16.8 | 4.49 |
| 3 | I | 42 | 0 | 73.9 | 17.4 | 8.7 |
| 3 | I | 47 | 0 | 79.6 | 16.3 | 4.08 |
| 3 | P | 22 | 0 | 84 | 10.6 | 5.45 |
| 3 | P | 27 | 0 | 83.9 | 10.8 | 5.2 |
| 3 | P | 32 | 0 | 85 | 10.8 | 4.26 |
| 3 | P | 37 | 0 | 82.6 | 13.1 | 4.3 |
| 3 | P | 42 | 0 | 83.3 | 14.3 | 2.38 |
| 3 | B | 22 | 0 | 86.1 | 9.66 | 4.28 |
| 3 | B | 27 | 0 | 88.2 | 8.1 | 3.67 |
| 3 | B | 32 | 0 | 88.3 | 8.61 | 3.12 |
| 3 | B | 37 | 0 | 85.4 | 14.6 | 0 |
| 3 | B | 42 | 0 | 100 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 23.3 | 44.5 | 18.5 | 13.8 |
| 4 | I | 27 | 27.5 | 43 | 15.5 | 14 |
| 4 | I | 32 | 20.1 | 53 | 18.1 | 8.72 |
| 4 | I | 37 | 32.3 | 45.2 | 0 | 22.6 |
| 4 | I | 42 | 50 | 0 | 50 | 0 |
| 4 | I | 47 | 50 | 50 | 0 | 0 |
| 4 | P | 22 | 27.5 | 43.2 | 17.1 | 12.2 |
| 4 | P | 27 | 35.9 | 41.4 | 13.5 | 9.25 |
| 4 | P | 32 | 41.7 | 39.9 | 11.2 | 7.24 |
| 4 | P | 37 | 47.8 | 34.8 | 13 | 4.35 |
| 4 | B | 22 | 32.9 | 42.5 | 15.4 | 9.17 |
| 4 | B | 27 | 43.2 | 40.2 | 10.7 | 5.89 |
| 4 | B | 32 | 50 | 29.2 | 20.8 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 12.9 | 40 | 22.6 | 24.5 |
| 5 | I | 27 | 14.4 | 33.8 | 26.6 | 25.3 |
| 5 | I | 32 | 11.5 | 35.9 | 29.5 | 23.1 |
| 5 | I | 37 | 0 | 42.9 | 14.3 | 42.9 |
| 5 | P | 22 | 17.7 | 39.6 | 22.6 | 20.1 |
| 5 | P | 27 | 27.9 | 40.3 | 16.8 | 15 |
| 5 | P | 32 | 28.6 | 40.3 | 22.1 | 9.09 |
| 5 | P | 37 | 0 | 70 | 20 | 10 |
| 5 | B | 22 | 24.6 | 39.9 | 19.7 | 15.9 |
| 5 | B | 27 | 33.2 | 45.9 | 11.8 | 9.09 |
| 5 | B | 32 | 22.2 | 55.6 | 22.2 | 0 |

ParkScene

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 1 | I | 22 | 94.2 | 5.3 | 0.4 | 0.06 |
| 1 | I | 27 | 95.6 | 4 | 0.3 | 0.07 |
| 1 | I | 32 | 96.8 | 2.9 | 0.3 | 0.01 |
| 1 | I | 37 | 97.1 | 2.6 | 0.2 | 0.03 |
| 1 | I | 42 | 98 | 1.9 | 0.1 | 0.01 |
| 1 | I | 47 | 98.6 | 1.4 | 0.1 | 0.01 |
| 1 | P | 22 | 96.4 | 3 | 0.4 | 0.13 |
| 1 | P | 27 | 96.4 | 3.1 | 0.4 | 0.12 |
| 1 | P | 32 | 97.1 | 2.6 | 0.3 | 0.05 |
| 1 | P | 37 | 97.9 | 1.9 | 0.1 | 0.01 |
| 1 | P | 42 | 98.5 | 1.4 | 0.1 | 0.01 |
| 1 | P | 47 | 98.6 | 1.3 | 0.1 | 0 |
| 1 | B | 22 | 98 | 1.7 | 0.2 | 0.04 |
| 1 | B | 27 | 97.5 | 2.2 | 0.3 | 0.04 |
| 1 | B | 32 | 98.4 | 1.5 | 0.1 | 0.01 |
| 1 | B | 37 | 99.4 | 0.6 | 0 | 0 |
| 1 | B | 42 | 99.7 | 0.2 | 0 | 0 |
| 1 | B | 47 | 99.5 | 0.5 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 1-2% | 2% | 2-3% | 3% | >3% |
|---|---|---|---|---|---|---|---|---|
| 2 | I | 22 | 84 | 13.6 | 1.34 | 0.47 | 0.07 | 0.48 |
| 2 | I | 32 | 89 | 8.98 | 1.32 | 0.37 | 0.02 | 0.23 |
| 2 | I | 27 | 86 | 11.5 | 1.21 | 0.43 | 0.07 | 0.49 |
| 2 | I | 37 | 90 | 8.23 | 1.11 | 0.29 | 0.01 | 0.18 |
| 2 | I | 42 | 91 | 7.91 | 1.06 | 0.28 | 0.03 | 0.03 |
| 2 | I | 47 | 89 | 8.95 | 1.28 | 0.32 | 0 | 0.16 |
| 2 | P | 22 | 90 | 7.61 | 1.45 | 0.54 | 0.08 | 0.53 |
| 2 | P | 27 | 89 | 8.36 | 1.62 | 0.62 | 0.07 | 0.5 |
| 2 | P | 32 | 91 | 7.12 | 1.26 | 0.37 | 0.05 | 0.15 |
| 2 | P | 37 | 92 | 6.23 | 1.05 | 0.31 | 0.01 | 0.06 |
| 2 | P | 42 | 91 | 8.45 | 0.61 | 0.28 | 0.05 | 0.09 |
| 2 | P | 47 | 87 | 12.5 | 0.69 | 0 | 0 | 0 |
| 2 | B | 22 | 92 | 5.48 | 1.47 | 0.46 | 0.07 | 0.31 |
| 2 | B | 27 | 93 | 5.15 | 1.42 | 0.41 | 0.03 | 0.22 |
| 2 | B | 32 | 95 | 3.84 | 1.02 | 0.34 | 0 | 0.04 |
| 2 | B | 37 | 96 | 2.84 | 0.3 | 0.42 | 0 | 0.06 |
| 2 | B | 42 | 95 | 4.32 | 1 | 0 | 0 | 0 |
| 2 | B | 47 | 79 | 21.2 | 0 | 0 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 3 | I | 22 | 0 | 95 | 3.5 | 1.26 |
| 3 | I | 27 | 0 | 95 | 3.3 | 1.27 |
| 3 | I | 32 | 0 | 96 | 3.1 | 0.94 |
| 3 | I | 37 | 0 | 94 | 4.5 | 1.22 |
| 3 | I | 42 | 0 | 93 | 5.5 | 1.6 |
| 3 | I | 47 | 0 | 93 | 5.7 | 1.55 |
| 3 | P | 22 | 0 | 97 | 2.1 | 1.21 |
| 3 | P | 27 | 0 | 96 | 2.5 | 1.03 |
| 3 | P | 32 | 0 | 97 | 2.3 | 0.61 |
| 3 | P | 37 | 0 | 96 | 2.8 | 0.96 |
| 3 | P | 42 | 0 | 95 | 4.1 | 1.09 |
| 3 | P | 47 | 0 | 95 | 3.3 | 1.64 |
| 3 | B | 22 | 0 | 98 | 1.3 | 0.64 |
| 3 | B | 27 | 0 | 98 | 1.4 | 0.54 |
| 3 | B | 32 | 0 | 99 | 1.2 | 0.28 |
| 3 | B | 37 | 0 | 99 | 1.3 | 0.16 |
| 3 | B | 42 | 0 | 96 | 4.3 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 4 | I | 22 | 42.6 | 40 | 12 | 5.46 |
| 4 | I | 27 | 45.8 | 39 | 10 | 4.72 |
| 4 | I | 32 | 50.8 | 38 | 7.9 | 3.15 |
| 4 | I | 37 | 47.2 | 40 | 9.2 | 3.1 |
| 4 | I | 42 | 37.9 | 48 | 9.2 | 4.6 |
| 4 | I | 47 | 45.1 | 39 | 11 | 4.51 |
| 4 | P | 22 | 55.6 | 32 | 8.2 | 3.98 |
| 4 | P | 27 | 49.5 | 38 | 9 | 3.88 |
| 4 | P | 32 | 52 | 38 | 7.3 | 2.29 |
| 4 | P | 37 | 47.3 | 42 | 7.9 | 3.01 |
| 4 | P | 42 | 55.2 | 31 | 9.9 | 3.77 |
| 4 | P | 47 | 100 | 0 | 0 | 0 |
| 4 | B | 22 | 68 | 25 | 5 | 2.1 |
| 4 | B | 27 | 62.2 | 31 | 5 | 1.66 |
| 4 | B | 32 | 62.9 | 32 | 3.5 | 1.15 |
| 4 | B | 37 | 57.6 | 35 | 6.4 | 0.85 |
| 4 | B | 42 | 50 | 50 | 0 | 0 |

| Block coeffs | Picture Type | QP | 1% | 2% | 3% | >3% |
|---|---|---|---|---|---|---|
| 5 | I | 22 | 26.2 | 44 | 18 | 11.4 |
| 5 | I | 27 | 30.4 | 45 | 16 | 8.95 |
| 5 | I | 32 | 34.2 | 43 | 16 | 6.66 |
| 5 | I | 37 | 33.3 | 42 | 15 | 9.73 |
| 5 | I | 42 | 23.7 | 49 | 14 | 12.9 |
| 5 | I | 47 | 32 | 48 | 16 | 4 |
| 5 | P | 22 | 40.4 | 38 | 13 | 8.59 |
| 5 | P | 27 | 34.9 | 42 | 15 | 8.13 |
| 5 | P | 32 | 34.1 | 46 | 14 | 5.89 |
| 5 | P | 37 | 30 | 48 | 16 | 6.33 |
| 5 | P | 42 | 32.6 | 48 | 17 | 2.17 |
| 5 | B | 22 | 54.4 | 32 | 9.3 | 4.7 |
| 5 | B | 27 | 44.5 | 41 | 10 | 4.42 |
| 5 | B | 32 | 41.8 | 44 | 11 | 3.52 |
| 5 | B | 37 | 36.5 | 43 | 17 | 3.17 |

# References


[1] Tinku Acharya and Ping-Sing Tsai. *JPEG2000 Standard for Image Compression: Concepts, Algorithms and VLSI Architectures*. John Wiley and Sons, Hoboken, New Jersey, 2005.

[2] S. Ameer. *Investigating Polynomial Fitting Schemes for Image Compression*. PhD thesis, University of Waterloo, 2009.

[3] S. Ameer and O. Basir. *Image Compression through Optimized Linear Mapping and Parametrically Generated Features*. 2008.

[4] Salah Ameer and Otman A. Basir. A simple three-parameter surface fitting scheme for image compression. In *Int. Conf. on Computer Vision Theory and Applications VISAPP*, pages 101–106, Setbal, Portugal, 2006.

[5] H. Aydinoglu and M. Hayes. Image coding with polynomial transforms,. 1:520–524, Conference Record of the Thirtieth Asilomar Conference: Signals, Systems and Computers.

[6] H. Schwarz B. Schumitsch and T. Wiegand. *Inter-frame optimization of transform coefficient selection in hybrid video coding.* Dec. 2004.

[7] R. Baseri and J. Modestino. Region-based coding of images using a spline model. *Proc. IEEE Int. Conf. Image Processing*, 3:866–870, 1994.

[8] T. Berger. *Rate Distortion Theory-A Mathematical Basis for Data Compression.* Prentice-Hall, Englewood Cliffs, NJ, 1971.

[9] S. Biswas. Segmentation based compression for gray level images. *Pattern Recognition*, pages 1501–1517, 2003.

[10] G. Bjontegaard and K. Lillevold. Context-adaptive vlc (cavlc) coding of coefficients. *JVT-C028, Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG*, May 2002.

[11] J. Bosworth and S. Acton. Segmentation-based image coding by morphological local monotonicity. pages 65–69, 2000.

[12] V. Bruni and D. Vitulano. Combined image compression and denoising using wavelets. *Image Commun.*, 22(1), January 2007.

[13] C. Cabrelli and U. Molter. Automatic representation of binary images. *IEEE Trans. PAMI*, 12(12):1190–1196, 1990.

[14] Lavagetto F Pampolini M Cermelli, M. A fast algorithm for region-oriented texture coding. *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference*, 5:285–288, 1994.

[15] C. Chang and B. Girod. Direction-adaptive discrete wavelet transform for image compression. ieee trans. ip. 16(5):1289–1302, 2007.

[16] Thomas M. Cover and Joy A. Thomas. *Elements of information theory.* Wiley-Interscience, New York, NY, USA, 1991.

[17] M. Crouse and K. Ramchandran. Joint thresholding and quantizer selection for transform image coding: Entropy constrained analysis and applications to baseline JPEG. *IEEE Transactions on Image Processing.*, 6(2):285–297, Feb. 1997.

[18] E. Delp and O. Mitchell. Image compression using block truncation. ieee trans. comm. (9):1335–1342, 1979.

[19] Laurent Demaret, Nira Dyn, and Armin Iske. Image compression by linear splines over adaptive triangulations. *Signal Process.*, 86(7):1604–1616, July 2006.

[20] W. Effelsberg and R. Steinmetz. *Video Compression Techniques.* Number v. 1. dpunkt-Verlag, 1998.

[21] D. Giusto F. DeNatale, G. Desoli and G. Vernazza. Polynomial approximation and vector quantization: a region-based integration. *IEEE Trans Comm.*, 43(2/3/4):198–206, 1995.

[22] Y. Fisher. *Fractal image compression: theory and application.* New York: Springer, 1995.

[23] B. Furht. A survey of multimedia compression techniques and standards part i: Jpeg standard. real-time imaging. pages 49–76, 1995.

[24] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, 1992.

[25] Michael Gilge. Region-oriented transform coding (ROTC) of images. In *International Conference on Acoustics, Speech, and Signal Processing*, 1990.

[26] B. Girod. Efficiency analysis of multihypothesis motion-compensated prediction for video coding. ieee transactions on image processing, volume = 9, number = 2, pages =.

[27] B. Girod. The efficiency of motion-compensating prediction for hybrid coding of video sequences. *IEEE Journal on Selected Areas in Communications*, 5(7):1140–1154, August 1987.

[28] B. Girod. Motion-compensating prediction with fractional-pel accuracy. *IEEE Transactions on Communications*, 41(4):604–612, Apr. 1993.

[29] V.K. Goyal. Transform coding with integer-to-integer transforms. *IEEE Transactions on Information Theory*, 46.

[30] J. Hanson. *Understanding Video: Applications, Impact and Theory.*

[31] Z. He and S. K. Mitra. A unified rate-distortion analysis framework for transform coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(12):1221–1236, Dec. 2001.

[32] S. S. Hemami and R. M. Gray. Subband-coded image reconstruction for lossy packet networks. *Trans. Img. Proc.*, 6(4):523–539, April 1997.

[33] I. Hussain and T. Reed. Segmentation-based image compression with enhanced treatment of textured regions. *28th Asilomar Conf. on Signals, Systems and Computers*, pages 965–969, 1994.

[34] M. Luttrell J. Wen and J. Villasenor. Trellis-based R-D optimal quantization in H.263+. *IEEE Transaction on Image Processing*, 9(8):1431–1434, Aug. 2000.

[35] A. Jain. Image data compression: a review. proc. ieee. 69(3):349–401.

[36] J. Jiang. Image compression with neural networks  a survey. signal processing: Image communication. pages 737–760, 1999.

[37] A. Kaup and T. Aach. Segment-oriented coding of texture images based on successive approximation. pages 197–200, April 1994.

[38] A. Kaup and T. Aach. Coding of segmented images using shape-independent basis functions. *Trans. Img. Proc.*, 7(7):937–947, July 1998.

[39] J. C. Kieffer. A survey of the theory of source coding with a fidelity criterion. *IEEE Transactions on Information Theory*, 39(5):1473–1490, Sep. 1993.

[40] V. T. Kieu and D. T. Nguyen. Surface fitting approach for reducing blocking artifacts in low bitrate dct decoded images. pages 150–153, 2001.

[41] Hwang-Soo Kim and Jae-Young Lee. Image coding by fitting rbf-surfaces to subimages. *Pattern Recogn. Lett.*, 23(11):1239–1251, September 2002.

[42] R.Chauhan L. Kaur and S. Saxena. Adaptive compression of medical ultrasound images. ieee proc.-vis. image signal process. 153(2):185–190, 2006.

[43] A. Laha, N. R. Pal, and B. Chanda. Design of vector quantizer for image compression using self-organizing feature map and surface fitting. *Trans. Img. Proc.*, 13(10):1291–1303, October 2004.

[44] W. Li and Y. Zhang. Vector-based signal processing and quantization for image and video compression. proc. ieee. 83(2):317–335, 1995.

[45] Y. Lim and K. Park. Image segmentation and approximation through surface type labeling and region merging. *Electronics Letters*, 24(22):1380–1381, Oct. 1988.

[46] Y. Lim and K. Park. Image segmentation and approximation through surface type labelling and region merging. *Elect. Lett.*, 24(22):1380–1381, 1988.

[47] Jianyu Lin and M. J.T. Smith. New perspectives and improvements on the symmetric extension filter bank for subband/wavelet image compression. *Trans. Img. Proc.*, 17(2), February 2008.

[48] Y. Lin and P. Vaidyanathan. Theory and design of two-dimensional filter banks: a review. multidimensional systems and signal processing. pages 263–330, 1996.

[49] Tianyu Lu, Zisheng Le, and D. Y. Y. Yun. Piecewise linear image coding using surface triangulation and geometric compression. In *Proceedings of the Conference on Data Compression*, pages 410–. IEEE Computer Society, 2000.

[50] A. Constantinides M. Biggar, O. Morris. Segmented-image coding: performance comparison with the discrete cosine transform. volume 135, pages 121–132, 1988.

[51] M. Unser M. Eden and R. Leonardi. Polynomial representation of pictures. signal processing. pages 385–393, 1986.

[52] J. Nieweglowski M. Karczewics and P. Haavisto. Video coding using motion compensation with polynomial motion vector fields. *Signal Processing: Image Communication*, pages 63–91, 1997.

[53] D. Marpe, H. Schwarz, and T. Wiegand. Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):620–636, July 2003.

[54] X. Muñoz, J. Freixenet, X. Cufí, and J. Martí. Strategies for image segmentation combining region and boundary information. *Pattern Recogn. Lett.*, 24(1-3):375–392, January 2003.

[55] et al. O. Egger. *High-performance compression of visual information A tutorial reviewPart I: Still Picture*, volume 87. 1999.

[56] A. Ortega and K. Ramchandran. Rate-distortion methods for image and video compression. *IEEE Signal Processing Magazine*.

[57] J. Casas P Salembier, P. Brigger and M. Pardas. Morphological operators for image and video compression. volume 5, pages 881–898, 1996.

[58] W.B. Pennebaker and J.L. Mitchell. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1992.

[59] K. Ramchandran, A. Ortega, and M. Vetterli. Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders. *IEEE Transactions on Image Processing*, 3(5):533–545, September 1994.

[60] X. Ran and N. Farvardin. A perceptually motivated three-component image modelpart ii: applications to image compression. *Trans. Img. Proc.*, 4(4):430–447, April 1995.

[61] Iain E. G. Richardson. *H.264 and MPEG-4 Video Compression: Video Coding for Next-Generation Multimedia.* Chichester; Hoboken, NJ : Wiley, 2003.

[62] Y. Roterman and M. Porat. Progressive image coding using regional color correlation. *4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications, Croatia*, pages 65–70, 2003.

[63] A. Tamhankar S. Kwon and K. R. Rao. An overview of the h.264/mpeg-4 part 10. *EC-VIP-MC 2003, 4th EURASIP Conference*, 2.

[64] H. Jong S. Wang, L. Kuo and Z. Wu. Representing images using points on image surfaces. *IEEE Trans. IP*, 14(82):1043–1056, 2005.

[65] P. Salembier and F. Marqus. Region-based representations of image and video: Segmentation tools for multimedia services, 1999.

[66] P. Salembier and M. Pardas. Hierarchical morphological segmentation for image sequence coding. *Trans. Img. Proc.*, 3(5):639–651, September 1994.

[67] D. Salomon. *Data Compression.* Springer-Verlag, New York, 2004.

[68] J. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. ieee trans. sp. 41(12):3445–3462, 1993.

[69] Yair Shoham and Allen Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(9):1445–1453, September 1988.

[70] T. Sikora. *Trends and perspectives in image and video coding.*

[71] Sarvajit S. Sinha and Brian G. Schunck. A two-stage algorithm for discontinuity-preserving surface reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(1):36–55, January 1992.

[72] P. Strobach. *Quadtree-structured recursive plane decomposition coding of images,*, volume 39. June 1991.

[73] G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, Nov. 1998.

[74] Gary Sullivan. Joint video team (JVT) of iso/iec mpeg and itu-t vceg. adaptive quantization encoding technique using an equal expected-value rule. iso/iec jtc1/sc29/wg11 and itu-t sg16 q.6. Jan. 2005.

[75] A. Joch F. Kossentini T. Wiegand, H. Schwarz and G. J. Sullivan. *Rate constrained coder control and Comparison of video coding standards.*, volume 13. Jul. 2003.

[76] G. Bjontegaard T. Wiegand, G. J. Sullivan and A. Luthra. Overview of the h.264/avc video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7), Jul. 2003.

[77] G. Sullivan TK Tan and T. Wedi. Recommendedsimulation common conditions for coding efficiency experiments revision 1. *ITUT Standardization Sector, Document VCEG-AE010*, Jan. 2007.

[78] T. Watanabe. Picture coding employing b-spline surfaces with multiple vertices. *Elect. and Comm. in Japan Part I*, 80(2):55–65, 1997.

[79] Brendt Wohlberg and Gerhard de Jager. A review of the fractal image coding literature, ieee trans. ip, 1999.

[80] B. Sankur Y. Yemez and E. Anarim.

[81] E.-H. Yang and L. j Wang. Joint Optimization of Run-Length Coding, Huffman Coding, and Quantization Table with Complete Baseline JPEG Decoder Compatibility. 18(1):63–74, January 2009.

[82] E.-H. Yang and S. y. Shen. Distortion program-size complexity with respect to a fidelity criterion and rate distortion function. *IEEE Transactions on Information Theory*, 39(1):288–292, 1993.

[83] E.-H. Yang and X. Yu. On Joint Optimization of Motion Compensation, Quantization and Baseline Entropy Coding in H.264 with Complete Decoder Compatibility. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2:325–328, Mar. 2005.

[84] E.-H. Yang and X. Yu. Optimal soft decision quantization design for H.264. *Proc. of the 9th Canadian Workshop on Information Theory (CWIT'2005)*, pages 223–226, Jun. 2005.

[85] E.-H. Yang and X. Yu. Rate Distortion Optimization of H.264 with Main Profile compatibility. *IEEE International Symposium on Information Theory*, pages 282–286, Jul. 2006.

[86] E.-H. Yang and X. Yu. Rate distortion optimization for h.264 inter-frame coding: A general framework and algorithms. *IEEE Trans. On Image Processing*, 16(7):1774–1784, Jul. 2007.

[87] E.-H. Yang and X. Yu. Soft decision quantization for H.264 with main profile compatibility. *IEEE Transaction on Circuit and Systems for Video Technology*, 19(1):122–127, January 2009.

[88] Xiaoyuan Yang, Hui Ren, and Bo Li. Embedded zerotree wavelets coding based on adaptive fuzzy clustering for image compression. *Image Vision Comput.*, 26(6), June 2008.