Entropy Coding and Post-processing for Image and Video Coding

FONG, Yiu Leung

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Philosophy

in

Electronic Engineering

The Chinese University of Hong Kong

August 2010


香港中文大學
大學
統系館書圖
27 AUG 2012
UNIVERSITY
LIBRARY SYSTEM
THE CHINESE UNIVERSITY OF HONG KONG

Abstract

This thesis consists of two parts which investigate two problems in image and video coding. The first part focuses on the development of a new context-based arithmetic coding scheme for JPEG. The second part studies the application of a post-processing technique for deblocking H.264 compressed video.

Context-based arithmetic coding is the state-of-the-art entropy coding method used in the video coding standards H.264 and AVS. It is widely known to have outstanding compression capability on motion-compensated video content, and is able to significantly outperform other entropy coding methods. In the first part of this thesis, a new context-based arithmetic coding scheme is developed for the popular image compression standard JPEG with the aim to enhance its entropy coding efficiency. In particular, the proposed scheme first uses a modified H.264 significance map coding technique to code the nonzero coefficient flags and End-of-Block (EOB) decisions of transform coefficients for an image block, and then uses contexts that exploit information from both the magnitudes and zig-zag scanning positions of transform coefficients to code the 'LEVEL' symbols. The performance of the proposed scheme was tested on coding the quantized discrete cosine transform (DCT) coefficients. Experimental results show that the proposed scheme is able to outperform JPEG two-pass Huffman coding in terms of average file size reduction by 12%-15%. The proposed scheme also demonstrates superior compression efficiency to JPEG arithmetic coding as well as other existing entropy coding methods for JPEG.

Video signals are often compressed to smaller data size in order to satisfy bandwidth limitation in signal transmission or storage requirement in memory devices. However, the compression process generally introduces compression artifacts to the original video signal and results in degradation of visual quality in the compressed video.

Post-processing is a technique that can be used to improve the visual quality of the compressed video by using signal restoration or enhancement algorithms without incurring extra cost such as change in data format or increase in data size for the video being processed. In the second part of this thesis, the application of an image post-processing technique for deblocking compressed video is studied. In particular, the technique is applied on H.264 compressed video and its performance is investigated.

摘要

本論文分爲兩部分以探討兩個關於圖像和視頻編碼的問題。第一部分是關於一項新的並適用於 JPEG 的算術編碼技術的發展。第二部分是關於一項後處理技術在 H.264 視頻去塊中的應用。

基於內文的算術編碼 (context-based arithmetic coding) 是目前視頻編碼標準 H.264 及 AVS 中最先進的熵編碼技術。此技術在已進行移動補償的視頻數據上具有卓越的壓縮能力, 並且對於其他熵編碼方法在性能上有著顯著的優勢。本論文的第一部分將探討一項新的並且適用於 JPEG 的基於內文的算術編碼技術之發展, 目的是要增強 JPEG 的熵編碼壓縮效率。本論文提出的方法首先使用一項經修改的 H.264 'significance map coding' 技術對一個圖像塊中的變換係數的非零係數旗號 (nonzero coefficient flags) 及圖像塊終端決策 (End-of-Block decisions) 進行編碼, 然後使用能有效地利用變換係數中的量 (magnitudes) 及鋸齒形掃描位置 (zig-zag scanning positions) 中的冗餘信息的編碼環境 (contexts) 對該圖像塊中剩餘的高度符號 (LEVEL) 進行編碼。此方法在已量化的 discrete cosine transform (DCT) 係數上的編碼效率已被測試。實驗結果顯示, 此方法能夠在平均文件大小的減少上超越 JPEG 中的 two-pass Huffman coding 方法達 12%-15%。此外, 此方法對於 JPEG 中的算術編碼方法以及其他現有的適用於 JPEG 的熵編碼方法亦能展示較優越的壓縮效率。

視頻信號往往被壓縮到較小的數據大小以滿足信號傳輸時的帶寬限制以及儲存設備對於數據的空間要求。然而, 壓縮過程在普遍的情況下均會對原本的視頻信號造成壓縮缺陷, 以至降低了壓縮視頻的視覺質量。於此, 後處理技術可以透過使用信號修復或信號增強算法以改善壓縮視頻的視覺質量而同時不會對視頻帶來額

外的代價例如數據格式的改變或數據大小的增加。有見及此，本論文在第二部分對於一項圖像後處理技術在壓縮視頻去塊中的應用進行了研究。更準確來說，此研究調查該後處理技術在 H.264 視頻中的應用及性能。

Acknowledgement

I would like to thank my supervisor, Professor W.K.Cham, for his guidance in research and advice on the content of this thesis.

Contents

1. Introduction

Image and video are widely used in communications. However, image and video data are generally large in size so that compression on these data is necessary. Entropy coding and post-processing are compression techniques that can be cascaded to the back-end of an image or video coding scheme to improve its compression efficiency without requiring any change to the original scheme. Hence, they can be used to improve the compression efficiency of existing image and video coding schemes with minimal change to the schemes. In this thesis, new entropy coding and post-processing techniques to improve the compression efficiency of existing image and video coding schemes are proposed and studied. In particular, in the first part of this thesis, a new context-based arithmetic coding scheme for improving the entropy coding efficiency of JPEG is presented. In the second part, the application of a post-processing technique for deblocking H.264 compressed video is investigated.

2. Background and Motivation

2.1 Context-Based Arithmetic Coding

Context-based arithmetic coding is the state-of-the-art entropy coding method used in the video coding standards H.264 [1][2] and AVS [3]. It exhibits outstanding compression efficiency on the quantized transform coefficients of motion-compensated video content. It is widely known to be able to significantly outperform other entropy coding methods. For instance, the context-based arithmetic coding scheme of H.264, known as the Context-Based Adaptive Binary Arithmetic Coding (CABAC), has been reported in [4] to outperform its variable-length coding counterpart, Context-Adaptive Variable Length Coding (CAVLC), by 9%-14% in terms of average bit-rate savings on standard video sequences.

Conventional entropy coding methods, such as Huffman coding [5], suffer from the absence of exploitation of local information of quantized transform coefficients and the limitation to assign only codewords of integral length. Context-based arithmetic coding solves the first issue by using the concept of contexts. Contexts are coding environments which change according to local information such as the magnitudes of previously coded coefficients in an image block or the zig-zag scanning position of a transform coefficient in an image block. Given the local information, such as the magnitudes of previously coded coefficients, a specific context is selected and the probability of the current symbol is estimated independently within that context. In such a method, local information can be exploited to allow a more accurate probability estimation of symbols and hence reduces the amount of information to be transmitted.

Context-based arithmetic coding solves the second issue by using arithmetic coding [6][7][8]. Based on the concept of recursive subdivision of a real number line, arithmetic codes are able to represent symbols with fractional code length. Therefore,

symbols which should be coded with fractional number of bits can be represented efficiently using arithmetic codes. For instance, symbols with probabilities higher than 0.5, which should be coded with less than one bit, can be represented efficiently using arithmetic codes, while other variable-length codes must assign at least one bit for the same symbol, leaving potentially a substantial amount of redundancy in the codes.

In this thesis, a new context-based arithmetic coding scheme is developed with the aim to enhance the compression efficiency of JPEG entropy coding. The proposed scheme uses contexts that exploit local information from both the magnitudes of previously coded coefficients in a block and zig-zag scanning positions of transform coefficients in a block to code the transform coefficients efficiently. Experimental results show that the proposed scheme outperforms JPEG two-pass Huffman coding in terms of average file size reduction by 12%-15%.

A number of approaches to improve the compression efficiency of JPEG entropy coding were proposed [9][10][11]. In [9], the author proposed to modify the run-level coding process of JPEG Huffman coding by pairing a non-zero coefficient with subsequent run of zeros as oppose to pairing it with preceding run of zeros. Such modification enables the use of position dependent Huffman codes which are optimized for the position of the non-zero coefficient denoted by the run-level pair. In [10], the author proposed a modification to the DC coefficient prediction process of JPEG. The proposed method predicts the DC coefficient from the block above instead of from the previously coded block. The author showed that such modification can improve the compression efficiency of JPEG arithmetic coding. In [11], the author proposed a new AC coefficient coding algorithm for JPEG arithmetic coding. The algorithm improves the compression efficiency of the EOB symbols by coding potentially fewer EOB decisions in each image block.

Other approaches to improve the compression efficiency of JPEG through lossless re-encoding were also proposed [12][13]. However, these approaches focus on additional redundancy reduction in JPEG through the use of lossless compression techniques and separate improvement due to their entropy coding methods are not reported.

Although the approaches in [9][10][11] can improve JPEG entropy coding efficiency, they are unsuccessful in taking advantage of either the concept of contexts in exploitation of local information, or the arithmetic codes in fractional code length representation of symbols. In [9], the author has exploited information from the zig-zag scanning position of transform coefficients in the construction of Huffman codes. However, Huffman codes are restricted to have integral code length and therefore are inefficient in coding symbols with high probabilities. In [10][11], although JPEG arithmetic coding is a context-based arithmetic coding scheme, its contexts are inefficient in thorough exploitation of the redundancy in the local information compared to the state-of-the-art schemes such as CABAC [4].

The proposed context-based arithmetic coding scheme aims to improve the entropy coding efficiency of JPEG by combining the advantages of contexts and arithmetic codes. The scheme first uses contexts that exploit local information from the magnitudes of previously coded coefficients in a block and zig-zag scanning positions of transform coefficients in a block to estimate the probabilities of transform coefficients, and then use arithmetic codes to code the coefficients according to their estimated probabilities. As a result, the proposed scheme is able to achieve superior compression efficiency to JPEG two-pass Huffman coding, JPEG arithmetic coding, as well as other existing entropy coding methods for JPEG.

2.2 Video Post-processing

Post-processing in the field of image and video processing generally refers to signal processing techniques that improve the visual quality of digital images and video during display or playback. The characteristic of post-processing is that it does not alter the original data format or data size of the image and video content as the technique is only applied when the content is displayed. Therefore, post-processing techniques can improve the visual quality of digital images and video without introducing extra cost such as increase in data size or change in data format.

Commonly used post-processing techniques include image scaling, image sharpening, contrast enhancement, deinterlacing, and compression artifacts removal techniques such as deblocking, deringing and deblurring. The deblocking technique is considered in this thesis. It is designed to remove blocking artifacts. Blocking artifact is a common type of compression artifact found in low bit-rate compressed images and video (Fig. 1). It appears as rectangular shaped discontinuity across image blocks. It is

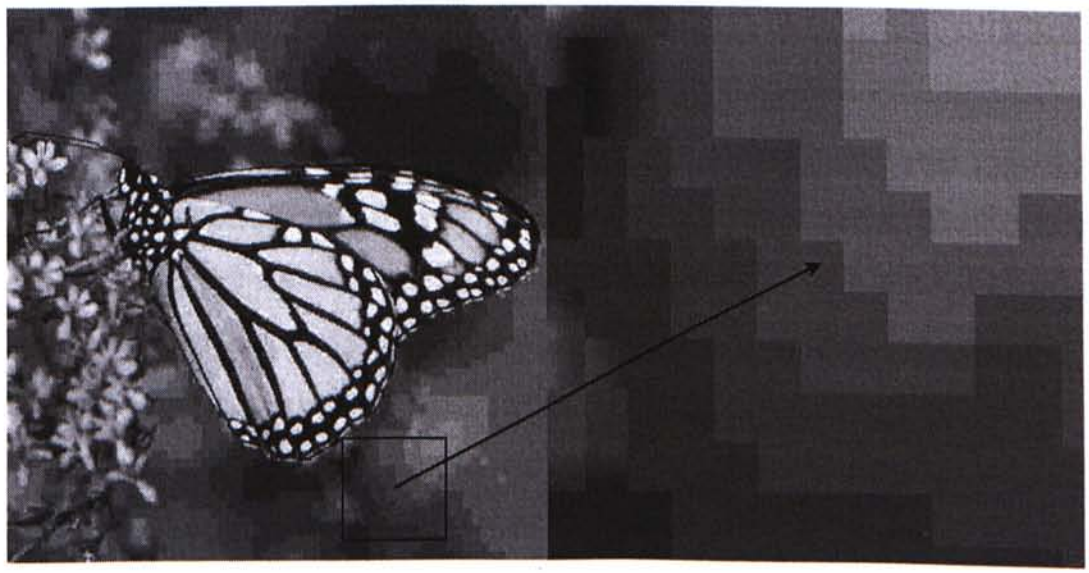


Fig. 1. Blocking artifacts in a low bit-rate JPEG compressed image.

the major source of visual degradation in compressed images and video along with ringing artifact and blurring artifact. Blocking artifact is resulted from independent block transform and quantization, which is a common coding technique found in popular image and video compression standards such as JPEG, MPEG-2 and H.264.

Many approaches to remove blocking artifacts in compressed images were proposed [32]-[43]. In [35], the authors have proposed a DCT domain image deblocking technique, in which the noise in the DCT coefficients is suppressed by low-pass filtering the coefficients in DCT domain. In such method, the filter reconstructs a DCT coefficient in a block by making use of the DCT coefficients at the same frequency from different shifts of the block. The filter is designed to be adaptive to the activity of the block so that the emphasis on artifact reduction and detail preservation can be adjusted. In particular, the filter is adaptively weighted based on the block activity which represents the masking ability of the block for artifacts. For blocks with low activity, blocking artifacts are more perceptible and a filter with large window size is used to smooth out the artifacts. For blocks with high activity, blocking artifacts are less perceptible and a filter with small window size and a large central weight is used for smoothing so as to preserve image details. Another DCT domain deblocking technique was also proposed in [36], in which a method for both measurement and reduction of blocking artifacts in compressed images was proposed by the authors. In such method, the blocking artifacts are modeled as two-dimensional step functions across two adjacent image blocks, and a DCT domain method that utilizes the properties of human visual system is used to measure the visibility of the artifacts. Then, based on the visibility measure of the blocking artifacts at each block edge, all the block edges in the image are divided into several categories and a specific filtering method is applied to block edges in each category so as to reduce blocking artifacts and preserve image details efficiently. In [41], the authors have proposed a spatial domain image deblocking technique by

performing averaging on the shifted blocks. In such method, each block in the compressed image is shifted to generate a set of shifted blocks, and the mean-square-error (MSE) between the block and each of the shifted blocks is calculated. Only those shifted blocks which have MSE smaller than a predetermined threshold will be used for filtering. In particular, the optimal threshold that maximizes the peak-signal-to-noise-ratio (PSNR) of the filtered image is trained from an image database with images compressed by JPEG at different JPEG qualities, and it is shown that it can be well approximated by an exponential function of the JPEG quality factor. In the filtering process, all the pixels in the eligible shifted blocks are averaged to form the reconstructed block. Other approaches such as deblocking by iterative projection onto convex sets (POCS) [32][34], overcomplete wavelet representation [33][37], shape-adaptive DCT [39], maximum a posteriori (MAP) estimation [38][40][42] and in-loop filtering [43] were also proposed.

In video communications, raw video signals are often compressed to smaller data size to meet bandwidth or storage requirement. However, the coding process generally introduces compression artifacts to the original video signal. At low bit-rates, these compression artifacts are particularly visible and severely degrade the visual quality of the compressed video. Post-processing techniques can be used to improve the visual quality of the compressed video without incurring extra cost such as change in data format or increase in data size for the video being processed. In this thesis, the application of a DCT domain post-processing technique [30] for removing blocking artifacts in compressed video is studied. In particular, the technique is applied on H.264 compressed video and its performance is investigated. H.264 is the state-of-the-art video coding standard widely used in the compression of digital video ranging from standard to high definition resolution.

3. Context-Based Arithmetic Coding for JPEG

3.1 Introduction

3.1.1 Huffman Coding

3.1.1.1 Introduction

Huffman coding [5] is the most well-known entropy coding method for data compression. The concept is widely employed in multimedia data compression applications because of its simplicity and optimal compression efficiency under the integral code length assumption. For instance, Huffman coding has been used as the entropy coding method for the popular image compression standard JPEG [14][15] and the popular audio compression standard MPEG-1 Audio Layer 3.

3.1.1.2 Concept

Given a message with alphabet of N symbols, the method first scans through the message to obtain the frequency of occurrence for each of the N symbols by simple counting. After that, the probability for each of the N symbols is calculated by taking the ratio between the counts for that symbol and the total number of counts for the N symbols (Exp. 1).

Message: AACAAABABAAADADEBECCDCBBD	
Counts for A: 10	Probability of A: 10/25 = 0.40
Counts for B: 5	Probability of B: 5/25 = 0.20
Counts for C: 4	Probability of C: 4/25 = 0.16
Counts for D: 4	Probability of D: 4/25 = 0.16
Counts for E: 2	Probability of E: 2/25 = 0.08

Exp. 1. A message of 25 letters with an alphabet of 5 symbols (A, B, C, D, E).

Then, a Huffman code tree is constructed to assign binary codewords to each of the N symbols according to their respective probabilities (Fig. 2). The symbols are first sorted into a list in descending order of probability. Then, the code tree is constructed by

iteratively combining the two symbols with the least probabilities in the list to form a composite symbol, in which the probability is the sum of the probabilities of the two symbols. Each composite symbol forms a node in the tree joined by two branches, and each branch is assigned a binary number (1/0). The Huffman code for each symbol is the concatenated binary codeword given by reading from the root of the tree to the branch connected to the symbol (table 1).

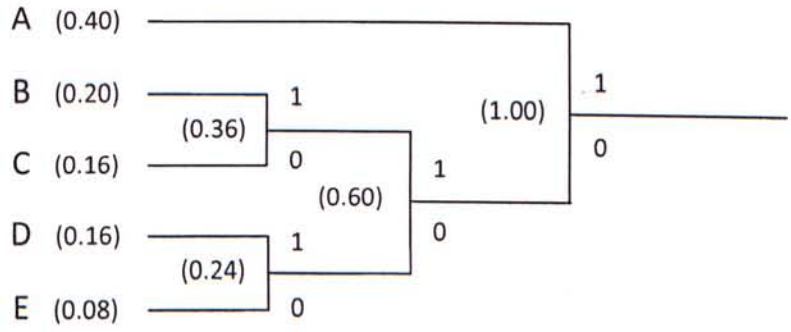


Fig. 2. Huffman code tree for symbols in Exp. 1.

Table 1

Huffman Code Table for Symbols in Exp. 1

Symbol	Huffman Code
A	1
B	011
C	010
D	001
E	000

In such a method, symbols with higher probabilities will be assigned shorter codewords and vice versa, and it has been shown in [5] that codewords under such assignment will be optimal in the sense that the average code length per symbol is minimized so that the total number of bits required to code the message will be minimal.

3.1.1.3 Drawbacks

However, the optimality is only valid if the codewords are restricted to have integral length, which is impractical as the minimum number of bits required to code a symbol is often fractional. The inability of Huffman codes to represent symbols with fractional code length will leave redundancy in the codes. For instance, if the average code length achieved by Huffman coding in Exp. 1 is compared with the entropy bound (table 2), a redundancy of about 3% in the codes is revealed. Such redundancy arises from the mismatch between the code length of Huffman codes in column 3 and the theoretical minimum code length in column 6. The amount of redundancy in table 2 may seem small, but it will become prominent in cases where the probability of one of the symbols is dominant. For instance, if the probability of a symbol in an alphabet is considerably higher than 0.5, then theoretically it should be coded with well less than one bit. However, due to the integral code length limitation of Huffman codes, a least one bit must be used to code the symbol, leaving significant amount of redundancy in the codes.

Table 2

Evaluation on the Redundancy of Huffman Codes with Reference to the Entropy Bound

Symbol	P_i	L_i	Code	P_iL_i	$-log_2P_i$
A	0.40	1	1	0.40	1.322
B	0.20	3	011	0.60	2.322
C	0.16	3	010	0.48	2.644
D	0.16	3	001	0.48	2.644
E	0.08	3	000	0.24	3.644
$Average\ Code\ Length = \sum_{i=1}^{N} P_i L_i = 2.2\ bits\ per\ symbol$					
$\text{Entropy} = -\sum_{i=1}^{N} P_i \log_2 P_i = 2.131\ bits\ per\ symbol$					
$\text{Redundancy (\%)} = \frac{Average\ Code\ Length - Entropy}{Entropy} \times 100 = 3.238\%$					

Where

N denotes the alphabet size.

P_i denotes the probability of a symbol.

L_i denotes the code length of a symbol.

Moreover, Huffman coding also suffers from the absence of exploitation of local information. In Huffman coding, only the global statistical information of the symbols obtained from counting (Exp. 1) is used in the construction of Huffman codes. Local information, such as the letter position, may leave significant amount of information to be exploited. For instance, in Exp. 1, the probability of letter A in the beginning positions of the message is much higher than in the other positions. If such information can be exploited in the construction of Huffman codes, a further reduction in the average code length may be achieved.

Context-based arithmetic coding addresses these two issues by using the concept of contexts to exploit local information for redundancy reduction and arithmetic codes for fractional code length representation of symbols.

3.1.2 Context-Based Arithmetic Coding

3.1.2.1 Introduction

Context-based arithmetic coding is a form of entropy coding scheme that combines the use of contexts and arithmetic codes for the coding of symbols. The scheme first uses contexts to exploit local information within the message to estimate the probabilities of symbols, and then uses arithmetic codes to code the symbols according to their estimated probabilities. Contexts, which exploit redundant local

information, enable a more accurate probability estimation of symbols compared to other schemes such as Huffman coding, which exploits only global statistical information. Moreover, arithmetic codes are able to represent symbols with fractional code length, thus leave less redundancy in the codes and approach the entropy bound closer.

3.1.2.2 Concept

Contexts are coding environments which change according to the local information within the message. In this thesis, the word 'local information' is loosely defined. It represents any information that can be exploited during the course of coding, and therefore different local information can have vastly different nature. For instance, in transform coefficient coding, the local information can be in the form such as the magnitudes of previously coded coefficients, zig-zag scanning positions of transform coefficients, color channel information, and the levels of the DC and AC coefficients of the previously coded block.

Given the local information, such as the position of a symbol, a specific context is selected and the probability of that symbol is estimated independently within that context. In such method, local information can be exploited to enable a more accurate estimation of the probabilities of symbols and hence reduces the number of bits required to code the symbols. Note that the efficiency of the contexts depends entirely on the amount of redundancy contained in the local information being exploited. Local information with significant amount of redundancy is those which, after being exploited, enable an accurate probability estimation of symbols in the corresponding contexts. Therefore, a thoughtful selection on local information for redundancy exploitation is of ultimate importance in the design of contexts.

To illustrate the concept of contexts, the message in Exp. 1 is re-considered. By observation, it is found that the probability of symbol A in the beginning positions of the message is much higher than in the other positions. Such characteristic of the message indicates significant amount of redundancy in the position information. Assume the characteristic is observed in advance, a simple context design that exploits information from the letter position can be used to enable a more accurate probability estimation of symbol A. For instance, the letter positions can be classified into 'beginning positions' and 'latter positions', where the former should be designed such that the probability of symbol A in in the former are much higher than in the latter. The letters in the 'beginning positions' and 'latter positions' are treated as they are within different coding environments, represented by Context A and Context B respectively, and their probabilities are estimated independently within their respective contexts (Exp. 2).

Message: AACAAABABAAADADEBECCDCBBD

Context A

Message	A	A	C	A	A	A	B	A	B	A	A	A
Beginning Positions	1	2	3	4	5	6	7	8	9	10	11	12
Counts for A: 9					Probability of A: 9/12 = 0.750							
Counts for B: 2					Probability of B: 2/12 = 0.167							
Counts for C: 1					Probability of C: 1/12 = 0.083							
Counts for D: 0					Probability of D: 0/12 = 0.000							
Counts for E: 0					Probability of E: 0/12 = 0.000							

Context B

Message	D	A	D	E	B	E	C	C	D	C	B	B	D
Latter Positions	13	14	15	16	17	18	19	20	21	22	23	24	25
Counts for A: 1						Probability of A: 1/13 = 0.077							
Counts for B: 3						Probability of B: 3/13 = 0.231							
Counts for C: 3						Probability of C: 3/13 = 0.231							
Counts for D: 4						Probability of D: 4/13 = 0.308							
Counts for E: 2						Probability of E: 2/13 = 0.154							

Exp. 2. A simple context design that exploits local information from letter position. Position 1-12 are 'beginning positions'. Position 13-25 are 'latter positions'.

Calculations from Exp. 2 show that through the exploitation of redundant information in the letter position, the context design in Exp. 2 is able is estimate the probability of symbol A more accurately as indicated by the significantly higher probability of symbol A in Context A compared to the probability of symbol A obtained from global statistics in Exp. 1. As known from the calculation of entropy, the more dominant is the probability of a symbol in a probability distribution, the lower the entropy. In the extreme case, where the probability of a symbol is unity in a probability distribution, the entropy will be zero. In the worst case, where the probabilities of all symbols in a probability distribution are equally probable, the entropy will be maximized.

In order to demonstrate that the use of the context design in Exp. 2 indeed reduces the number of bits required to code the message, the entropy under Context A and Context B are calculated (table 3 and 4), and from there the entropy under the context design in Exp. 2 is derived. The entropy is then compared with the entropy obtained without using contexts as given earlier in table 2, where the reduction in entropy due to the use of the context design in Exp. 2 can be evaluated (table 5).

Table 3

Entropy under Context A in Exp. 2

Symbol	P_i	$-log_2 P_i$
A	0.750	0.415
B	0.167	2.582
C	0.083	3.591
D	0.000	N/A
E	0.000	N/A
Entropy $(E_A) = -\sum_{i=1}^{N} P_i \, log_2 P_i = 1.040 \; bits \; per \; symbol$		

Table 4

Entropy under Context B in Exp. 2

Symbol	P_i	$-log_2 P_i$
A	0.077	3.699
B	0.231	2.114
C	0.231	2.114
D	0.308	1.699
E	0.154	2.699
Entropy $(E_B) = -\sum_{i=1}^{N} P_i \, log_2 P_i = 2.200 \; bits \; per \; symbol$		

Table 5

Reduction in Entropy by the Context Design in Exp. 2

Entropy $(E_C) = P_A E_A + P_B E_B = \frac{12}{25}(1.040) + \frac{13}{25}(2.200) = 1.643 \; bits \; per \; symbol$
Entropy (E) = 2.131 bits per symbol (from table 2)
Reduction in Entropy (%) $= \frac{E - E_C}{E} \times 100 = 22.90\%$

Remark: The overhead required to signal the probability information in the contexts is not included in the calculation.

Where

E_C denotes the entropy under the context design in Exp. 2.

E denotes the entropy obtained without using contexts.

P_A denotes the probability of letters being in Context A.

P_B denotes the probability of letters being in Context B.

Calculations from table 5 show that with the use of the context design in Exp. 2, the entropy is reduced significantly by 22.90%. Although the data in Exp. 1 and 2 are synthetic, the results in table 5 are still able to demonstrate the potential use of the context concept in improving the data compression efficiency. In particular, Exp. 2 has shown that if significant redundant information exists in the message and such information is known in advance, then contexts that efficiently exploit such information can be designed to improve the data compression efficiency considerably. In transform coefficient coding, it is well-known that the magnitude and position of transform coefficients in a block contain significant redundant information. For instance, magnitudes of coefficients in a block generally demonstrate a decreasing trend along the zig-zag scan and magnitudes of coefficients at different zig-zag scanning positions of a block are generally significantly different. In such case, an efficient context design that can thoroughly exploit such redundant information may be incorporated into a transform coefficient coding scheme to improve its compression efficiency.

Note that the context design for the message in Exp. 2 is merely one of the many possible designs. For instance, contexts that are based on the previously coded symbol may be used to exploit information from the previously coded symbol to estimate the probability of the current symbol more accurately. In such design, the total number of contexts will be five as the previously coded symbol can be any one of the five symbols in the alphabet. The context for the current symbol will be selected according to the

previously coded symbol. For instance, if the previously coded symbol is A, then the context for the current symbol will be Context 1. If the previously coded symbol is B. then the context for the current symbol will be Context 2 and so on (Exp. 3). In such method, each symbol will be in either one of the five contexts depending on its previous symbol, and the probabilities of the symbols are estimated independently within their corresponding contexts.

Message: AACAAABABAAADADEBECCDCBBD

Message	A	A	C	A	A	A	B	A	B	A	A	A
Context Number	1	1	1	3	1	1	1	2	1	2	1	1

Message	D	A	D	E	B	E	C	C	D	C	B	B	D
Context Number	1	4	1	4	5	2	5	3	3	4	3	2	2

Remark: Context 1 is assumed to be the context for the first symbol in the message.

Context 1

Counts for A: 6	Probability of A: 6/11 = 0.545
Counts for B: 2	Probability of B: 2/11 = 0.182
Counts for C: 1	Probability of C: 1/11 = 0.091
Counts for D: 2	Probability of D: 2/11 = 0.182
Counts for E: 0	Probability of E: 0/11 = 0.000
Entropy $(E_1) = 1.687$ *bits per symbol*	

Context 2

Counts for A: 2	Probability of A: 2/5 = 0.400
Counts for B: 1	Probability of B: 1/5 = 0.200
Counts for C: 0	Probability of C: 0/5 = 0.000
Counts for D: 1	Probability of D: 1/5 = 0.200
Counts for E: 1	Probability of E: 1/5 = 0.200
Entropy $(E_2) = 1.922$ *bits per symbol*	

Context 3

Counts for A: 1	Probability of A: 1/4 = 0.250
Counts for B: 1	Probability of B: 1/4 = 0.250
Counts for C: 1	Probability of C: 1/4 = 0.250
Counts for D: 1	Probability of D: 1/4 = 0.250
Counts for E: 0	Probability of E: 0/4 = 0.000
Entropy $(E_3) = 2.000$ *bits per symbol*	

Context 4

Counts for A: 1	Probability of A: 1/3 = 0.333
Counts for B: 0	Probability of B: 0/3 = 0.000
Counts for C: 1	Probability of C: 1/3 = 0.333
Counts for D: 0	Probability of D: 0/3 = 0.000
Counts for E: 1	Probability of E: 1/3 = 0.333
Entropy $(E_4) = 1.585$ *bits per symbol*	

Context 5

Counts for A: 0	Probability of A: 0/2 = 0.000
Counts for B: 1	Probability of B: 1/2 = 0.500
Counts for C: 1	Probability of C: 1/2 = 0.500
Counts for D: 0	Probability of D: 0/2 = 0.000
Counts for E: 0	Probability of E: 0/2 = 0.000
Entropy $(E_5) = 1.000$ *bits per symbol*	

Exp. 3. An alternative context design for the message in Exp. 2. The contexts are designed to exploit information from the previously coded symbol.

Table 6

Reduction in Entropy by the Context Design in Exp. 3

$\text{Entropy } (E_C) = \sum_{i=1}^{5} P_i E_i = \frac{11}{25}(1.687) + \frac{5}{25}(1.922) + \frac{4}{25}(2.000) + \frac{3}{25}(1.585) + \frac{2}{25}(1.000)$ $= 1.717\ bits\ per\ symbol$ Entropy (E) = 2.131 bits per symbol (from table 2)
$\text{Reduction in Entropy (\%)} = \frac{E - E_C}{E} \times 100 = 19.43\%$

Remark: The overhead required to signal the probability information in the contexts is not included in the calculation.

Where

E_C denotes the entropy under the context design in Exp. 3.

E denotes the entropy obtained without using contexts.

P_i denotes the probability of letters being in Context i.

It should be noted that the total number of contexts in a context design should be controlled, as in practice additional bits in the form of transmission overhead are required to signal the probability information contained in each context to the decoder, usually in the form of code tables as in Huffman coding, for successful decoding. If the number of contexts is unrestricted, the total overhead required may exceed the reduction in number of bits for coding the message as brought by the context design.

In a context-based arithmetic coding scheme, the context design is incorporated into the front-end of the arithmetic coding procedure to enable a more accurate probability estimation of symbols so as to improve the compression efficiency of the scheme (Fig. 3). After the probabilities of symbols are estimated by the contexts,

arithmetic coding is used to code each of the symbols according to its estimated probability in its respective context.

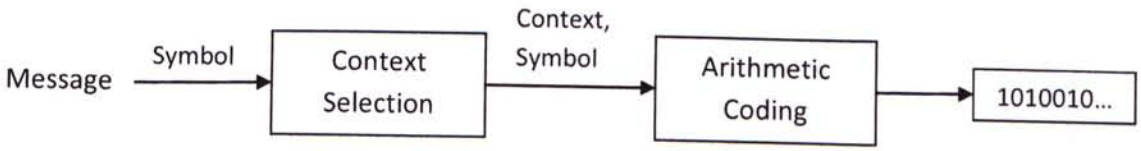


Fig. 3. A simple context-based arithmetic coding scheme. The 'Context Selection' procedure is widely referred as 'Context Modeling' in literatures. The context that the symbol is belonged contains the probability information for that symbol.

Arithmetic coding [6][7][8] is based on the concept of recursive subdivision of a real number line of unity length [0 1]. Given a message with alphabet of N symbols, the real number line is first divided into N partitions, with each symbol occupying one of the partitions. The length of each partition is directly proportional to the probability of the occupying symbol (Fig. 4). In particular, the length of a partition is given by:

$$L_i = L \times P_i \quad for\ i = 1,2, \dots, N \quad \dots\dots\dots (1)$$

Where

L denotes the length of the current line.

L_i denotes the length of Partition i within the current line.

P_i denotes the probability of the symbol occupying Partition i.

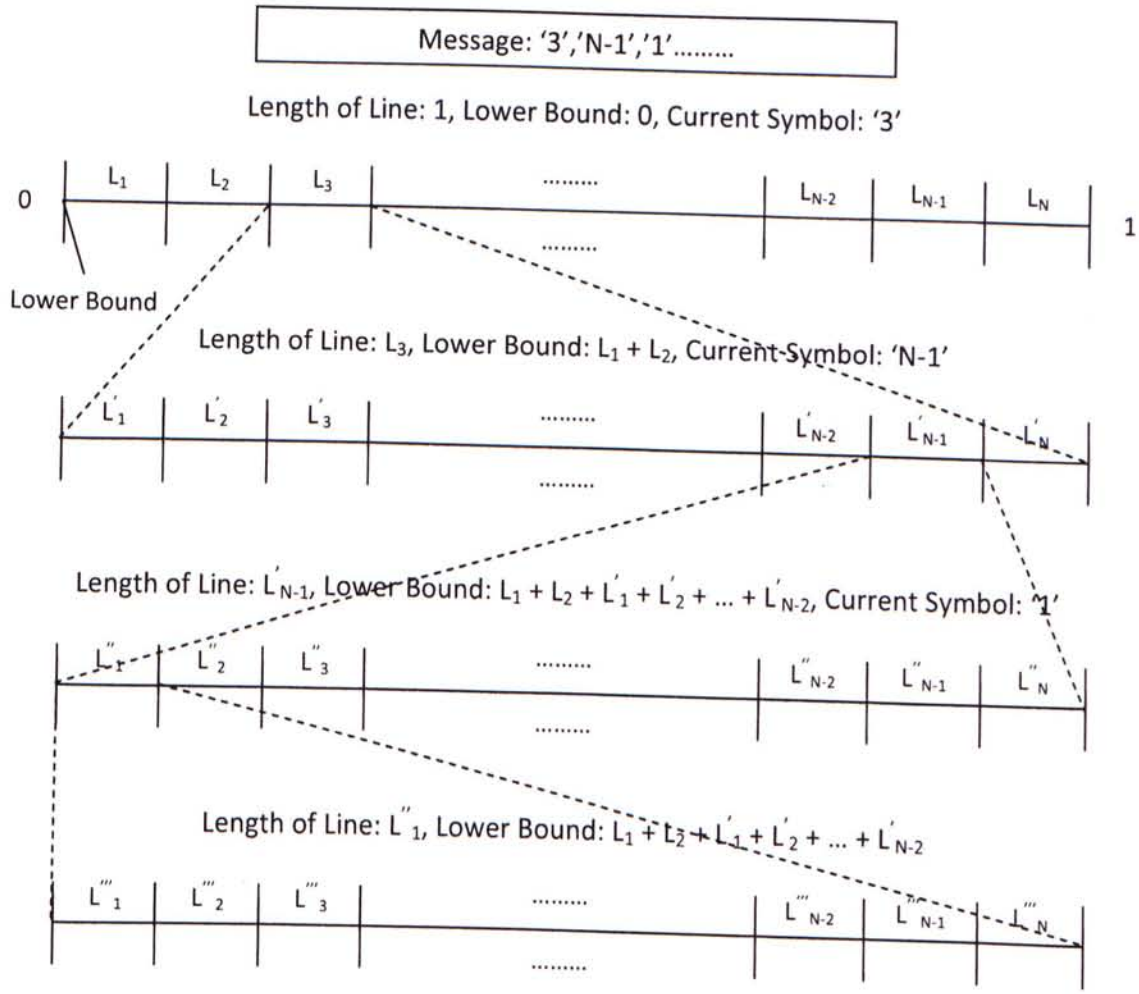


Fig. 4. Recursive subdivision of a real number line as in arithmetic coding.

Then, to code a symbol in the message, the partition that corresponds to the current symbol is selected to become a new line, where the length of the new line is the length of the partition, and the lower bound of the new line with respect to the original line is recorded. Then, the new line is divided into N partitions again according to (1) and the next symbol in the message is coded. Such a line subdivision process is repeated until all symbols in the message is coded. As more symbols in the message are coded, the length of the line becomes smaller and the number of bits required to precisely

specifying the lower bound of the line increases. More probable symbols introduce less reduction to the length of the line in subdivision than less probable symbols. Hence, coding more probable symbols require fewer extra bits to specify the lower bound of the line. The codeword that represents the message is the bit pattern that specifies the lower bound of the final line. As the entire message in arithmetic coding is represented by a single integral length codeword, as oppose to one integral length codeword for one symbol in the message as in Huffman coding, the codeword generated by arithmetic coding is able to represent each individual symbol in the message with fractional number of bits.

Note that arithmetic coding is preferred over Huffman coding because an efficient context design will often result in symbols with dominant probabilities in the contexts, and such symbols, which should be coded with well less than one bit, can be coded efficiently using arithmetic codes with fractional code length representation. In such case, arithmetic codes has distinct advantage over Huffman codes as they leave smaller amount of redundancy in the codes and therefore approach the entropy bound closer.

3.2 Proposed Method

3.2.1 Introduction

In transform coefficient coding, it is well known that intra block local information, such as the magnitudes of previously coded coefficients in a block and zig-zag scanning positions of transform coefficients in a block, contain significant redundancy. If such redundancy can be exploited efficiently by a context design, significant improvement in compression efficiency of transform coefficients may be obtained.

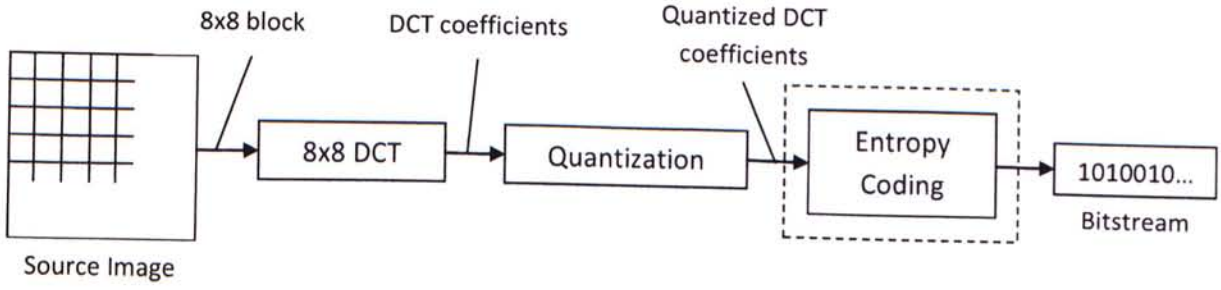


Fig. 5. A simplified diagram of JPEG. The part enclosed by dotted line indicates the scope of the proposed method.

In this thesis, a new context-based arithmetic coding scheme for enhancing the compression efficiency of JPEG entropy coding is developed (Fig. 5). The proposed scheme uses a context design that efficiently exploit the redundancy exists in quantized DCT coefficients. In particular, the scheme first uses a modified H.264 significance map coding technique, which exploits redundant information from the zig-zag scanning positions of transform coefficients, to code the non-zero coefficient flags and EOB decisions of transform coefficients for a block. Then, contexts that efficiently exploit redundant information from both the magnitudes of previously coded coefficients in the block and the zig-zag scanning positions of transform coefficients in the block are used to code the remaining 'LEVEL' symbols of transform coefficients. Moreover, the context design also exploits information from color channel by using three independent sets of contexts to code the coefficients from the three color channels separately, with one set of contexts for one channel.

3.2.2 Redundancy in Quantized DCT Coefficients

3.2.2.1 Zig-Zag Scanning Position

JPEG is a popular image compression standard that employs the traditional block-based DCT coding technique for the compression of image data [14][15]. In JPEG, the source image is partitioned into non-overlapping 8x8 blocks, and each block is transformed by the DCT into a block of DCT coefficients. Then, the DCT coefficients in the block are quantized to form the quantized DCT coefficients, which are then entropy coded to become a bitstream (Fig. 5).

It is well known that, in a block of quantized DCT coefficients, the zig-zag scanning position of transform coefficient contains significant redundancy. For instance, the magnitudes of non-zero transform coefficients in a block are strongly dependent on the zig-zag scanning positions, or equivalently the magnitudes of non-zero transform coefficients at different zig-zag scanning positions of a block are significantly different.

Fig. 6 shows the probability distributions of the magnitudes of non-zero transform coefficients at different zig-zag scanning positions for the image Lena (512x512) at 90% JPEG quality obtained by simple counting. The results show that the probability distributions at some of the zig-zag scanning positions are much more skewed than the one obtained from global statistics (Fig. 7), which indicates that the probabilities of the magnitudes of non-zero transform coefficients at those zig-zag scanning positions can be estimated more accurately, provided that the estimation is performed independently at each zig-zag scanning position. Such observation demonstrates that the information from the zig-zag scanning position of transform coefficients contains significant redundancy. Such redundancy may be exploited efficiently by a context design for improved compression efficiency on the transform coefficients. In particular, the zig-zag scanning position in a block may be used as the local information for context selection.

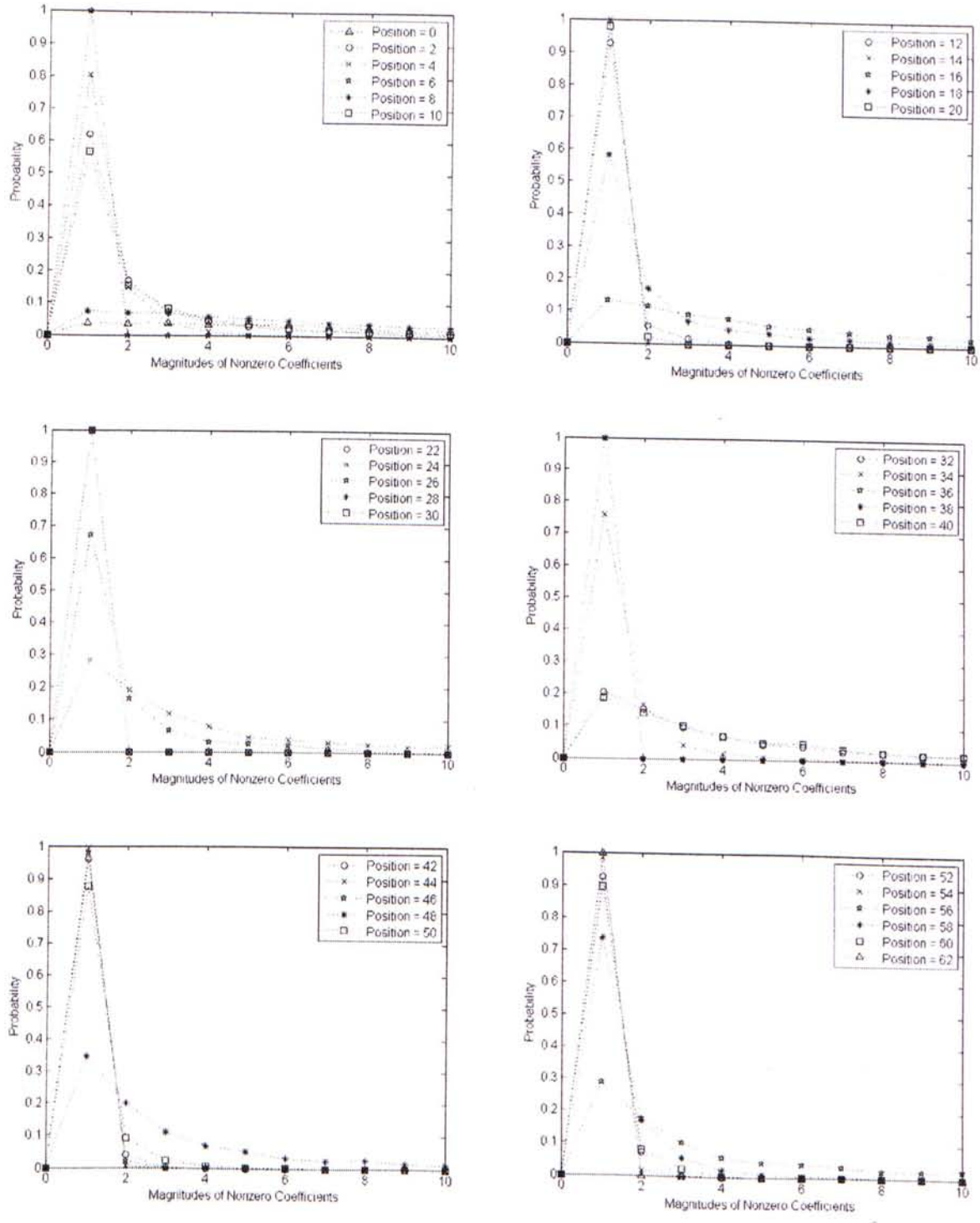


Fig. 6. Probability distributions of the magnitudes of non-zero transform coefficients at different zig-zag scanning positions for the image Lena (512x512) at 90% JPEG quality.

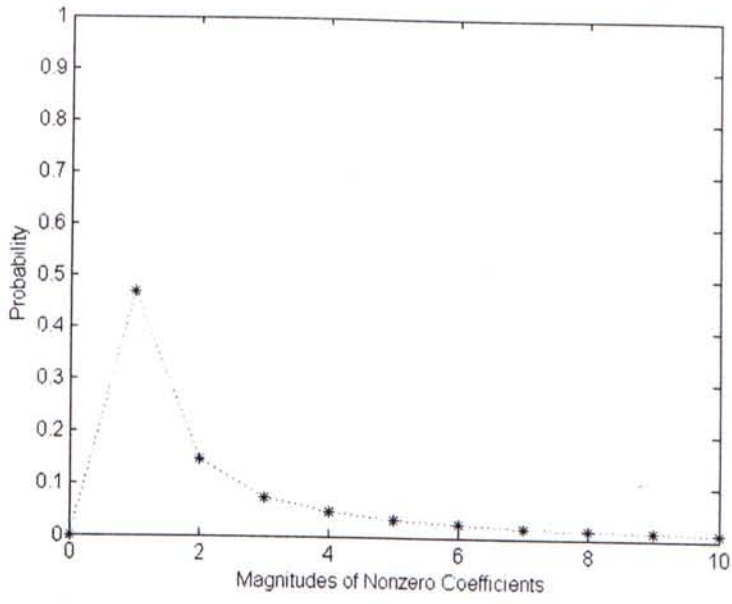
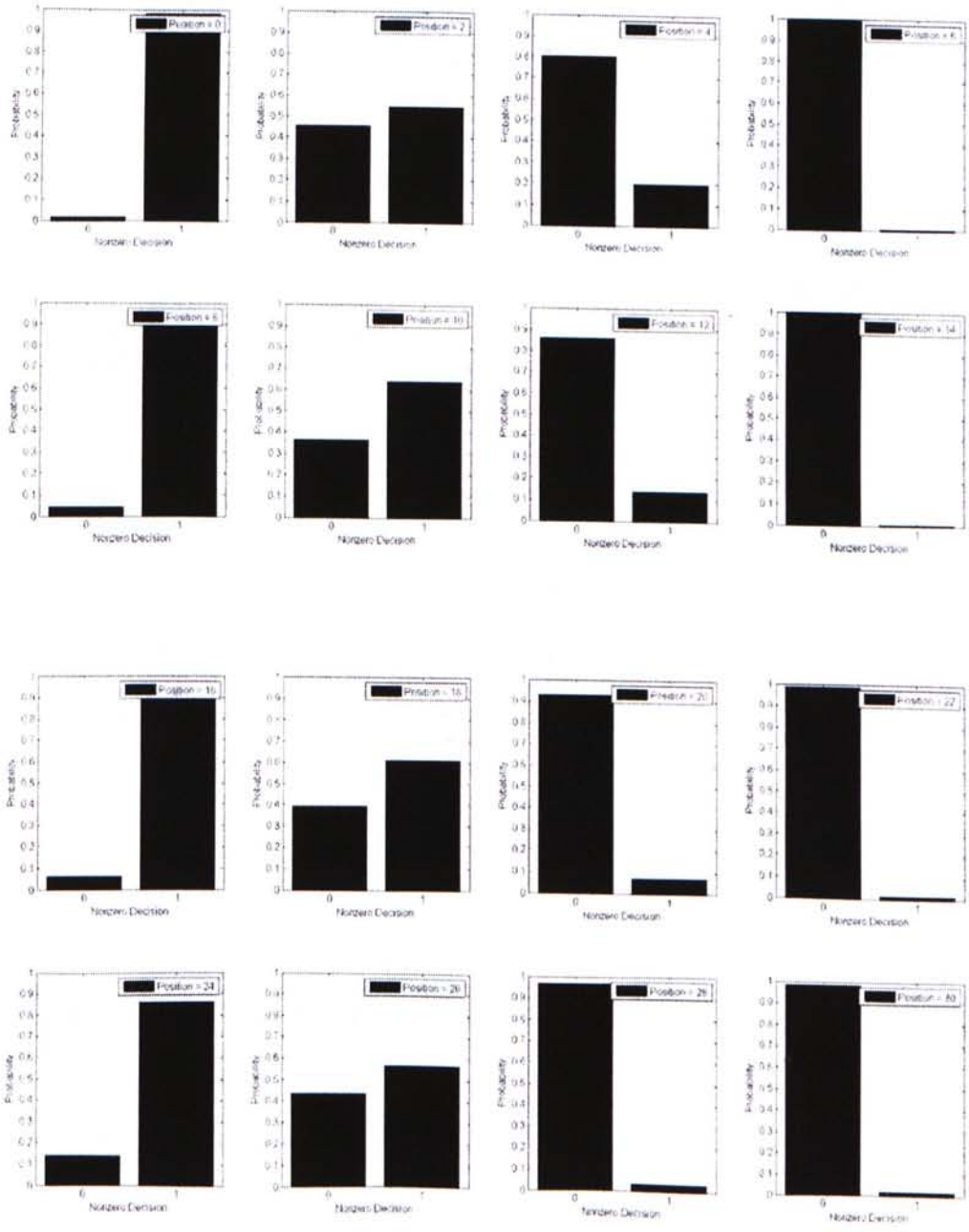


Fig. 7. Probability distribution of the magnitudes of non-zero transform coefficients obtained from global statistics for the image Lena (512x512) at 90% JPEG quality.

In such context design, each zig-zag scanning position will correspond to a context such that there will be 64 contexts in total, and the probability of the magnitude of each non-zero transform coefficient in a block will be estimated independently within its own context indicated by its zig-zag scanning position in the block. Results in Fig. 6 have shown that such estimation method enables the probabilities of the magnitudes of non-zero transform coefficients to be estimated more accurately.

Apart from that, it is also observed that the occurrence of EOB and non-zero transform coefficient in a block also depends strongly on the zig-zag scanning positions of transform coefficients in the block. Fig. 8 shows the probability distributions of the non-zero decisions of transform coefficients at different zig-zag scanning positions for Lena at 90% JPEG quality, where the symbol '0' indicates the magnitude of the transform coefficient is zero and the symbol '1' indicates the magnitude of the transform coefficient is non-zero. The results show that the probabilities of the non-zero decisions

of transform coefficients can be estimated more accurately at most zig-zag scanning positions compared to the estimation obtained from global statistics (Fig. 9).



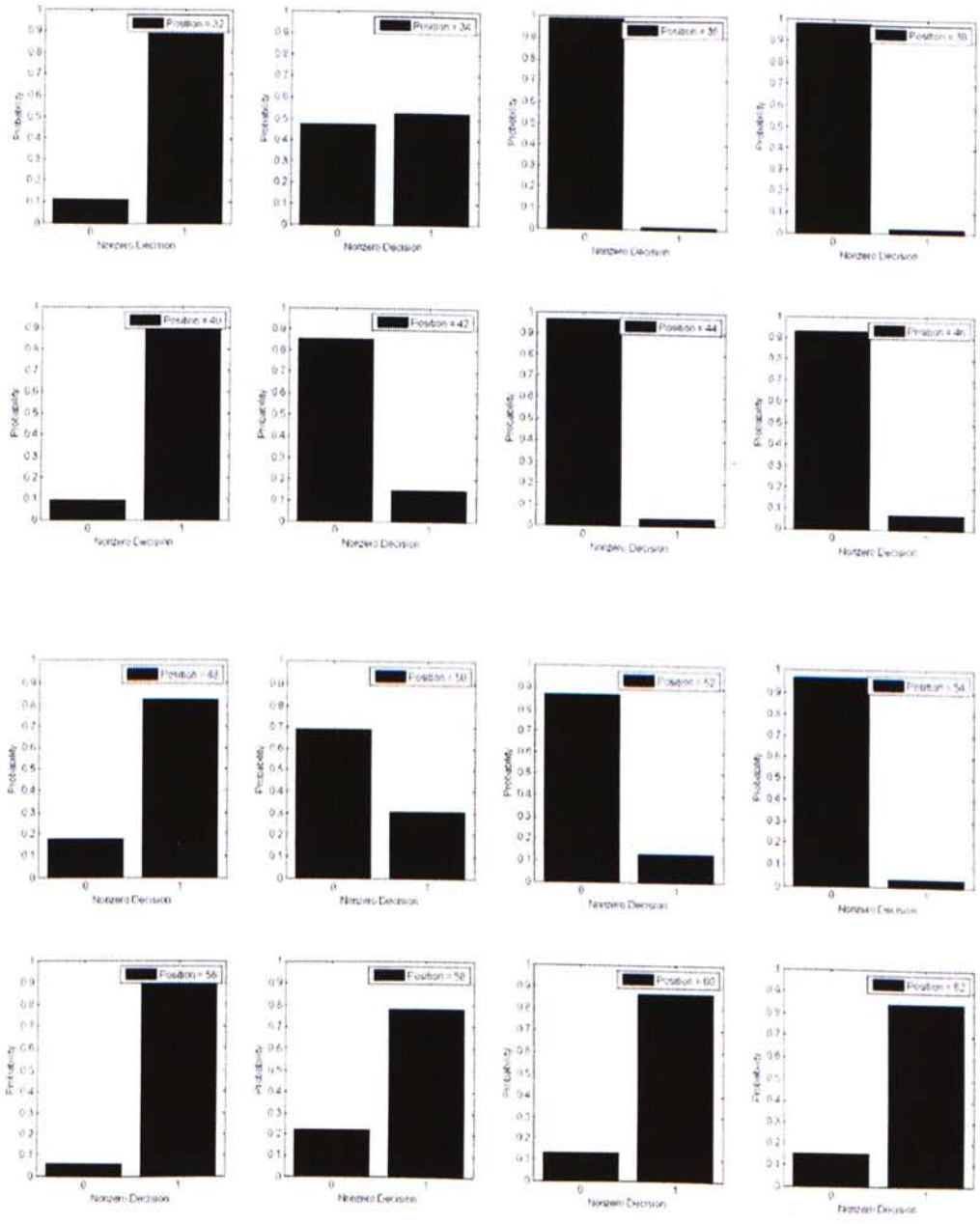


Fig. 8. Probability distributions of the non-zero decisions of transform coefficients at different zig-zag scanning positions for the image Lena (512x512) at 90% JPEG quality.

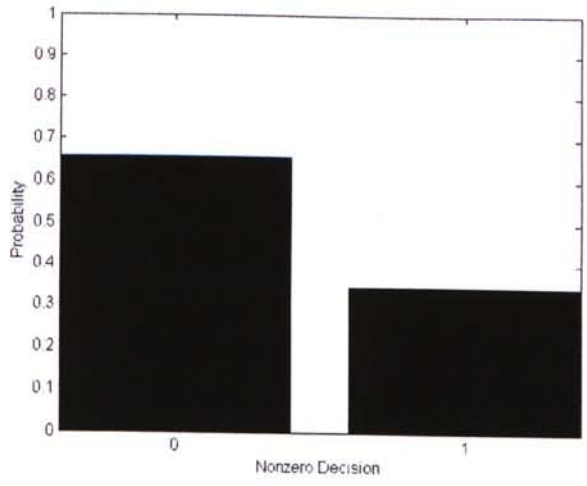
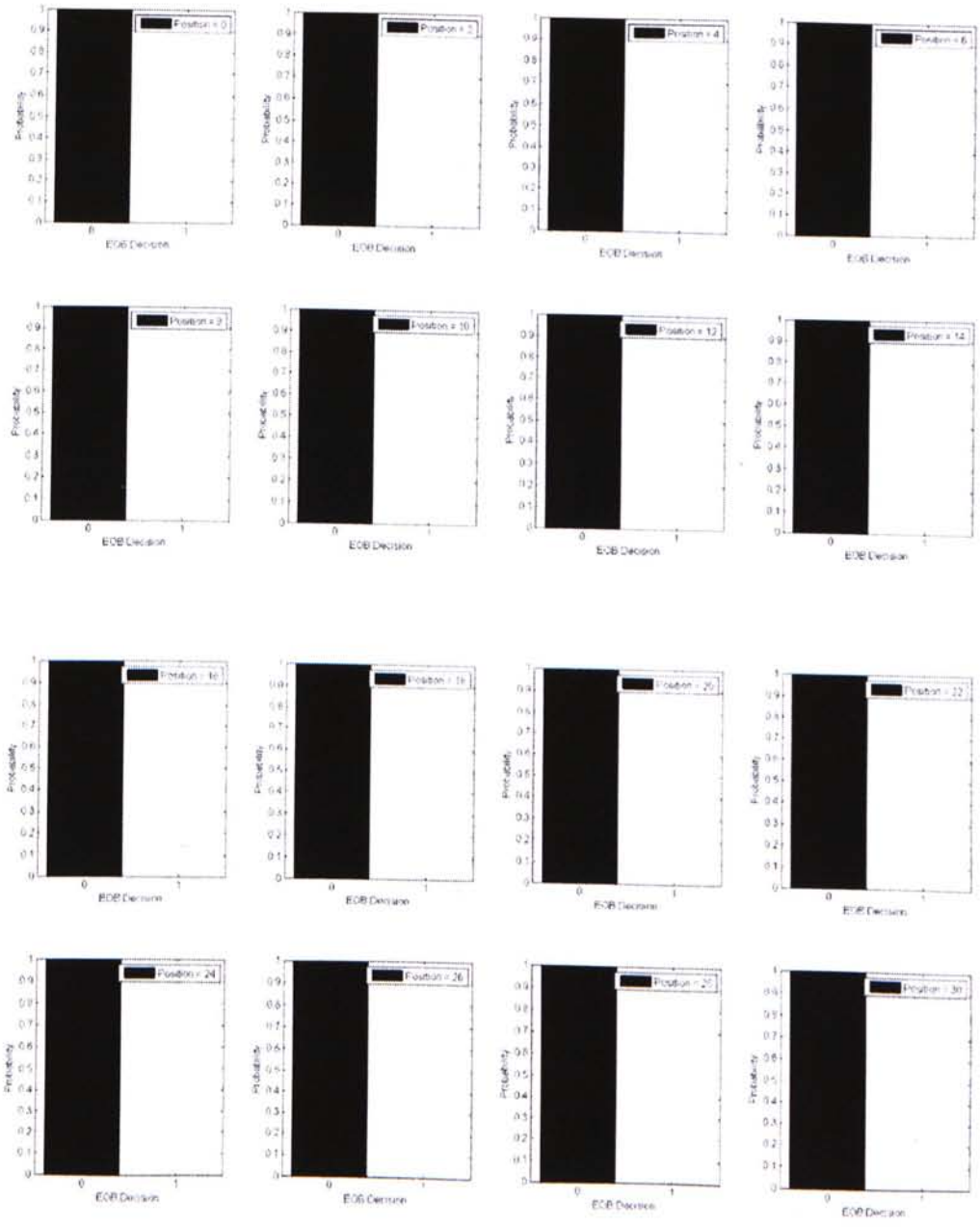


Fig. 9. Probability distribution of the non-zero decisions of transform coefficients obtained from global statistics for the image Lena (512x512) at 90% JPEG quality.

Similarly, Fig. 10 shows the probability distributions of the EOB decisions of non-zero transform coefficients at different zig-zag scanning positions for Lena at 90% JPEG quality, where the symbol '1' indicates the non-zero transform coefficient is the last coefficient in a block and the symbol '0' indicates that it is not. The results show that the probabilities of the EOB decisions of non-zero transform coefficients can be estimated slightly more accurately at most zig-zag scanning positions compared to the estimation obtained from global statistics (Fig. 11).



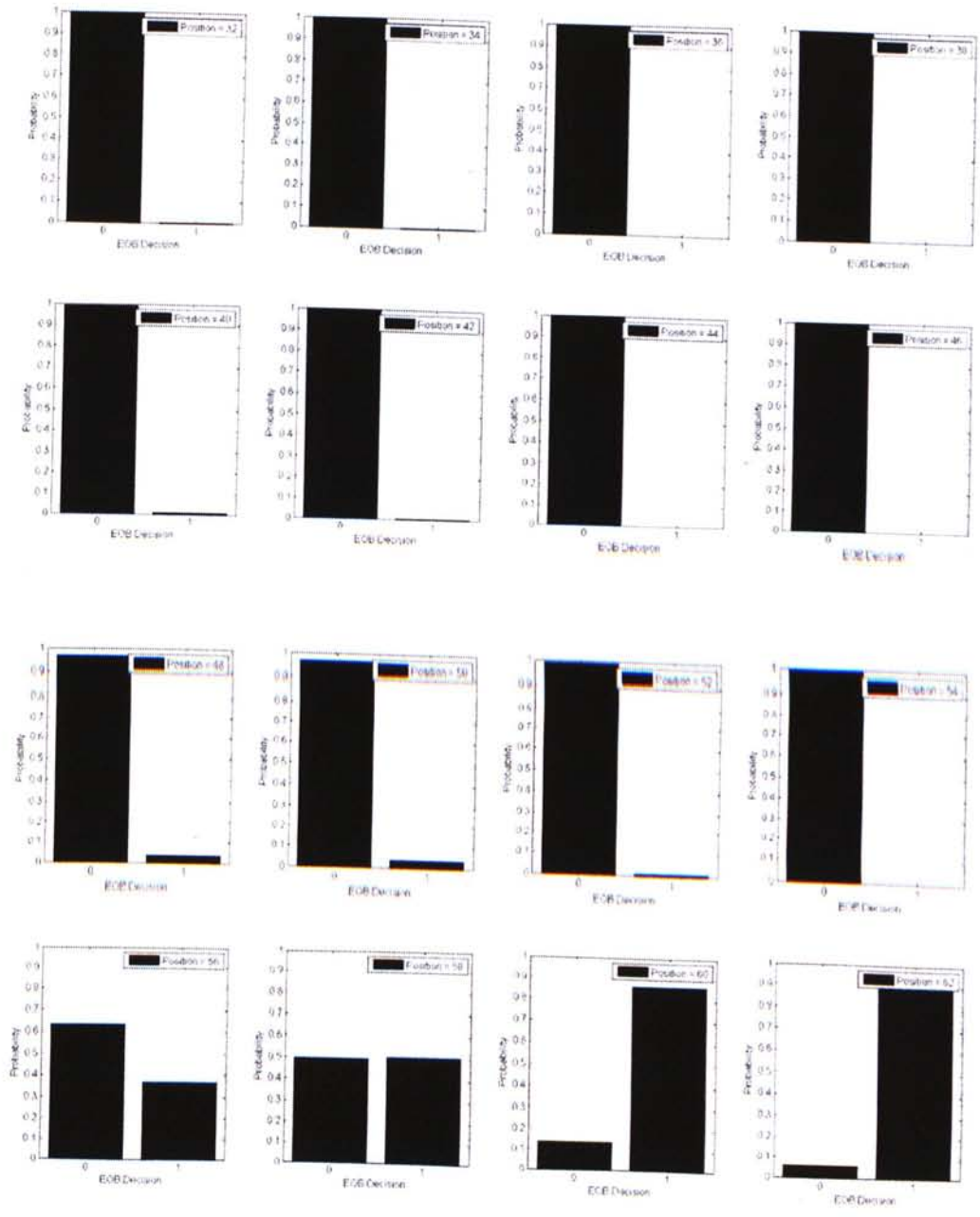


Fig. 10. Probability distributions of the EOB decisions of non-zero transform coefficients at different zig-zag scanning positions for the image Lena (512x512) at 90% JPEG quality.

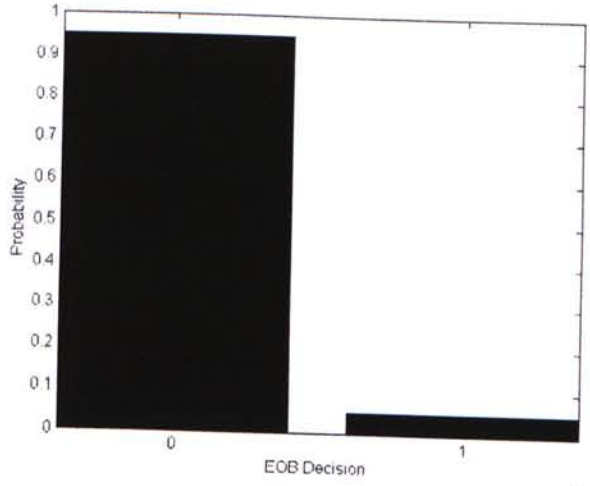


Fig. 11. Probability distribution of the EOB decisions of non-zero transform coefficients obtained from global statistics for the image Lena (512x512) at 90% JPEG quality.

The results from Fig. 8 and 10 indicate that redundant information exists in the zig-zag scanning position of transform coefficients. Such redundancy has been exploited by the significance map coding technique of CABAC [4], in which the zig-zag scanning positions of transform coefficients are used as local information for selecting contexts to estimate the probabilities of the non-zero coefficient flags and EOB decisions of transform coefficients. In such context design, each zig-zag scanning position corresponds to a context so that there are 64 contexts in total. The probabilities of the non-zero coefficient flags and EOB decisions of transform coefficients are estimated independently within their respective contexts indicated by the zig-zag scanning positions of the transform coefficients they correspond to.

3.2.2.2 Magnitudes of Previously Coded Coefficients

It is well known that, in a block of quantized DCT coefficients, the magnitudes of transform coefficients demonstrate a decreasing trend along the zig-zag scan. Such observation indicates that redundant information exists in the magnitudes of transform coefficients. For instance, if the transform coefficients in a block are traversed and coded in reverse zig-zag scan order (Exp. 4), the magnitude of each non-zero transform coefficient in the block will demonstrate strong dependency on the maximum magnitude of the previously coded non-zero transform coefficients in the same block (Exp. 5).

Zig-Zag Scanning Position	1	2	3	4	5	6	7	8
Transform Coefficients (Zig-Zag Scan Order)	9	-2	3	0	-2	0	0	-1
Transform Coefficients (Reverse Zig-Zag Scan Order)	-1	0	0	-2	0	3	-2	9

Exp. 4. Transform coefficients in reverse zig-zag scan order. Note that '-1' is the last coefficient in the block.

Non-zero Transform Coefficients (Reverse Zig-Zag Scan Order)	-1	-2	3	-2	9
Lmax	0	1	2	3	3

Exp. 5. Maximum magnitude of the previously coded non-zero transform coefficients.

Where

L_{max} denotes the maximum magnitude of the previously coded non-zero transform coefficients for the current non-zero transform coefficient.

L_{max} is assumed to be zero for the first non-zero transform coefficient.

Fig. 12 shows the probability distributions of the magnitudes of non-zero transform coefficients at different L_{max} for Lena at 90% JPEG quality. The results show that the probabilities of the magnitudes of non-zero transform coefficients are estimated more accurately at some values of L_{max} compared to the estimation obtained from glob-

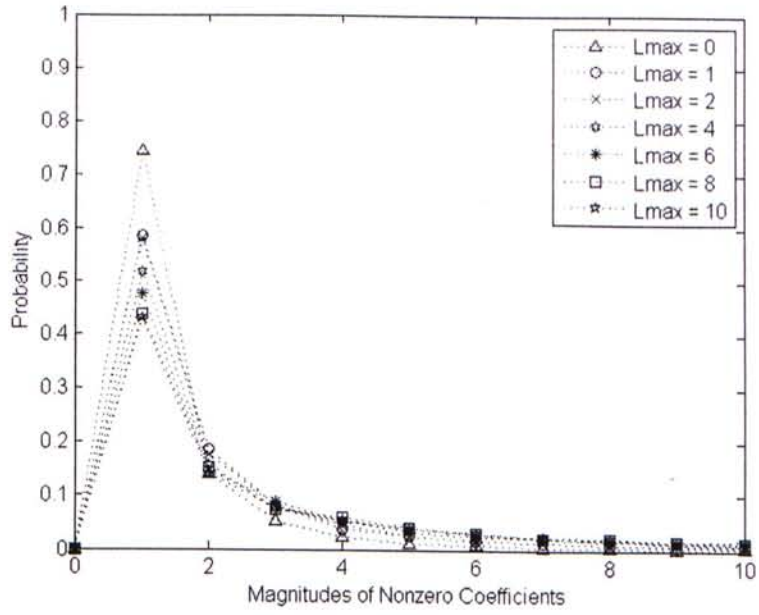


Fig. 12. Probability distributions of the magnitudes of non-zero transform coefficients at different Lmax for the image Lena (512x512) at 90% JPEG quality.

al statistics (Fig. 7). Such observation indicates that redundant information exists in the magnitudes of previously coded coefficients in a block, provided that the block is traversed and coded in reverse zig-zag scan order. Such redundancy has been exploited by the context design in the context-based arithmetic coding scheme of AVS [16] for improving the compression efficiency on the magnitudes of non-zero transform coefficients. In such context design, Lmax is used as the local information for selecting contexts to estimate the probabilities of the magnitudes of non-zero transform coefficients in a block. However, instead of mapping each Lmax to a context, a range of Lmax is mapped to one context so that there are only 5 contexts in total. Such mapping solves the issue of a large number of contexts due to the broad range of values that Lmax can take. Note that mapping a range of Lmax to one context is equivalent to merging the probability distributions of the magnitudes of non-zero transform coefficients at different Lmax within the range.

3.2.3 Proposed Scheme

3.2.3.1 Overview

The basic coding unit in the proposed scheme is an 8x8 block of quantized DCT coefficients, where the information contained in the coefficients are represented by three symbols, which are the non-zero coefficient flag, EOB decision, and 'LEVEL'.

The non-zero coefficient flag is a binary symbol (1 or 0) which indicates whether the magnitude of a transform coefficient in the block is zero or non-zero. Transform coefficients which are indicated by the non-zero coefficient flags to have zero magnitude are not required to be coded in subsequent coding process. The EOB decision is a binary symbol (1 or 0) which indicates whether a non-zero transform coefficient in the block is the last coefficient in the block. 'LEVEL' is a symbol which indicates the level of a non-zero transform coefficient in the block, which in turn consists of the sign (1 or 0) and magnitude of the non-zero transform coefficient. In section 3.2.2, it is observed that the probabilities of non-zero coefficient flags, EOB decisions and the magnitudes of non-zero transform coefficients as in 'LEVEL' can be estimated efficiently by exploiting the redundancy that exists in quantized DCT coefficients (Fig. 6, 8, 10, 12). Hence, to maximize the compression efficiency on quantized DCT coefficients, the contexts in the proposed scheme are designed so as to exploit such redundancy efficiently.

The proposed scheme consists of two stages. In the first stage, a modified H.264 significance map coding technique, in which contexts are designed to exploit redundant information in the zig-zag scanning positions of transform coefficients (Fig. 8 and 10), are used to code the non-zero coefficient flags and EOB decisions of transform coefficients in the block. In the second stage, contexts that exploit redundant information from both the zig-zag scanning positions of transform coefficients (Fig. 6) and magnitudes of previously coded coefficients (Fig. 12) are used to code the magnitudes of

non-zero transform coefficients as in the 'LEVEL' symbols. The sign in 'LEVEL' is coded separately with a fixed probability of 0.5.

3.2.3.2 Preparation of Coding

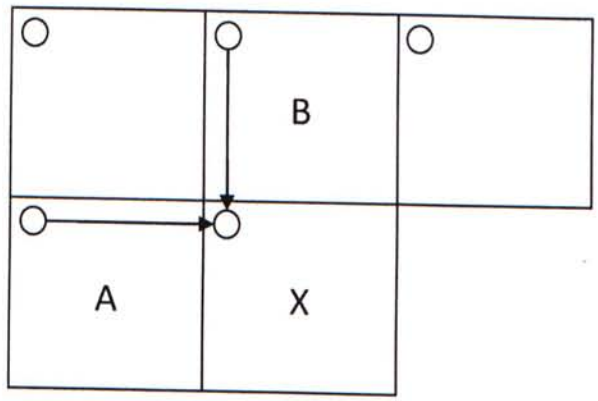


Fig. 13. The DC coefficient prediction technique used in the proposed scheme. The block marked 'X' is the current block. The circular shape denotes the DC coefficient in a block.

Before the coding process begins, the quantized DCT coefficients in a block are rearranged in zig-zag scan order. Moreover, the quantized DC coefficient in the block is predicted by the quantized DC coefficients from previously coded blocks using a DC coefficient prediction technique. However, instead of using the technique from JPEG, the DC coefficient prediction technique in Fig. 13 is used, which is found to offer improved compression efficiency to the proposed scheme on quantized DCT coefficients. The technique operates on a simple mechanism according to four rules:

1) If both Block A and Block B are available, then the DC coefficient of the current block, Block X, is predicted from the average of the DC coefficients of Block A and Block B.

2) Else if Block A is available, the DC coefficient of Block X is predicted from the DC coefficient of Block A.

3) Else the DC coefficient of Block X is predicted from the DC coefficient of Block B.

4) The DC coefficient of the first block in an image is not predicted.

3.2.3.3 Coding of Non-zero Coefficient Flags and EOB Decisions

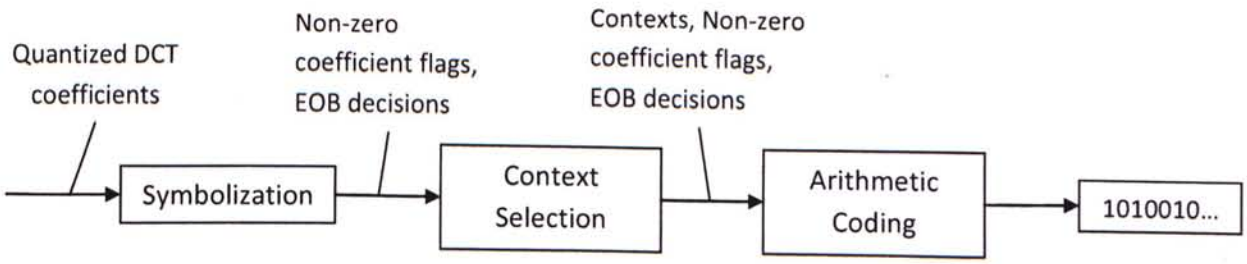


Fig. 14. First stage of the proposed scheme. The symbols to be coded are the non-zero coefficient flags and EOB decisions of transform coefficients.

In the first stage (Fig. 14), the non-zero coefficient flags and EOB decisions are first obtained from the quantized DCT coefficients through the 'Symbolization' procedure (Exp. 6). In the procedure, a non-zero coefficient flag is generated for each transform coefficient, with the symbol '0' indicates the transform coefficient is zero and the symbol '1' indicates the transform coefficient is non-zero. If the transform coefficient is indicated to be non-zero, then an additional EOB decision is generated for that non-zero transform coefficient, with the symbol '1' indicates the non-zero transform coefficient is the last coefficient in the block and the symbol '0' indicates that it is not.

Zig-Zag Scanning Position	1	2	3	4	5	6	7	8	9
Transform Coefficients	9	0	-5	3	0	0	-1	0	1
Non-zero Coefficient Flag	1	0	1	1	0	0	1	0	1
EOB Decision	0		0	0			0		1

Exp. 6. The symbolization procedure. Note that '1' is the last coefficient in the block.

Then, contexts that exploit the redundant information in the zig-zag scanning positions of transform coefficients are selected by the 'Context Selection' procedure to code the non-zero coefficient flags and EOB decisions. In the procedure, the zig-zag scanning position that the non-zero coefficient flag and EOB decision correspond to is used as the local information for context selection (Exp. 7 and 8). In such context design, each zig-zag scanning position corresponds to a context so that there are 64x2 contexts in total, 64 for coding the non-zero coefficient flags and 64 for coding the EOB decisions. The probabilities of the non-zero coefficient flags and EOB decisions are estimated independently within their own contexts indicated by the zig-zag scanning positions they correspond to. Results in Fig. 8 and 10 have shown that the probabilities of non-zero coefficients flags and EOB decisions can be estimated more accurately under such estimation compared to the estimation obtained from global statistics (Fig. 9 and 11).

Zig-Zag Scanning Position	1	2	3	4	5	6	7	8	9
Non-zero Coefficient Flag	1	0	1	1	0	0	1	0	1
Context Number	1	2	3	4	5	6	7	8	9

Exp. 7. Context selection for the non-zero coefficient flags. Each zig-zag scanning position corresponds to a context.

Zig-Zag Scanning Position	1	2	3	4	5	6	7	8	9
EOB Decision	0		0	0			0		1
Context Number	1		3	4			7		9

Exp. 8. Context selection for the EOB decisions. Each zig-zag scanning position corresponds to a context.

After the 'Context Selection' procedure, the non-zero coefficient flags and EOB decisions are coded using arithmetic codes along with their estimated probabilities provided by their respective contexts. In the proposed scheme, the probability of a symbol in a context is estimated by the adaptive probability estimation algorithm from the context-based arithmetic coding scheme of AVS [16]. In the algorithm, the

probability of a symbol in a context is estimated based on the frequency of occurrence of the previously coded symbols in the same context. The benefit for such estimation is that code tables that should be used to signal the probability information in the contexts are not required, provided that both the encoder and decoder use the same adaptive algorithm for probability estimation.

Note that the context design of the proposed scheme is different from the context design used in the significance map coding technique of H.264. In the latter one, a context selected by a zig-zag scanning position is used to code both the non-zero coefficient flag and EOB decision at that position so that there are only 64 contexts in total. In the former one, a zig-zag scanning position selects two independent contexts, one for coding the non-zero coefficient flag, the other one for coding the EOB decision. Separate contexts are used for coding the non-zero coefficient flag and EOB decision because the results in Fig. 8 and 10 show that the probability distributions of the non-zero coefficient flags and EOB decisions at the same zig-zag scanning positions are significantly different, which indicate that the probabilities of the two symbols should be estimated in separate contexts for a more accurate estimation. Experiments are carried out and indeed verify that using separate contexts to code the non-zero coefficient flags and EOB decisions would improve the compression efficiency of the proposed scheme considerably.

3.2.3.4 Coding of 'LEVEL'

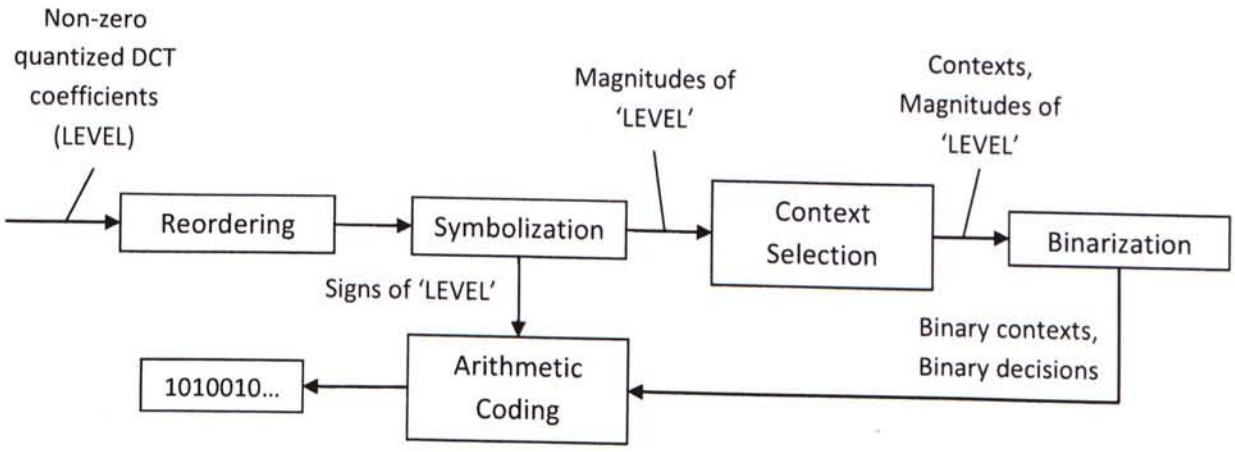


Fig. 15. Second stage of the proposed scheme. The symbols to be coded are the magnitudes of 'LEVEL' and the signs of 'LEVEL'.

The quantized DCT coefficients which are indicated to have zero magnitude are not required to be coded and therefore are discarded after the first stage. In the second stage (Fig. 15), the remaining non-zero quantized DCT coefficients, or equivalently the 'LEVEL' symbols, are first rearranged into reverse zig-zag scan order by the 'Reordering' procedure (Exp. 9). Then, the magnitudes of 'LEVEL' and signs of 'LEVEL' are obtained from the reordered 'LEVEL' through the 'Symbolization' procedure (Exp. 10).

LEVEL	9	-5	3	-1	1
LEVEL (Reordered)	1	-1	3	-5	9

Exp. 9. The reordering procedure.

LEVEL (Reordered)	1	-1	3	-5	9
Magnitude of 'LEVEL'	0	0	2	4	8
Sign of 'LEVEL'	0	1	0	1	0

Exp. 10. The symbolization procedure.

In the procedure, a sign flag is generated for each 'LEVEL' symbol, with the symbol '0' indicates the 'LEVEL' is positive and the symbol '1' indicates the 'LEVEL' is negative. Note that the magnitude of 'LEVEL' is defined as the magnitude of the non-zero transform coefficient subtracted by one. Since the transform coefficients with zero magnitude are discarded after the first stage, the remaining transform coefficients in the second stage must have magnitudes greater than or equal to one. Therefore, subtracting one from the magnitudes of the coefficients will not affect decoding and can save the bits for coding one less binary decision for the magnitude of each 'LEVEL' symbol at later coding process.

Following the 'Symbolization' procedure, the signs of 'LEVEL' are coded without contexts using arithmetic codes with a fixed probability of 0.5. Then, contexts that are designed to exploit the redundant information from both the zig-zag scanning position and magnitudes of previously coded coefficients are selected by the 'Context Selection' procedure to code the magnitudes of 'LEVEL'. In the procedure, two different local information, one from the zig-zag scanning position of the 'LEVEL' symbol, the other one from the L_{max} associated with the 'LEVEL' symbol in reverse zig-zag scan order (Exp. 5), are combined into the (Info 1, Info 2) format for context selection. 'Info 1' is defined as the zig-zag scanning position of the 'LEVEL' symbol and 'Info 2' is the quantized L_{max} associated with the 'LEVEL' symbol, and each pair of (Info 1, Info 2) corresponds to a context.

Note that L_{max} is quantized before being used as the local information for context selection so as to solve the issue of unrestrictedly large number of contexts, which is known to be the cause of the phenomenon called 'Context Dilution', in which the number of possible contexts is too large so that the number of symbols within some of the contexts are too small to allow an adaptive probability estimation algorithm to accurately estimate the probabilities of symbols within those contexts.

For reference, the quantization mapping for L_{max} is given by:

$$L_{qmax} = \begin{cases} L_{max}, & L_{max} \in [0,2] \\ 3, & L_{max} \in [3,4] \\ 4, & L_{max} > 4 \end{cases} \quad \ldots\ldots\ldots (2)$$

Where

L_{qmax} denotes the quantized L_{max}.

It should be noted that the same quantization mapping is also used by the context-based arithmetic coding scheme of AVS on motion-compensated video data. Such quantization mapping is obtained by minimizing the entropy of symbols under the set of contexts that exploit L_{max} for probability estimation with the constraint that the number of L_{qmax} values, or equivalently the number of contexts, after the quantization mapping is restricted to five. The quantization mapping is also verified to perform quite optimally on image data in the sense that the entropy of the magnitudes of non-zero transform coefficients under the set of contexts that exploit L_{qmax} for probability estimation obtained from a set of 24 standard test images (Lena, Peppers, Barbara, Baboon...etc) compressed at standard JPEG qualities (80, 85, 90, 95) is only larger than the entropy obtained under the set of contexts that exploit L_{max} directly for probability estimation by around 1%. Although it is found that a more optimal quantization mapping can be obtained by minimizing the entropy of the magnitudes of non-zero transform coefficients on the same set of standard test images compressed at the same JPEG qualities (80, 85, 90, 95) using the minimization procedures in AVS, the mapping obtained can only reduce the entropy further by around 0.1% compared with the entropy obtained using the quantization mapping in (2). Such observation indicates that the quantization mapping in (2) is already quite optimal for image data. Therefore, the same mapping is employed in this thesis for simplicity.

Magnitude of 'LEVEL'	0	0	2	4	8
Zig-Zag Scanning Position (Info 1)	9	7	4	3	1
Lmax	0	0	0	2	4
Lqmax (Info 2)	0	0	0	2	3
Context (Info 1, Info 2)	(9,0)	(7,0)	(4,0)	(3,2)	(1,3)

Exp. 11. Context selection for the magnitudes of 'LEVEL'. Note that the zig-zag scanning positions are decreasing as the 'LEVEL' symbols are in reverse zig-zag scan order. The zig-zag scanning positions for the magnitudes of 'LEVEL' are obtained from Exp. 6.

The 'Context Selection' procedure, which uses both the zig-zag scanning position and L_{qmax} of 'LEVEL' as the local information for selection is illustrated in Exp. 11. In the procedure, each (Info 1, Info 2) pair corresponds to a context, and the probabilities of the magnitudes of 'LEVEL' are estimated independently within their own contexts indicated by the (Info 1, Info 2) pairs. Since there are 64 zig-zag scanning positions and 5 values of L_{qmax}, the total number of contexts that (Info 1, Info 2) can select is 64x5.

The distinct advantage of such context design is that both the redundant information from the zig-zag scanning positions of transform coefficients (Fig. 6) and magnitudes of previously coded coefficients (Fig. 12) can be exploited in a single context design, which enables a more thorough exploitation of the redundancy in quantized DCT coefficients. The more thorough the exploitation of redundant information, the more accurate the contexts can estimate the probabilities of symbols. Hence, the proposed scheme with the context design in Exp. 11 incorporated are able to demonstrate an additional improvement in compression efficiency of about 2%-3% over JPEG two-pass Huffman coding compared to the scheme that only exploits the redundant information in the magnitudes of previously coded coefficients.

In preparation of arithmetic coding, each magnitude of 'LEVEL' is further decomposed into a sequence of binary decisions (1 or 0) using the 'Binarization'

procedure, which employs a unary binarization scheme similar to the one used in the context-based arithmetic coding scheme of AVS. The context in which the magnitude of 'LEVEL' is belonged is also decomposed into a sequence of binary contexts so that the probability of each binary decision in the sequence is estimated by a corresponding binary context. The purpose of such binarization is to speed up the arithmetic coding procedure by effectively reducing the number of partitions involved in the recursive subdivision process to only two. After the 'Binarization' procedure, each binary decision in the sequence is coded using arithmetic codes along with its estimated probability provided by the respective binary context.

3.2.3.5 Separate Coding of Color Planes

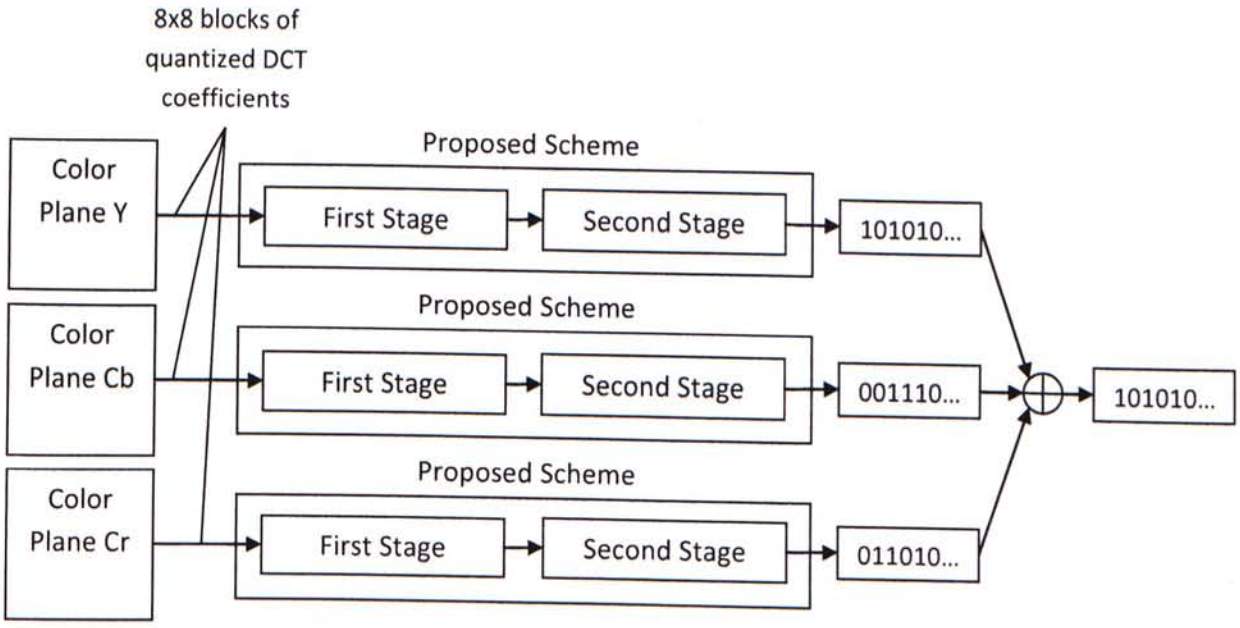


Fig. 16. Separate coding of color planes for the exploitation of color channel information.

The context design in section 3.2.3.3 and 3.2.3.4 only exploits intra block information for the coding of symbols. In order to estimate the probabilities of symbols more efficiently, color channel information is also exploited, in which the probabilities of symbols from the three color planes are estimated independently from each other to enable more accurate probability estimation. In practice, the quantized DCT coefficients from the three color planes can be treated as they are coded by three independent instances of the proposed schemes. Therefore, there are three identical and independent sets of contexts for coding the symbols from each of the three color planes. The bitstream produced by the schemes are concatenated together to form a complete bitstream (Fig. 16).

3.3 Experimental Results

3.3.1 Evaluation Method

The compression efficiency of the proposed scheme is tested on a set of 24 color images obtained from [17], which are in resolution of either 768x512 or 512x768. The set contains images with different characteristics, ranging from mostly smooth to highly irregular, and therefore is suitable as a test image set for evaluating the efficiency of compression algorithms under different image characteristics.

In the test, each of the test images is compressed at four JPEG qualities (30, 50, 70, 90) in 4:4:4 format using the proposed scheme and the compressed file size measured in total number of bytes is recorded. The compression efficiency of the scheme is then evaluated by the percentage of file size reduction over the default JPEG two-pass Huffman coding scheme. The same test is also repeated for other entropy coding methods for JPEG such that the compression efficiency of the proposed scheme can be evaluated against other entropy coding methods with reference to the file size obtained by JPEG two-pass Huffman coding. For reference, the calculation of file size reduction is given by:

$$File\ Size\ Reduction\ (\%) = \frac{F_H - F_P}{F_H} \times 100 \ldots\ldots\ldots (3)$$

Where

F_H denotes the compressed file size measured in bytes obtained using JPEG two-pass Huffman coding.

F_P denotes the compressed file size measured in bytes obtained using the proposed scheme or other entropy coding schemes for JPEG.

3.3.2 Methods under Evaluation

The entropy coding methods evaluated in the test include the proposed scheme, JPEG two-pass Huffman coding [14][15], JPEG arithmetic coding [14][15], the ITU-T arithmetic coding proposal [18][19], and the entropy coding method in [11].

In JPEG two-pass Huffman coding, the DC coefficient and AC coefficients of a block are coded separately. The scheme first scan through the entire image to collect the global statistics of SIZE symbols for predicted DC coefficients and (RUN, SIZE) symbols for AC coefficients, where 'SIZE' denotes the magnitude category of a transform coefficient and 'RUN' denotes the number of consecutive zeros preceding a non-zero transform coefficient. Then, the scheme assigns Huffman codes to the SIZE symbols of predicted DC coefficient and (RUN, SIZE) symbols of AC coefficients separately for each block according to the collected statistics. In such a method, exploitation of local information is limited only to the separate coding of DC coefficient and AC coefficients. Also, Huffman codes are restricted to have integral code length, therefore are inefficient in coding symbols which should be coded with fractional number of bits.

JPEG arithmetic coding is a context-based arithmetic coding scheme. Similar to JPEG two-pass Huffman coding, the scheme codes the DC coefficient and AC coefficients of a block separately. For DC coefficient coding, the scheme uses contexts that exploit information from the level of the predicted DC coefficient of the previously coded block to code the non-zero coefficient flag, sign, magnitude category and magnitude bits of the current DC coefficient. For AC coefficient coding, the scheme uses contexts that exploit information from the zig-zag scanning positions of transform coefficients to code the EOB decisions, non-zero coefficient flags, magnitude categories and magnitude bits of the AC coefficients. Although contexts are used in the scheme, the

context design of the scheme is inefficient in thorough exploitation of redundant information. For instance, the information from the magnitudes of coefficients is not exploited in the coding of transform coefficients.

The ITU-T proposal [18][19] defines an alternative arithmetic coder known as the Q15-coder for JPEG. The aim of the proposal is to substitute the patent-protected JPEG arithmetic coder by the patent-free Q15-coder. The Q15-coder inherits the same context design from JPEG arithmetic coding, but with different arithmetic coding procedures such as bit-stuffing and a new table for probability estimation. As a result, the Q15-coder demonstrates nearly identical compression efficiency to JPEG arithmetic coding, but can be used free of charge.

The entropy coding method in [11] is a modification of the JPEG arithmetic coding scheme. The method introduces a new AC coefficient coding algorithm which efficiently reduces the number of EOB decisions for each block and hence improves the compression efficiency of the EOB symbol. The JPEG arithmetic coding scheme codes the EOB decision after coding each non-zero AC coefficient. However, beginning AC coefficients of a block are generally non-zero thus some initial EOB decisions are redundant. The method efficiently avoid these redundant EOB decisions by start coding the first EOB decision only after a run of two zeros is encountered for the first time. The method inherits the same context design from JPEG arithmetic coding.

3.3.3 Average File Size Reduction

Table 7

Average File Size Reduction (%) Obtained by Different Entropy Coding Methods at Different JPEG Qualities

	JPEG Quality (%)			
	30	50	70	90
JPEG Arithmetic Coding	8.314	6.426	5.508	5.686
ITU-T Proposal	8.525	6.667	5.766	5.853
Entropy Coding Method [11]	10.367	8.423	7.424	7.073
Proposed Method	14.878	13.004	11.992	11.565

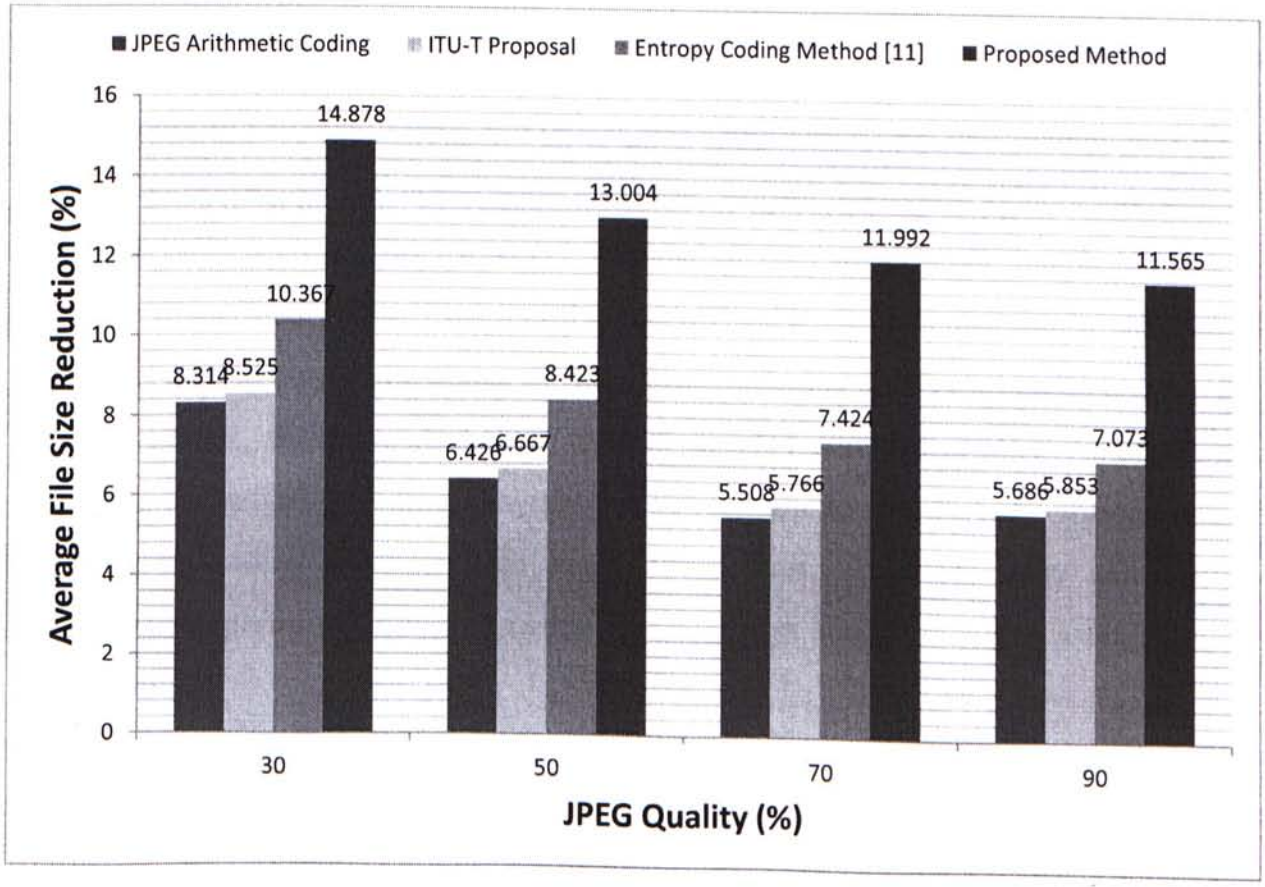


Fig. 17. Average file size reduction obtained by different entropy coding methods at different JPEG qualities.

Experimental results in table 7 and Fig. 17 show that the proposed scheme is able to outperform JPEG two-pass Huffman coding in terms of average file size reduction by 12%-15%. Moreover, the scheme also demonstrates superior compression efficiency to the other entropy coding methods for JPEG. For instance, the scheme has around twice the amount of average file size reduction over JPEG arithmetic coding and the ITU-T proposal [18][19], and is about 4%-5% better than the entropy coding method in [11].

An interesting observation on the results obtained by each entropy coding method is that they show an increasing trend in performance towards lower JPEG qualities. Such phenomenon may arise from the inefficiency of Huffman codes at lower JPEG qualities. At lower JPEG qualities, the variation in DCT coefficients is significantly reduced under the normalization effect of quantization. In JPEG two-pass Huffman coding, a certain SIZE symbol of DC coefficients and (RUN, SIZE) symbol of AC coefficients may have probability exceeding 0.5 under such situation. In practice, such symbols should be coded with less than one bit. However, because of the integral code length limitation, Huffman codes must at least use one bit to code such symbols, leaving a substantial amount of redundancy in the codes. It should be noted that JPEG arithmetic coding, the ITU-T proposal, the entropy coding method in [11] and the proposed method use arithmetic codes for the coding of symbols, thus are able to approach the entropy bound closer when coding symbols with high probabilities. Therefore, they are able to demonstrate superior performance over JPEG two-pass Huffman coding towards lower JPEG qualities.

3.3.4 File Size Reduction on Individual Images

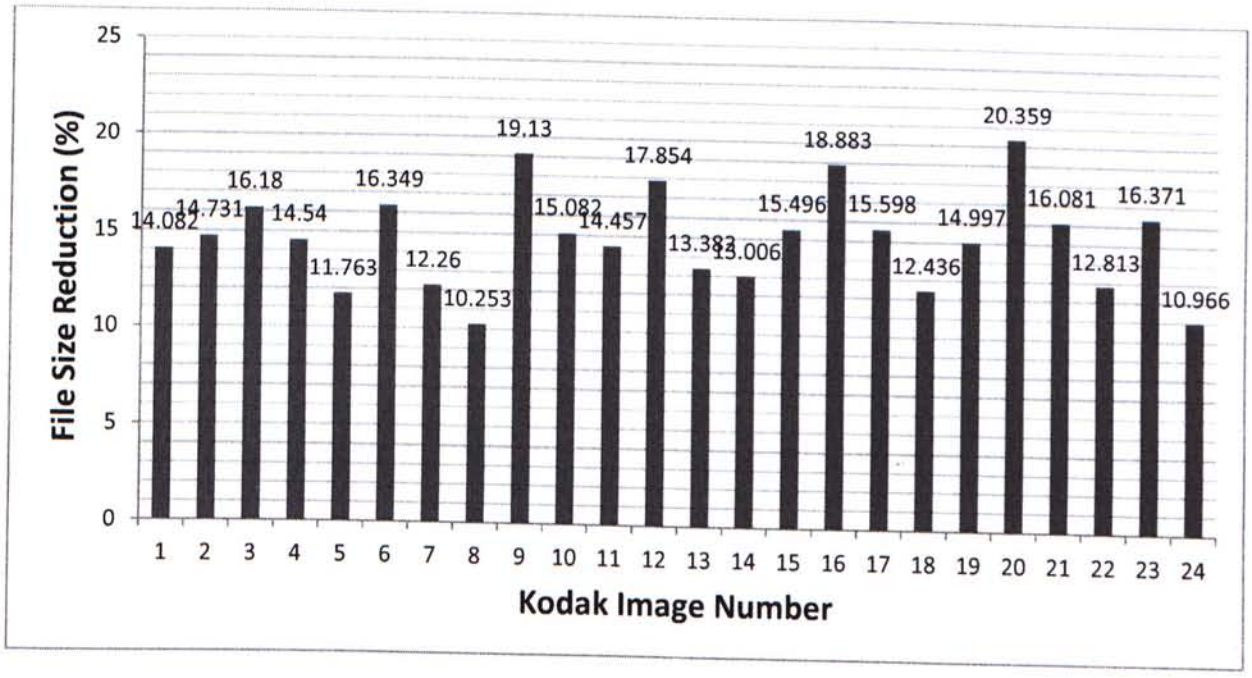


Fig. 18. File size reduction for individual images obtained by the proposed method at JPEG quality of 30.

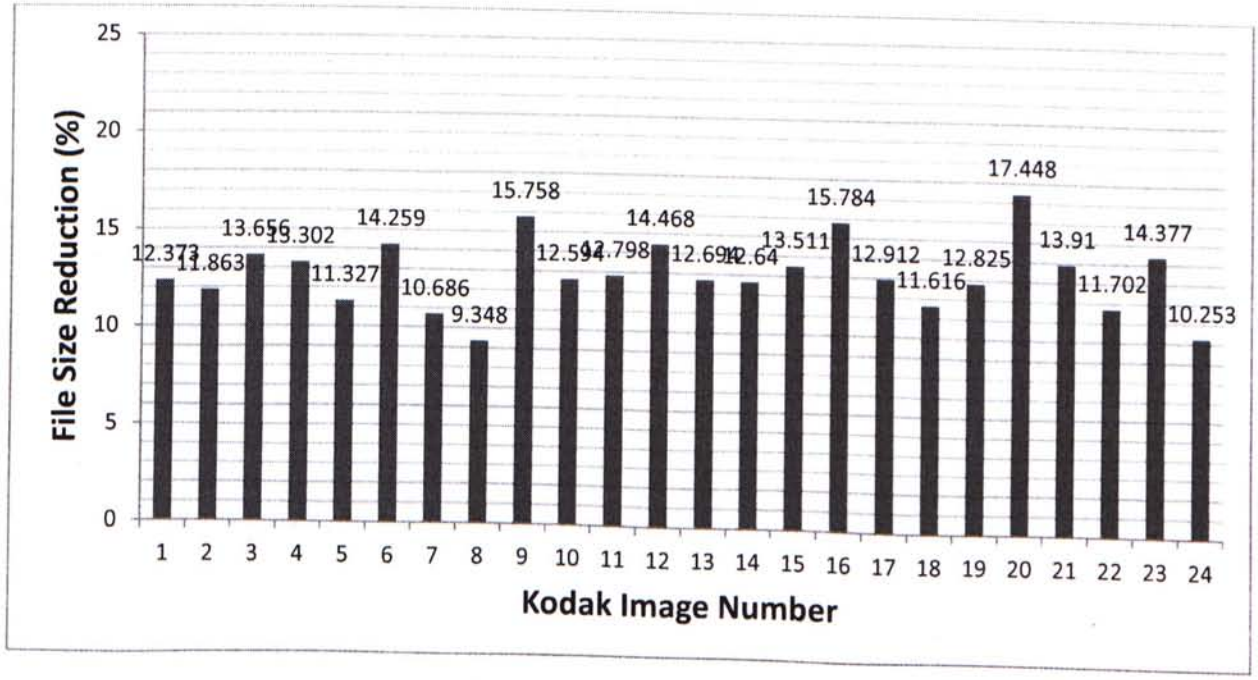


Fig. 19. File size reduction for individual images obtained by the proposed method at JPEG quality of 50.

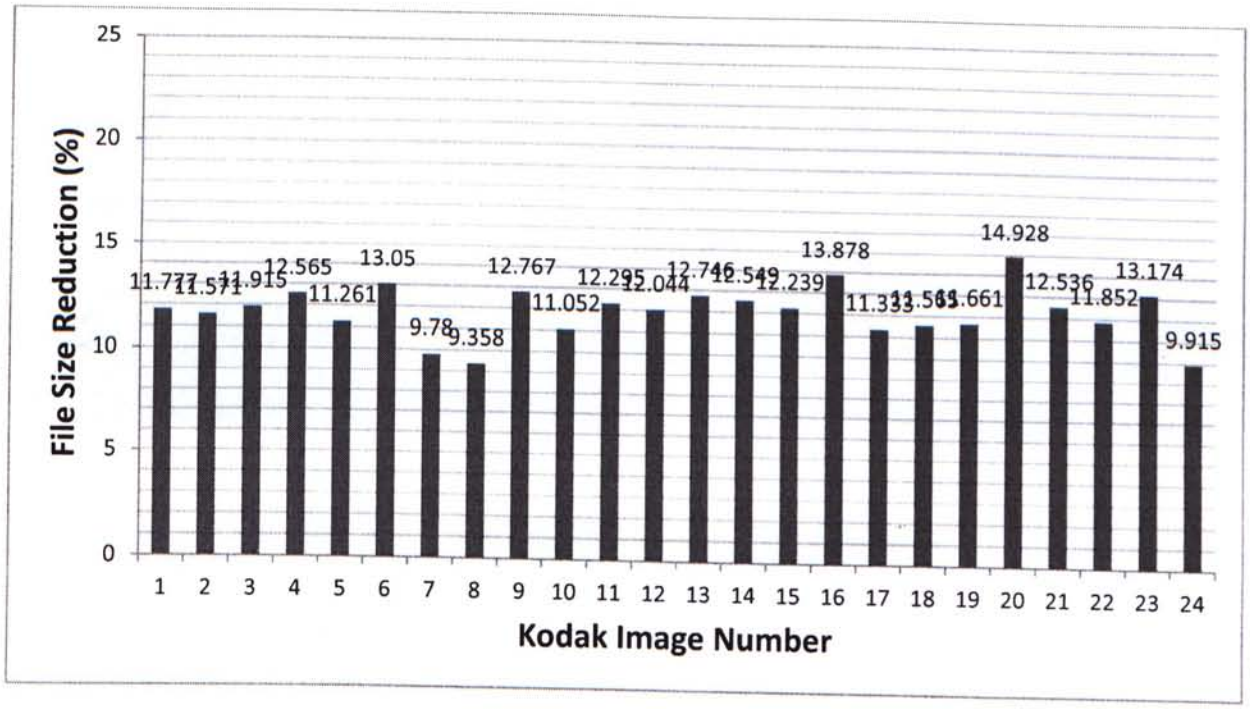


Fig. 20. File size reduction for individual images obtained by the proposed method at JPEG quality of 70.

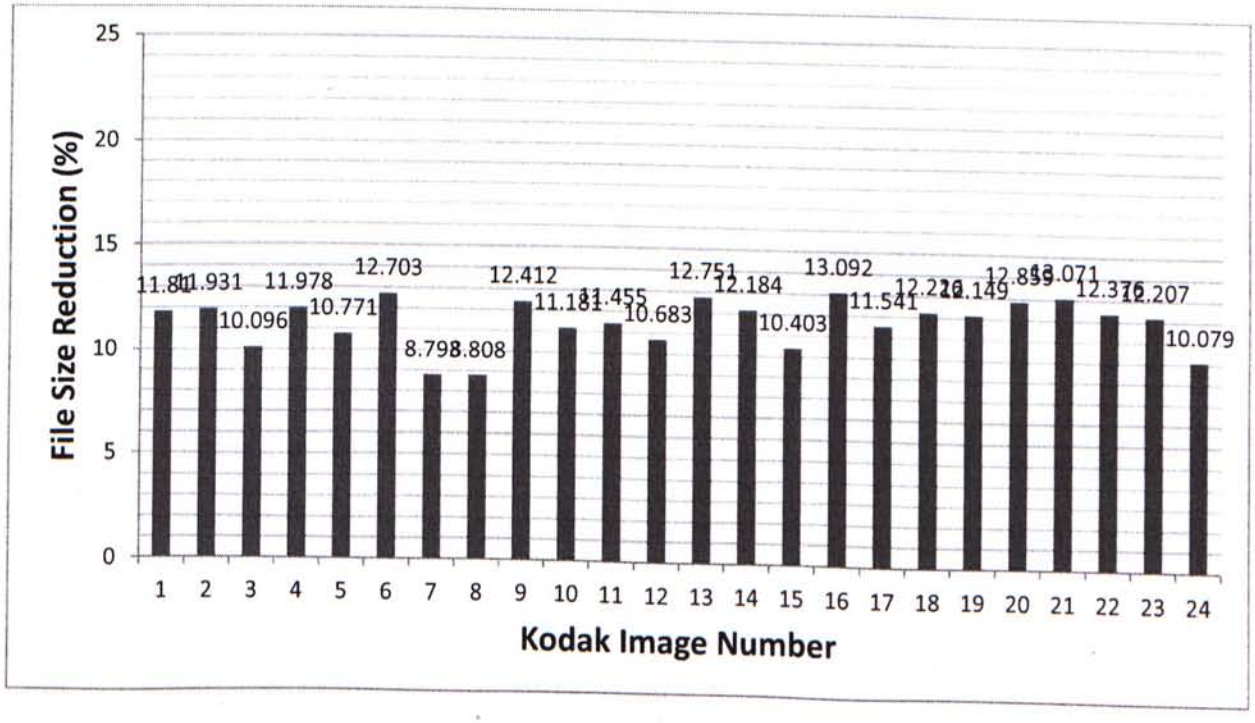
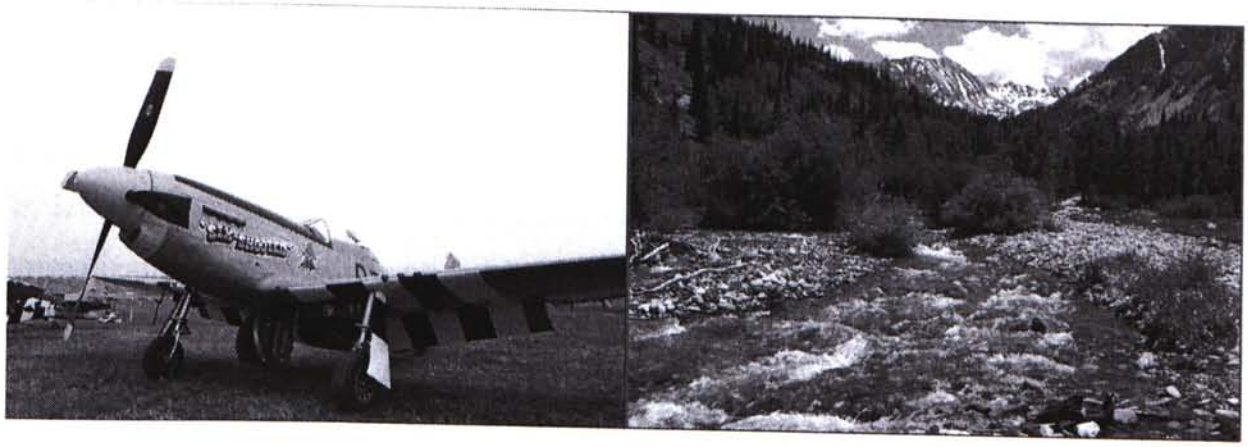


Fig. 21. File size reduction for individual images obtained by the proposed method at JPEG quality of 90.



(a) (b)

(c)

Fig. 22. Images with different characteristics. (a) An image with a large proportion of smooth region. (b) A highly irregular image. (c) An image with a large number of distinct edges.

Experimental results from Fig. 18 – Fig. 21 show that the proposed scheme is able to obtain reduction in file size over JPEG two-pass Huffman coding for every test image at various JPEG qualities. The percentage of reduction is ranging from 8% to 20% and is generally more significant towards lower JPEG qualities.

Moreover, it is observed that the performance of the proposed scheme is significantly better at lower JPEG qualities for images which contain large proportion of smooth region, such as Kodak image number 20 (Fig. 22a). In such images, the variation in DCT coefficients will be relatively smaller than the other images under the normalization effect of quantization at lower JPEG qualities. In such situation, it is likely that a certain SIZE symbol or (RUN, SIZE) symbol in JPEG two-pass Huffman coding would have dominant probability. Such symbols with high probabilities cannot be coded efficiently using Huffman codes due to the integral code length limitation. Thus, the proposed scheme, which uses arithmetic codes for the coding of symbols, would demonstrate significantly better performance at lower JPEG qualities for smooth images.

For highly irregular images, such as Kodak image number 13 (Fig. 22b), the performance of the proposed scheme remains relatively constant across JPEG qualities. In such images, the variation in DCT coefficient will be relatively higher than the other images, so that it is unlikely that there will be symbols with dominant probabilities in JPEG two-pass Huffman coding after quantization. In such case, Huffman codes can code symbols with relatively small redundancy in the codes, and therefore arithmetic codes in such situation does not have distinct advantage over Huffman codes. Hence, the proposed scheme demonstrates relatively constant performance across JPEG qualities.

It is observed that the proposed scheme demonstrates worse performance for images with a large number of distinct edges, such as Kodak image number 8 (Fig. 22c). In such images, the frequency contents of image blocks are changing abruptly across the image because of the presence of a large number of distinct edges. Local information from the frequency contents of image blocks, such as the magnitudes of previously coded coefficients, may not leave significant redundancy to be exploited as they are fluctuating abruptly across the image. In such situation, contexts may not be able to

estimate the probability of symbols as accurate as they can achieve in other images by exploiting local information, which result in worse performance of the scheme.

3.3.5 Performance of Individual Techniques

Experimental results from section 3.3.3 and 3.3.4 demonstrate the overall performance of the proposed scheme. In order to evaluate the performance of individual techniques of the scheme, the context-based arithmetic coding scheme of AVS [16], known as the Context-Based Binary Arithmetic Coding (CBAC), is treated as the reference for the proposed scheme. In particular, a series of modifications are performed to replace the techniques in CBAC by the techniques of the proposed scheme incrementally such that additional compression efficiency due to individual techniques of the proposed scheme can be measured with reference to the performance of CBAC.

In the CBAC scheme, the quantized DCT coefficients in zig-zag scan order of a block are first symbolized into the (LEVEL, RUN) pairs, where 'LEVEL' denotes a non-zero transform coefficient and 'RUN' denotes the number of consecutive zeros preceding that non-zero transform coefficient. Then, contexts that exploit redundant information in the magnitudes of previously coded coefficients in the block are used to code each (LEVEL, RUN) pair. Additionally, information from the magnitude of the 'LEVEL' symbol in a (LEVEL, RUN) pair is also exploited in coding the 'RUN' symbol of the same pair. In such scheme, redundant information in the zig-zag scanning positions of transform coefficients is not exploited in the coding of symbols. Moreover, the scheme uses the same set of contexts for coding symbols from different color channels. Hence, color channel information is also unexploited in the scheme.

In the experiment, three incremental modifications (A, B, C) are made to the CBAC scheme. In modification A, three independent sets of contexts are used to code symbols from the three color channels, with one set of contexts for each channel. The

context design for each set of contexts is the same as that in the original CBAC. The same method in exploiting color channel information is also used in the proposed scheme (Fig. 16). In modification B, the DC coefficient prediction technique of the proposed scheme (Fig. 13) is used to replace the one used by CBAC. CBAC is assumed to use the DC coefficient prediction technique from JPEG. In modification C, the CBAC scheme is replaced entirely by the proposed scheme (Fig. 14 and 15).

Experimental results from table 8 and Fig. 23 show that by exploiting color channel information in modification A, a further reduction in average file size from 1.7% to 4% over JPEG two-pass Huffman coding is obtained for the CBAC scheme. Comparing the results from table 7 and 8, it is interesting that the CBAC scheme without modification A demonstrates similar compression efficiency with JPEG arithmetic coding. However, such comparison may be unfair as JPEG has its own mechanism in exploiting color channel information. The results from modification B show that with the use of the DC coefficient prediction technique of the proposed scheme, a further reduction in average file size of about 0.4% to 0.6% is obtained for the CBAC scheme. The results from modification C show that the proposed scheme (Fig. 14 and 15) is more efficient in coding the quantized DCT coefficients than the CBAC scheme, which is indicated by a further reduction in average file size of about 3% for the CBAC scheme with modification C. The improvement in compression efficiency is brought by the context design of the proposed scheme in more thorough exploitation of the redundant information in quantized DCT coefficients. As a result, the probabilities of symbols are estimated more accurately in the proposed scheme, which reduces the number of bits required to code the symbols.

Table 8

Average File Size Reduction (%) Obtained by Different Modifications of CBAC at Different JPEG Qualities

	JPEG Quality (%)			
	30	50	70	90
CBAC	7.493	6.307	6.048	6.373
CBAC (Modification A)	11.505	9.589	8.694	8.154
CBAC (Modification B)	12.181	10.204	9.202	8.593
CBAC (Modification C)	14.878	13.004	11.992	11.565

Remark: CBAC with modification C is equivalent to the proposed scheme.

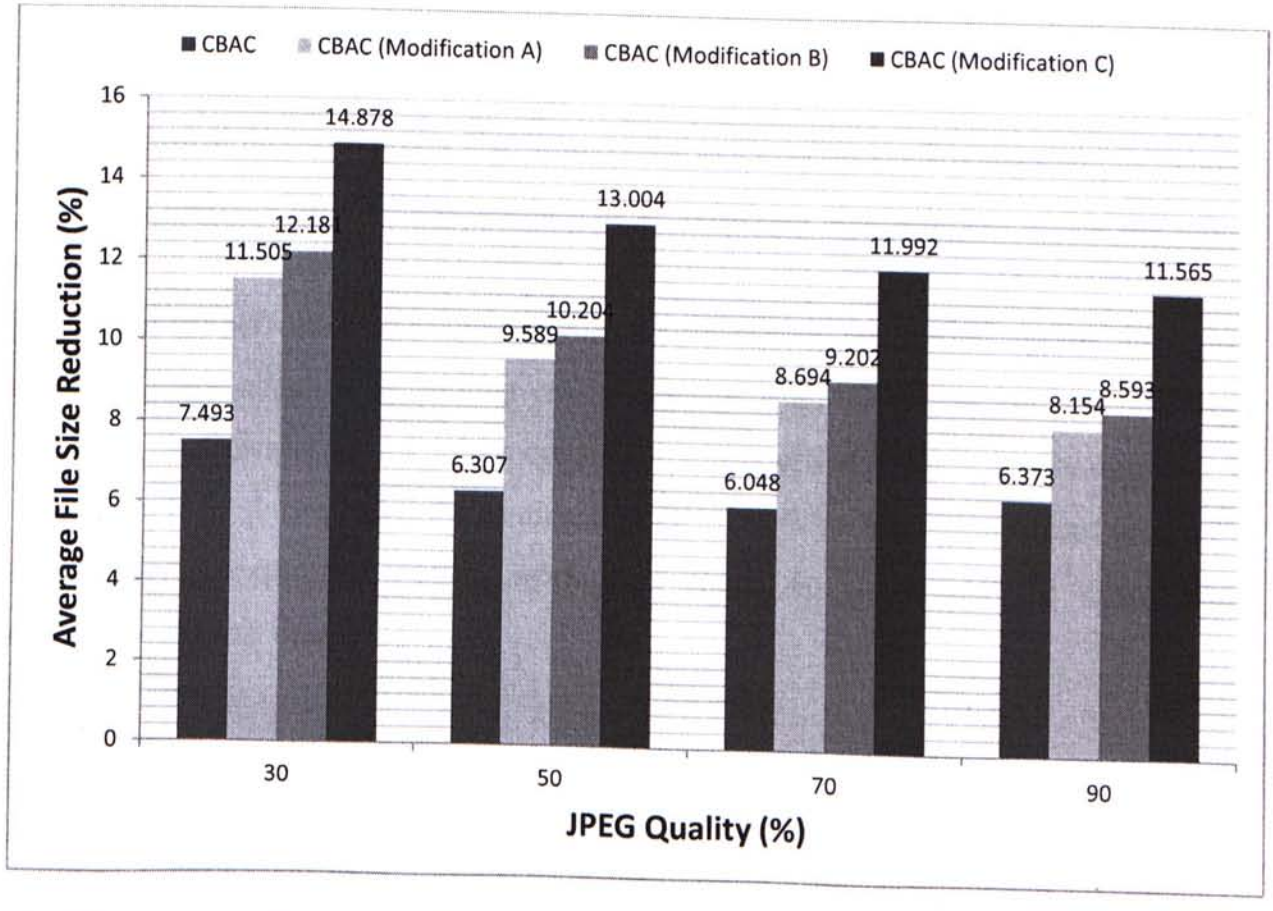


Fig. 23. Average file size reduction obtained by different modifications of CBAC at different JPEG qualities.

3.4 Discussions

There are several possible improvements to the proposed scheme. 1) The context design of the scheme focuses mainly on exploiting intra-block information. Non intra-block information, such as block-based information, inter-block and inter-colorplane information may be incorporated into the context design to exploit redundant information more thoroughly. For instance, a block-based classification method may be used to classify blocks into a finite number of classes according to their frequency contents. In such a method, blocks in the same class would have similar probability distribution of symbols, and therefore the probabilities of the symbols of these blocks can be estimated more accurately under the same context selected using their class information. 2) The probability of a symbol in a context is currently estimated using the adaptive probability estimation algorithm in the context-based arithmetic coding scheme of AVS. Analysis has not been performed to determine the accuracy of the estimation algorithm in the current context design. Future work may focus on either analyzing and improving the performance of the algorithm in the current context design, or developing an entirely new adaptive probability estimation algorithm for the current context design. 3) The EOB decisions account for a considerable proportion of the total file size, and the method in [11] showed that the coding of EOB decisions leave significant amount of redundancy unexploited. Future work may focus on improving the compression efficiency of EOB decisions.

4. Video Post-processing for H.264

4.1 Introduction

In [30], the authors have proposed a DCT domain image deblocking algorithm based on the concept of finite impulse response Wiener filter for reduction of blocking artifacts in low bit-rate images compressed by block-based DCT coding technique. The deblocking technique has been reported in [30] to perform well in terms of both objective and subjective measure. In particular, the technique is able to achieve state-of-the-art PSNR performance and excellent perceptual quality compared to other existing deblocking techniques.

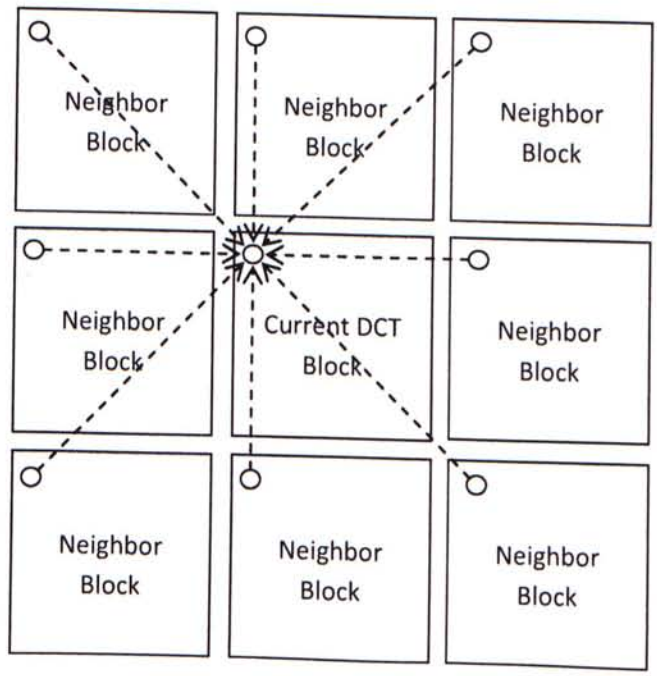


Fig. 24. A simplified diagram of the deblocking technique proposed in [30]. The circular shape in the current DCT block denotes the DCT coefficient under restoration.

The basic idea of the technique is to restore each DCT coefficient in a DCT block by using the collocated DCT coefficients from neighboring DCT blocks (Fig. 24). The

restoration is by means of forming a weighted average between the DCT coefficient under restoration and the collocated DCT coefficients, where the optimal weighting known as Wiener filter coefficients are found by minimizing the mean-square-error between the restored DCT coefficient and the original DCT coefficient. Such a formulation involves the estimation of the mean and variance of the DCT coefficients of the original image. For details, the reader is referred to [30].

4.2 Proposed Method

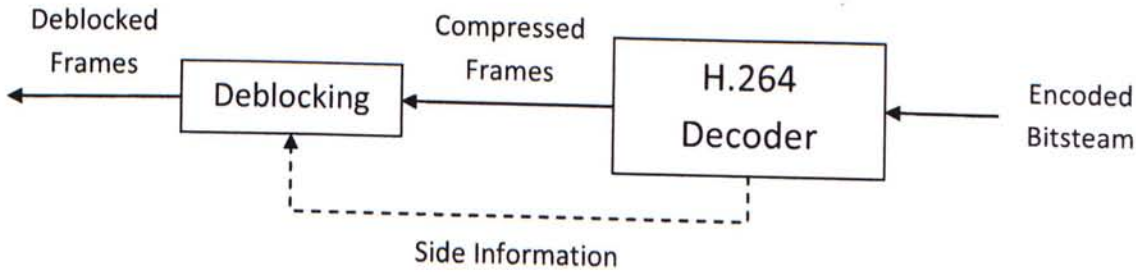


Fig. 25. Framework of the proposed method.

In this thesis, the application of the deblocking technique in [30] on H.264 compressed video is studied. In particular, experiments are carried out to investigate the performance of the deblocking technique by applying the technique on H.264 compressed video in a frame-by-frame basis (Fig. 25). In each experiment, a set of encoder settings are first chosen to generate a H.264 encoded bitstream. Then, the bitstream is decoded by the H.264 decoder to form the H.264 compressed frames, which will then be deblocked by the deblocking technique in [30] to obtain the deblocked frames. The objective and subjective visual quality of the deblocked frames are then recorded for evaluating the performance of the deblocking technique. Through varying the encoder settings, the performance of the deblocking technique can be tested on H.264 compressed frames with different characteristics.

4.3 Experimental Results

4.3.1 Deblocking on Compressed Frames

Encoder Settings

Profile:	High Profile
In-loop Filter:	Off
Prediction Structure:	Intra/Bidirectional Inter/Bidirectional Inter/Inter (IBBP)

Experiment Setup

- Technique applied to first four frames of Foreman (352x288) encoded at quantization parameter (QP) = 32.

Fig. 26. First four frames of Foreman (352x288) encoded at QP=32.

- Mean and variance estimated from the DCT coefficients of the compressed frames.
- Restoration performed on the DCT coefficients of the compressed frames.

Objective Performance

Table 9

PSNR Performance of the Deblocking Technique on the First Four Frames of Foreman

	PSNR Before (dB)	PSNR After (dB)	PSNR Gain (dB)
1st Frame (I)	35.388	35.503	0.115
2nd Frame (B)	34.664	34.766	0.102
3rd Frame (B)	34.604	34.748	0.144
4th Frame (P)	35.021	35.185	0.164

Remark: All parameters of the deblocking technique are fine tuned

Subjective Performance

(a) (b)

Fig. 27. Subjective performance of the deblocking technique. (a) 2nd frame of Foreman. (b) 2nd frame of Foreman after deblocking.

Supplementary Data

Table 10

PSNR Performance of the Deblocking Technique on the 2nd Frame of Foreman Compressed by JPEG at Different Bit-rates

Bits per pixel (bpp)	PSNR (dB)		
	JPEG	JPEG (Deblocked)	Gain
0.089	24.1230	25.6870	1.5640
0.132	26.7123	28.1338	1.4215
0.229	29.6595	30.8374	1.1779
0.373	32.2405	33.0571	0.8166
0.496	33.6018	34.3192	0.7174
0.600	**34.4817**	**35.2340**	**0.7523**
0.698	35.2213	35.8977	0.6764
0.807	36.0060	36.6857	0.6797
0.976	37.1030	37.7236	0.6206
1.229	38.6746	39.1164	0.4418
1.440	39.8505	40.1068	0.2563

Remark: All parameters of the deblocking technique are fine tuned

Discussion

Experimental results from table 9 and 10 show that the objective performance of the deblocking technique on H.264 compressed frames is insignificant. From table 10, the PSNR gain for the 2nd frame of Foreman compressed by JPEG at 34.48 dB is 0.75 dB, while from table 9 the gain for the same frame compressed by H.264 at 34.66 dB is only 0.10 dB. Also, the technique can only improve the subjective quality of the compressed frame slightly (Fig. 27). Blocking artifacts are still visible in the deblocked frame.

4.3.2 Deblocking on Residue of Compressed Frames

Encoder Settings

Profile: High Profile

In-loop Filter: Off

Prediction Structure: IBBP

Experiment Setup

- Technique applied to first four frames of Foreman (352x288) encoded at QP=32.
- Mean and variance estimated from the DCT coefficients of the prediction residue of the compressed frames.
- Restoration performed on the DCT coefficients of the prediction residue of the compressed frames.

Implementation

- The prediction residue is obtained by subtracting the prediction from a compressed frame. The prediction is supplied by the H.264 decoder as side information (Fig. 25).
- After restoration, the restored prediction residue is added to the prediction to form the deblocked frame.

Objective Performance

Table 11

PSNR Performance of the Deblocking Technique on the First Four Frames of Foreman with Restoration on Prediction Residue

	PSNR Before (dB)	PSNR After (dB)	PSNR Gain (dB)
1st Frame (I)	35.388	35.352	-0.036
2nd Frame (B)	34.664	34.661	-0.003
3rd Frame (B)	34.604	34.599	-0.005
4th Frame (P)	35.021	34.983	-0.038

Remark: All parameters of the deblocking technique are fine tuned

Subjective Performance

(a) (b)

Fig. 28. Subjective performance of the deblocking technique with restoration on prediction residue. (a) 2nd frame of Foreman. (b) 2nd frame of Foreman after deblocking.

Discussion

Experimental results from table 9 and 11 show that the objective performance of the deblocking technique on prediction residue is even worse. The PSNR gain for all four frames of Foreman is slightly negative. Also, the deblocked frame (Fig. 28) does not show improvement in subjective quality.

4.3.3 Performance Investigation

Experimental results from section 4.3.1 and 4.3.2 have shown that the deblocking technique is unable to achieve satisfactory performance on H.264 compressed frames (table 9 and 11) as oppose to its state-of-the-art performance on JPEG compressed images (table 10). An investigation is carried out to determine those coding techniques in H.264 which are critical to the performance of the deblocking technique.

The differences in coding techniques between H.264 and JPEG are studied and summarized in table 12. By using the coding techniques of JPEG as the reference point, one or more of the coding techniques in H.264 are replaced by those used in JPEG each time and experiments are carried out to investigate the performance of the deblocking technique on the compressed frames.

Table 12

Differences in Coding Techniques Between JPEG and H.264

Coding Techniques	JPEG	H.264
Prediction	N/A	Intra/Inter Prediction in Various Block Sizes
Transform	8x8 Discrete Cosine Transform (DCT)	4x4/8x8 Integer Cosine Transform (ICT)
Quantization	No Transform Post-Scaling and Quantization Rounding Threshold of 0.5	Transform Post-Scaling and H.264 Quantization Rounding Threshold
Entropy Coding	Huffman Coding	Context-Based Adaptive Binary Arithmetic Coding

4.3.4 Investigation Experiment 1

Encoder Settings

Profile: High Profile

In-loop Filter: Off

Intra/Inter Prediction: Off

Transform: 8x8 ICT Only

Experiment Setup

- Technique applied to the 2nd frame of Foreman (352x288) encoded at various QPs.
- Mean and variance estimated from the DCT coefficients of the compressed frames.
- Restoration performed on the DCT coefficients of the compressed frames.

Objective Performance

Table 13

PSNR Performance of the Deblocking Technique on the 2nd Frame of Foreman at Different Bit-rates with Prediction Off and 8x8 ICT

QP	Bits per pixel (bpp)	PSNR (dB)		
		H.264	H.264 (Deblocked)	Gain
20	1.5974	41.327	40.3892	-0.9387
24	1.1690	38.594	38.0094	-0.5846
28	**0.8350**	**35.937**	**35.6192**	**-0.3178**
32	0.5822	33.442	33.4865	0.0445
36	0.4061	31.122	31.4010	0.2790
40	0.2679	28.747	29.0826	0.3356
44	0.1699	26.265	26.7564	0.4914
48	0.1328	23.948	24.4125	0.4645
51	0.1039	22.458	22.9899	0.5319

Remark: All parameters of the deblocking technique are fine tuned

Discussion

Experimental results from table 10 and 13 show that the deblocking technique achieves insignificant objective performance with prediction off and 8x8 ICT. From table 10, the PSNR gain for the 2nd frame of Foreman compressed by JPEG at 36.01 dB is 0.68 dB, while from table 13 the gain for the same frame compressed by H.264 at 35.94 dB is -0.32 dB. The gain is also significantly higher for JPEG compressed frame at other PSNR.

4.3.5 Investigation Experiment 2

Encoder Settings

Profile:	High Profile
In-loop Filter:	Off
Intra/Inter Prediction:	Off
Transform:	**8x8 DCT Only**
Quantization:	**JPEG Quantization Scheme**

For reference, the quantization process is defined as:

$$Sq_{ij} = round\left(\frac{S_{ij}}{Q_{step}}\right) \ldots\ldots\ldots (4)$$

Q_{step} is related to QP by:

$$Q_{step}(QP) = Q_{step}(QP\%6) \cdot 2^{floor\left(\frac{QP}{6}\right)} \ldots\ldots\ldots (5)$$

QP	0	1	2	3	4	5
$Q_{step}(QP)$	0.625	0.6875	0.8125	0.875	1.0	1.125

where

S_{ij} is the DCT coefficient under quantization.

Sq_{ij} is the quantized DCT coefficient.

Q_{step} is the quantization step-size for S_{ij}.

Experiment Setup

- Technique applied to the 2nd frame of Foreman (352x288) encoded at various QPs.
- Mean and variance estimated from the DCT coefficients of the compressed frames.
- Restoration performed on the DCT coefficients of the compressed frames.

Objective Performance

Table 14

PSNR Performance of the Deblocking Technique on the 2nd Frame of Foreman at Different Bit-rates with Prediction Off, 8x8 DCT and JPEG Quantization

QP	Bits per pixel (bpp)	PSNR (dB)		
		H.264	H.264 (Deblocked)	Gain
24	1.4946	41.061	41.1204	0.0594
28	1.0802	38.159	38.6970	0.5380
32	0.7580	35.433	36.3238	0.8908
36	0.5443	33.147	34.3724	1.2254
40	0.3683	30.648	32.1301	1.4821
44	0.2322	28.289	29.9402	1.6512
48	0.1770	26.073	27.8026	1.7296
51	0.1335	24.416	25.9293	1.5133

Remark: All parameters of the deblocking technique are fine tuned

Discussion

Experimental results from table 13 and 14 show that the performance of the deblocking technique is vastly improved with 8x8 DCT and JPEG quantization scheme. The performance of the technique given in table 14 is more aligned with those given in table 10. The results show that either the 8x8 ICT or the H.264 quantization process is critical to the performance of the deblocking technique. Further investigation has shown that the 8x8 ICT and H.264 quantization process is essentially an efficient implementation of the 8x8 DCT and quantization scheme in JPEG, except with a different quantization rounding threshold.

4.3.6 Investigation Experiment 3

Encoder Settings

Profile:	High Profile
In-loop Filter:	Off
Intra/Inter Prediction:	Off
Transform:	8x8 ICT Only
Quantization:	H.264 Quantization with Rounding Threshold set to 0.5

Remark:

1. A rounding threshold denotes the smallest value the fractional part of a number has to reach in order for it to be rounded up. Otherwise it is rounded down.
2. H.264 uses a default rounding threshold of 0.833.
3. If the rounding threshold is set to 0.833 in the encoder settings, then the same results as in table 13 are obtained.

Experiment Setup

- Technique applied to the 2nd frame of Foreman (352x288) encoded at various QPs.
- Mean and variance estimated from the DCT coefficients of the compressed frames.
- Restoration performed on the DCT coefficients of the compressed frames.

Objective Performance

Table 15

PSNR Performance of the Deblocking Technique on the 2nd Frame of Foreman at Different Bit-rates with Prediction Off, 8x8 ICT and Rounding Threshold set to 0.5

QP	Bits per pixel (bpp)	PSNR (dB)		
		H.264	H.264 (Deblocked)	Gain
20	2.0093	43.841	43.5066	-0.3344
24	1.5260	40.880	41.0627	0.1827
28	1.1017	38.066	38.7191	0.6531
32	0.7645	35.355	36.3274	0.9724
36	0.5452	33.049	34.3416	1.2926
40	0.3697	30.606	32.1486	1.5426
44	0.2310	28.256	29.9026	1.6466
48	0.1757	26.023	27.7181	1.6951
51	0.1336	24.425	25.9111	1.4861

Remark: All parameters of the deblocking technique are fine tuned

Discussion

Experimental results from table 13 and 15 show that the performance of the deblocking technique is significantly improved with quantization rounding threshold set to 0.5. The results show that the rounding threshold is critical to the performance of the deblocking technique, which ascertains the finding obtained earlier in Investigation Experiment 2.

4.4 Discussions

Experimental results from section 4.3.1 and 4.3.2 have shown that the deblocking technique is unable to achieve significant improvement in both objective and subjective visual quality on H.264 compressed frames. The results are unsatisfactory regarding the state-of-the-art performance of the technique on JPEG compressed images. In order to determine the coding techniques in H.264 which are degrading the efficiency of the deblocking technique, a series of investigation experiments in section 4.3.4 – 4.3.6 are conducted to observe the performance of the deblocking technique under different H.264 coding techniques, and the rounding threshold in H.264 quantization is identified to be a critical factor. In particular, if the rounding threshold is changed from 0.833 to 0.5, the gain obtained from deblocking will be vastly improved. Further investigation is necessary to reveal the relationship between the rounding threshold and the performance of the deblocking technique.

5. Conclusions

In this thesis, new entropy coding and post-processing techniques for improving the compression efficiency of JPEG and H.264 are studied respectively.

In the first part of this thesis, a new context-based arithmetic coding scheme for improving the entropy coding efficiency of JPEG has been presented. The proposed scheme features a new context design that exploits redundant information in the zig-zag scanning positions of transform coefficients in a block, magnitudes of previously coded coefficients in a block, as well as the color channel information of transform coefficients, for efficient probability estimation of symbols. Experimental results have shown that the proposed scheme significantly outperforms JPEG two-pass Huffman coding and JPEG arithmetic coding in terms of average file size reduction, and demonstrates superior compression efficiency to the ITU-T arithmetic coding proposal [18][19] and the entropy coding method in [11].

In the second part of this thesis, the deblocking technique in [30] has been studied for application on H.264 compressed frames. Experimental results show that the technique has insignificant objective and subjective performance on H.264 compressed frames as oppose to its state-of-the-art performance on JPEG compressed images. An investigation has been carried out and found that the rounding threshold in H.264 quantization is critical to the performance of the deblocking technique. Further investigation is necessary to reveal the relationship between the rounding threshold and the performance of the deblocking technique.

References

[1] Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, and Ajay Luthra, ***"Overview of the H.264/AVC Video Coding Standard"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 13, no. 7, pp. 560-576, July 2003.

[2] Detlev Marpe, Thomas Wiegand, and Stephen Gordon, ***"H.264/MPEG4-AVC Fidelity Range Extensions: Tools, Profiles, Performance, and Application Areas"***, IEEE Int. Conf. on Image Processing (ICIP 2005), Sep. 2005.

[3] Lu Yu, Sijia Chen, and Jianpeng Wang, ***"Overview of AVS-Video Coding Standards"***, Signal Processing: Image Communication 24, pp. 247-262, 2009.

[4] Detlev Marpe, Heiko Schwarz, and Thomas Wiegand, ***"Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 13, no. 7, pp. 620–636, July 2003.

[5] D.A. Huffman, ***"A Method for the Construction of Minimum-Redundancy Codes"***, Proceedings of the I.R.E., pp. 1098–1102, Sep. 1952.

[6] Glen G. Langdon, ***"An Introduction to Arithmetic Coding"***, IBM Journal of Research and Development, vol. 28, no. 2, pp. 135-149, Mar. 1984.

[7] Ian H. Witten, Radford M. Neal, and John G. Cleary, ***"Arithmetic Coding for Data Compression"***, Communications of the ACM, vol. 30, no. 6, pp. 520-540, June 1987.

[8] Eric Bodden, Malte Clasen, and Joachim Kneis, ***"Arithmetic Coding Revealed"***, McGill University, Sable Research Group, May 2007.

[9] Gopal Lakhani, ***"Optimal Huffman Coding of DCT Blocks"***, IEEE Trans. on Circuits and Syst. for Video Technol., pp. 522-527, Apr. 2004.

[10] Gopal Lakhani, ***"Improving DC Coding Models of JPEG Arithmetic Coder"***, IEEE Signal Processing Letters, pp. 505-508, May 2004.

[11] Gopal Lakhani, ***"Remodeling JPEG Arithmetic Coder for Improved End-of-Block Marker Coding"***, Opt. Eng., vol. 43, no. 10, Oct. 2004.

[12] M. Stirner, and G. Seelmann, ***"Improved Redundancy Reduction for JPEG Files"***, Proc. of Picture Coding Symposium (PCS 2007), Nov. 2007.

[13] Ichiro Matsuda, Yukio Nomoto, Kei Wakabayashi, and Susumu Itoh, ***"Lossless Re-encoding of JPEG Images Using Block-Adaptive Intra Prediction"***, 16th European Signal Processing Conference (EUSIPCO 2008), Aug. 2008.

[14] William B. Pennebaker, and Joan L. Mitchell, ***"JPEG Still Image Data Compression Standard"***, Springer, 1st Edition, 1992.

[15] The International Telecommunication Union, Rec. T.81, ***"Information Technology – Digital Compression and Coding of Continuous-Tone Still Images – Requirements and Guidelines"***, Sep. 1992. Available: http://www.itu.int/rec/T-REC-T.81-199209-I/en

[16] Li Zhang, Qiang Wang, Ning Zhang, Debin Zhao, Xiaolin Wu, and Wen Gao, ***"Context-Based Entropy Coding in AVS Video Coding Standard"***, Signal Processing: Image Communication 24, pp. 263-276, 2009.

[17] *Kodak Lossless True Color Image Suite*. Available: http://r0k.us/graphics/kodak

[18] The International Telecommunication Union, Rec. T.851, ***"ITU-T T.81 (JPEG-1)-Based Still-Image Coding using an Alternative Arithmetic Coder"***, Sep. 2005. Available: http://www.itu.int/rec/T-REC-T.851-200509-I/en

[19] Joan L. Mitchell, and Arianne T. Hinds, ***"ITU-T T.851: An Enhanced Entropy Coding Design for JPEG Baseline Images"***, Proc. of SPIE, vol. 7073, 707313, 2008.

[20] William B. Pennebaker, Joan L. Mitchell, Glen G. Langdon, and Ronald B. Arps, ***"An Overview of the Basic Principles of the Q-Coder Adaptive Binary Arithmetic Coder"***, IBM Journal of Research and Development, vol. 32, no. 6, pp. 717-726, Nov. 1988.

[21] William B. Pennebaker, and Joan L. Mitchell, ***"Probability Estimation for the Q-Coder"***, IBM Journal of Research and Development, vol. 32, no. 6, pp. 737-752, Nov. 1988.

[22] N. Memon, ***"Adaptive Coding of DCT Coefficients by Golomb-Rice Codes"***, IEEE Int. Conf. on Image Processing (ICIP 1998), Oct. 1998.

[23] Li Zhang, Xiaolin Wu, Ning Zhang, Wen Gao, Qiang Wang, and Debin Zhao, ***"Context-Based Arithmetic Coding Reexamined for DCT Video Compression"***, IEEE Int. Symposium on Circuits and Syst. (ISCAS 2007), pp. 3147-3150, May 2007.

[24] Xiaolin Wu, ***"Lossless Compression of Continuous-Tone Images via Context Selection, Quantization, and Modeling"***, IEEE Trans. on Image Processing, vol. 6, no. 5, pp. 656-664, May 1997.

[25] Iain E. Richardson, ***"The H.264 Advanced Video Compression Standard"***, Wiley, 2nd Edition, 2010.

[26] Jae-Beom Lee, and Hari Kalva, ***"The VC-1 and H.264 Video Compression Standards for Broadband Video Services"***, Springer, 1st Edition, 2008.

[27] Khalid Sayood, ***"Lossless Compression Handbook"***, Academic Press, 1st Edition, 2003.

[28] Khalid Sayood, ***"Introduction to Data Compression"***, Morgan Kaufmann, 3rd Edition, 2005.

[29] David Salomon, and Giovanni Motta, ***"Handbook of Data Compression"***, Springer, 5th Edition, 2009.

[30] Renqi Zhang, Wanli Ouyang, and Wai-Kuen Cham, ***"Image Deblocking Using Dual Adaptive FIR Wiener Filter in the DCT Transform Domain"***, IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2009), pp. 1181-1183, April 2009.

[31] Renqi Zhang, Yiu-Leung Fong, and Wai-Kuen Cham, ***"Image Deblocking by the Dual Adaptive FIR Wiener Filter and Overcomplete Representation"***, 7th Int. Conf. on Information, Communications and Signal Processing (ICICS 2009), pp. 1-4, Dec. 2009.

[32] Yongyi Yang, Nikolas P. Galatsanos, and Aggelos K. Katsaggelos, ***"Projection-based Spatially Adaptive Reconstruction of Block-transform Compressed Images"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 4, no. 7, pp. 896-908, July 1995.

[33] Zixiang Xiong, Michael T. Orchard, and Ya-Qin Zhang, ***"A Deblocking Algorithm for JPEG Compressed Images Using Overcomplete Wavelet Representations"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 7, no. 2, pp. 433-437, Apr. 1997.

[34] Hoon Paek, Rin-Chul Kim, and Sang-Uk Lee, ***"On the POCS-Based Postprocessing Technique to Reduce the Blocking Artifacts in Transform Coded Images"***, IEEE. Trans. on Circuits and Syst. for Video Technol., vol. 8, no. 3, pp. 358-367, June 1998.

[35] Tao Chen, Hong Ren Wu, and Bin Qiu, ***"Adaptive Postfiltering of Transform Coefficients for the Reduction of Blocking Artifacts"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 11, no. 5, pp. 594-602, May 2001.

[36] Shizhong Liu, and Alan C. Bovik, ***"Efficient DCT-Domain Blind Measurement and Reduction of Blocking Artifacts"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 12, no. 12, pp. 1139-1149, Dec. 2002.

[37] A. Liew, and H. Yan, ***"Blocking Artifacts Suppression in Block-Coded Images Using Overcomplete Wavelet Representation"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 14, no. 4, pp. 450-461, Apr. 2004.

[38] Zhen Li, and Edward J. Delp, ***"Block Artifact Reduction Using a Transform-Domain Markov Random Field Model"***, IEEE. Trans. on Circuits and Syst. for Video Technol., vol. 15, no. 12, pp. 1583-1593, Dec. 2005.

[39] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian, ***"Pointwise Shape-Adaptive DCT for High-Quality Denoising and Deblocking of Grayscale and Color Images"***, IEEE. Trans. on Image Processing, vol. 16, no. 5, pp. 1395-1411, May 2007.

[40] Deqing Sun, and Wai-Kuen Cham, ***"Postprocessing of Low Bit-Rate Block DCT Coded Images Based on a Fields of Experts Prior"***, IEEE Trans. on Image Processing, vol. 16, no. 11, pp. 2743-2751, Nov. 2007.

[41] Guangtao Zhai, Wenjun Zhang, Xiaokang Yang, Weisi Lin, and Yi Xu, ***"Efficient Image Deblocking Based on Postfiltering in Shifted Windows"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 18, no. 1, pp. 122-126, Jan. 2008.

[42] Mark A. Robertson, and Robert L. Stevenson, ***"DCT Quantization Noise in Compressed Images"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 15, no. 1, pp. 27-38, Jan. 2005.

[43] Peter List, Anthony Joch, Jani Lainema, Gisle Bjontegaard, and Marta Karczewicz, ***"Adaptive Deblocking Filter"***, IEEE Trans. on Circuits and Syst. for Video Technol., vol. 13, no. 7, pp. 614-619, July 2003.