NOVEL WORD-LEVEL ALGORITHM OF EMBEDDED BLOCK CODING IN JPEG 2000

Hung-Chi Fang, Tu-Chih Wang, Yu-Wei Chang, Ya-Yun Shih, and Liang-Gee Chen

DSP/IC Design Lab
Graduate Institute of Electronics Eng. and Dept. of Electrical Eng.
National Taiwan University
NO. 1, Sec. 4, Roosevelt Road, Taipei 106, Taiwan
Phone: +886-2-23635251-332, Fax: +886-2-23681679
Email: {honchi, eric, wayne, yayun, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

In this work, a novel word-level algorithm of the embedded block coding in JPEG 2000 is proposed. Unlike conventional approaches, all bitplanes of coefficients are processed in parallel in the proposed algorithm. The algorithm is based on the observations and the analysis of the significant state of a coefficient and its contribution to neighbors. As a result of the analysis, there is only the coding pass of the MSB of a coefficient is the required information for the word-level processing. An algorithm for finding the coding pass of the MSB of a coefficient is also proposed. The number of scans of the entire code block is reduced to two scans and is independent of the number of bit planes.

1. INTRODUCTION

JPEG 2000 [1], [2] is the latest image coding standard. It is well known by its state-of-the-art compression ability as well as its abundant features such as region of interest (ROI), spatial scalability, signal to noise ratio (SNR) scalability and error resilience. Furthermore, it can support bi-level, gray-level, color or multiple component images and lossy or lossless coding in an unified algorithm. Although some features are also supported by JPEG [3] standard, they are provided by altering the coding algorithm to a certain extent. All the features in JPEG 2000 are provided in an unified algorithm by the arrangement of the embedded bitstreams. There are three major factors that make it possible to combine all features together. They are discrete wavelet transform (DWT), embedded block coding with optimized truncation (EBCOT) and the JPEG 2000 syntax.

D. Taubman proposed the EBCOT [4], [5] algorithm as the entropy coding engine for JPEG 2000 coding system. A fractional bit plane coding scheme is used in EBCOT that a bit plane is divided into three coding passes. EBCOT is a two tiered coding algorithm that the embedded block coding is called tier-1 and the rate-distortion optimized rate control is called tier-2. The embedded block coding is further divided into two major parts: context formation (CF) and arithmetic encoder (AE). For each sample coefficient, some context decision pairs (XD) are generated by CF engine and coded by the AE to form the embedded bitstream.

EBCOT accounts for the most computational load in JPEG 2000 coding system [6]. There are some software implementations of JPEG 2000 coding system. Both the verification model of JPEG 2000 standard [2] and JasPer [7] implement the EBCOT-tier1 algorithm in a bit plane by bit plane manner. Three scans are needed to code the three coding passes in a bit plane except the MSB bit plane of a code block. This is not an efficient way since the central processing unit (CPU) is not good at bit-level computations.

In this work, we propose a novel word-level algorithm for EBCOT tier-1. Unlike the conventional approaches, contexts of all bitplanes of a coefficient are done in parallel and saved. After all coefficients of a code block are scanned, the embedded bitstream are formed at a time.

This paper is organized as follows. Sec. 2 explains how to do the context formation without state variables. The proposed word-level EBCOT algorithm is described in Sec. 3. We compare our approach with conventional ones in Sec. 4. Finally, we conclude this paper in Sec. 5.

2. CONTEXT FORMATION WITHOUT STATE VARIABLES

EBCOT is a fractional bit plane coding algorithm that a bit plane is divided into three coding passes. The three coding passes are called significant propagation pass (pass 1), magnitude refinement pass (pass 2) and clean up pass (pass 3). An $n \times n$ code block is divided into $4 \times n$ stripes and the scan order of the coefficients is shown in Fig. 1. In the conventional approach, the first nonzero bit plane of the code block is scanned first, then the second one and so on. For each bit plane, three scans are needed to encode the three coding

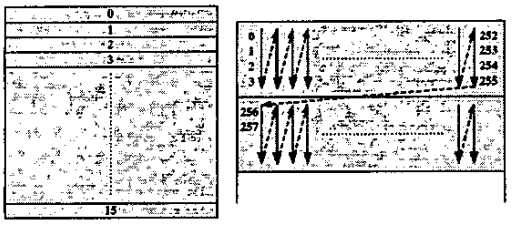

Fig. 1. Scan order of stripes in a code block and scan order of columns in a stripe.

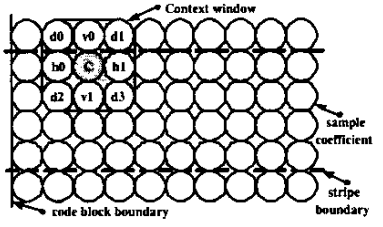

Fig. 2. Context window for context formation.

passes except the MSB which has only pass 3. Therefore, the number of scans required to finish a code block with n nonzero bitplanes is $3 \times (n-1)+1$. To explore detailed operation of the MQ coder, the reader is referred to [5]. We elaborate how to find the coding pass and do the context formation without state variables in the following sections.

2.1. Coding Pass Classification

Let μ_s denote the DWT coefficient where the suffix denotes its position as shown in Fig. 2. The sample coefficient lies at c is the one to be coded. Let p_c^k be the coding pass of the k-th bit plane of the central coefficient. It is determined by the contributions of its neighbors at bit plane k and the relative position of the MSB and k. The contribution of a neighboring coefficient at s to the k-th bit plane is represented by ϕ_s^k. For the neighbor whose scan order is posterior to the central coefficient, its contribution is determined on the related position of k and its MSB

$$\phi_s^k = \begin{cases} 0, & k \geq m_s \\ 1, & k < m_s \end{cases} \tag{1}$$

where

$$m_s = \begin{cases} -1, & \mu_s = 0 \\ m, & 2^m \leq \mu_s < 2^{m+1} \end{cases} . \tag{2}$$

On the other hand, the contribution of neighbor scanned prior to the central coefficient is

$$\phi_s^k = \begin{cases} 1, & k < m_s \\ 1, & k = m_s \cap p_s^{m_s} = 1 \\ 0, & otherwise \end{cases} . \tag{3}$$

Note that, $p_s^{m_s}$ is available because s is scanned prior c.

The sample coefficients located on h0, v0, d0 are always scanned prior to the central coefficient while those on h1, v1 and d3 are always posterior to the central coefficient. The scan orders of sample coefficient at d1 and d2 relative to the central coefficient are not always fixed. When the central coefficient is the first one in a column of the stripe, it is scanned posterior to the sample coefficient at d1; otherwise, it is scanned prior to the one at d1. On the other hand, the central coefficient is scanned prior to the coefficient at d2 when it is the last one in a column; otherwise, it is scanned posterior to the one at d2.

The coding pass of the k-th bit plane of the central coefficient is

$$p_c^k = \begin{cases} 2, & k < m_c \\ 3, & k \geq m_c \cap \sum \phi_s = 0 \\ 1, & otherwise \end{cases} \tag{4}$$

where m_c denotes the bit plane number of MSB of c.

2.2. Context Formation

The context are formed according to the contributions of significant states of the neighboring coefficients. These contributions are abbreviated as HVD. We define two new variables, λ_s^k and κ_s^k, as

$$\lambda_s^k = \begin{cases} 1, & k < m_s \\ 0, & k \geq m_s \end{cases} \tag{5}$$

$$\kappa_s^k = \begin{cases} 1, & k = m_s \\ 0, & k \neq m_s \end{cases} \tag{6}$$

where m_s is defined in (3).

2.2.1. Magnitude coding

Let σ_s^k denote the contribution of the k-th bit plane of the sample coefficient located at s with respect to the central coefficient. The group contribution data, HVD, for the context modelling can be obtained by

$$H^k = \sum \sigma_{hi}^k \tag{7}$$

$$V^k = \sum \sigma_{vi}^k \tag{8}$$

$$D^k = \sum \sigma_{di}^k. \tag{9}$$

For the sample coefficient scanned prior to the central coefficient, the contribution is given by

$$\sigma_s^k = \begin{cases} \lambda_s^k & \kappa_s^k = 0 \\ 1, & \kappa_s^k = 1 \cap p_c^k \neq 1 \cap p_s^k = 1 \\ 0, & otherwise \end{cases} . \quad (10)$$

For the sample coefficient scanned posterior to the central coefficient, the contribution is given by

$$\sigma_s^k = \begin{cases} \lambda_s^k & \kappa_s^k = 0 \\ 0, & \kappa_s^k = 1 \cap p_c^k \neq 3 \cap p_s^k = 3 \\ 1, & otherwise \end{cases} . \quad (11)$$

2.2.2. Sign Coding

Let χ_s denote the sign of the sample μ_s, where "1" stands for negative coefficient. We define two variables, α_s and β_s, as

$$\alpha_s = \sum_k \lambda_s^k \& \kappa_c^k \quad (12)$$

$$\beta_s = \sum_k \kappa_s^k \& \kappa_c^k \quad (13)$$

where λ_s^k and κ_s^k are defined in (5) and (6), respectively. Note that the value of α_s or β_s is either zero or one because there is at most one nonzero κ_c^k, which is one. The contribution of the coefficient prior to the central coefficient for sign coding is

$$\sigma_s^\chi = \begin{cases} \alpha_s, & \beta_s = 0 \\ 1, & \beta_s = 1 \cap p_c^{m_c} = 3 \cap p_s^{m_c} = 1 \\ 0, & otherwise \end{cases} \quad (14)$$

where m_c denotes the MSB of the central coefficient. The contribution of the coefficient posterior to the central coefficient for sign coding is

$$\sigma_s^\chi = \begin{cases} \alpha_s, & \beta_s = 0 \\ 1, & \beta_s = 1 \cap p_c^{m_c} = 1 \cap p_s^{m_c} = 3 \\ 0, & otherwise \end{cases} . \quad (15)$$

The truth table of the group contribution data for sign coding, H^χ or V^χ, is shown in table 1.

3. NOVEL WORD-LEVEL EBCOT ALGORITHM

Based on the equations derived in previous section, a new word-level EBCOT algorithm is proposed. The coding flow is shown in Fig. 3. All the coefficients in the code block are pre-scanned before context formation. The coding pass of the MSB of every coefficient is found and stored during pre-scan. Note that the MSB will be coded only in pass 1 or pass 3. We use a binary flag, say $p_s^{m_s}$, to represent whether it belongs to pass 1. The context of all bitplanes of a coefficient is formed at the same time when it is scanned. Finally, all contexts are processed by the arithmetic encoder at a time.

Table 1. Truth table of H^χ and V^χ for sign coding.

σ_{h0}^χ (σ_{v0}^χ)	χ_{h0} (χ_{v0})	σ_{h1}^χ (σ_{v1}^χ)	χ_{h1} (χ_{v1})	H^χ (V^χ)
1	0	1	0	1
1	0	0	X	1
0	X	1	0	1
1	1	1	0	0
1	0	1	1	0
0	X	0	X	0
1	1	1	1	-1
1	1	0	X	-1
0	X	1	1	-1

3.1. Prescan

The coding pass of MSB of the central coefficient is determined by

$$p_c^{m_c} = \begin{cases} 0 & m_s \leq m_c, \forall s \\ 1 & otherwise \end{cases} . \quad (16)$$

When $p_c^{m_c}$ is found, m_c must be updated as

$$m_c = (p_c^{m_c} == 1)?m_c + 1 : m_c \quad (17)$$

for future references. The updating process simplifies the computations of the p^{m_c} of coefficients scanned after this coefficient.

3.2. Context Formation

The pseudo-code for computing the H contribution is shown in Fig. 4. The number N represents the number of non-zero bitplanes of the code block. The approaches to get V and D contributions are similar to H. The pseudo-code for computing the H contribution of sign coding is shown in Fig.5. It is exactly the same for computing the V contribution. After the HVD data of the four coefficients in a column is acquired, the contexts are formed.

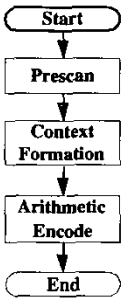

Fig. 3. Flow chart of the proposed word-level EBCOT tier-1 algorithm.

```
/* h0 does not contribute to c at bit plane > m_h0 */
for ( k = N-1; k > m_h0 ; k-- )
      h[k] = 0;

/* conditions that h0 contributes to c at m_h0 */
if ( p_c^m_c != 0 && p_h0^m_h0 == 0 ) // 0 means pass 3
      h[k] = 0;
else
      h[k] = 1;

/* h0 contributes to c at every bit plane < m_h0 */
for ( k = m_h0-1; k >= 0 ; k-- )
      h[k] = 1;

/* conditions that h1 contributes to c at m_h1 */
if ( p_c^m_c != 1 && p_h1^m_h1 == 1 )
      h[k]++;

/* h1 contributes to c at every bit plane < m_h1 */
for ( k = m_h1-1; k >= 0 ; k-- )
      h[k]++;
```

Fig. 4. Pseudo-code for computing the H contribution.

```
h0 = 1;          // h0 is scanned prior to c
if ( m_c > m_h0 )
      h0 = 0; // MSB of h0 is lower than that of c
else if ( m_c == m_h0 && p_c^m_c == 1 && p_h0^m_h0 == 0 )
      h0 = 0; // MSBs of h0 and c are the same but MSB of h0 is scanned after c
h1 = 0;          // h1 is scanned posterior to c
if ( m_c < m_h1 )
      h1 = 1; // MSB of h1 is higher than that of c
else if ( m_c == m_h1 && p_c^m_c == 0 && p_h1^m_h1 == 1 )
      h1 = 1; // MSBs of h1 and c are the same but MSB of h1 is scanned before c
h[N] = 0;
if ( h0 == 1 ) {
      if ( positive coefficient )
            h[N] += 1;
      else
            h[N] -= 1;
}
if ( h1 == 1 ) {
      if ( positive coefficient )
            h[N] += 1;
      else
            h[N] -= 1;
}
h[N] = CLIP (h[N],1,-1); // h[N]={-1,0,1}
```

Fig. 5. Pseudo-code for computing the H contribution of sign coding.

4. COMPARISONS

The proposed word-level EBCOT tier-1 algorithm needs only two scans for every coefficient to finish a code block while it needs $3 \times (N-1)+1$ scans in conventional approaches. Although there are some fast algorithms proposed, the lower bound is N for bit-level algorithm. The memory requirement of the proposed algorithm is higher than that of conventional approaches. The state variables in conventional approaches are not required in the proposed algorithm. The only state variable of the proposed algorithm is p_3^s, which is an one-bit flag. The additional memory requirement comes from the context-decision pairs since AE must processes them bit plane by bit plane. We can prevent from storing these information by choosing parallel mode. The performance of parallel mode is about $0.2dB$ lower than that of default mode. However, code-stream of parallel mode is more robust to errors than default mode.

5. CONCLUSIONS

In this paper, we proposed a novel word-level EBCOT tier-1 algorithm of JPEG 2000. By analyzing the significant state of a coefficient, a fast algorithm to find the coding pass of the MSB of a coefficient is proposed. The proposed algorithm can finish encoding a code block by only two scans of code block coefficients. The memory requirement of the proposed algorithm is higher than that in others. However, the size of memory is not critical in software implementation in today's technology.

6. REFERENCES

[1] ISO/IEC JTC1/SC29/WG1 N1855, *JPEG 2000 Part I: Final Draft International Standard (ISO/IEC FDIS15444-1)*, Aug. 2000.

[2] ISO/IEC JTC1/SC29/WG1 N1646, *JPEG2000 Verification Model 5.2 (Technical Description)*, Oct. 1999.

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

[4] D. Taubman, "High Performance Scalable Image Compression with EBCOT," *IEEE Trans. Image Processing*, pp. 1158–1170, July 2000.

[5] D. Taubman, E. Ordentlich, M. Weinberger, and G. Seroussi, "Embedded Block Coding in JPEG 2000," in *Proc. IEEE Int. Conf. Image Processing (ICIP 2000)*, 2000, pp. 33–36.

[6] K. F. Chen, C. J. Lian, H. H. Chen, and L. G. Chen, "Analysis and Architecture Design of EBCOT for JPEG-2000," in *IEEEE Int. Symp. Circuits and Systems (ISCAS 2001)*, 2001, pp. 765–768.

[7] M. D. Adams and F. Kossentini, "Jasper: A software-based jpeg-2000 codec implementation," in *Proc. IEEE Int. Conf. Image Processing (ICIP 2000)*, 2000, pp. 53–56.