An Efficient Context-Based BPGC Scalable Image Coder

Rong Zhang, Qibin Sun, and Wai-Choong Wong

Abstract—In this brief, we present an image entropy coder, context-based bit-plane Golomb coder (CB-BPGC), for wavelet-based scalable image coding. CB-BPGC follows the idea of the state-of-the-art image coding standard JPEG2000 entropy coding to apply the rate-distortion optimization algorithm after block coding. However, it explores a more efficient block coding where statistical properties of the block coefficients are considered. Compression ratio and error resilience performances of the proposed coder are evaluated, and the experimental results show that, compared with the JPEG2000, it achieves better lossless and lossy coding performance with lower complexity and greater resilience to transmission errors when simulated on the wireless Rayleigh fading channel.

Index Terms—Bit-plane Golomb coding, error resilience, image scalable coding, JPEG2000.

I. INTRODUCTION

WAVELET-BASED image coding is widely used in modern multimedia applications. It is often combined with bit-plane coding techniques to achieve an embedded bitstream which provides better bit-rate control and distortion scalability. Successful examples of these coders include the embedded zerotree wavelet coding (EZW) [1], the set partitioning in hierarchical trees (SPIHT) [2], and the embedded block coding with optimal truncation (EBCOT) [3].

EZW and SPIHT exploit the structures of the wavelet multiscale coefficients (zerotree in EZW and partitioned sets in SPIHT) and adaptively encode them to capture the interscale and intrascale dependencies, while EBCOT divides subbands to code blocks, and each block is independently and adaptively coded based on its context. Although EBCOT exploits only the intrascale dependencies, it owns attractive features such as resolution/quality scalability and error resilience. With good compression ratio and rich functionalities, EBCOT now serves as the entropy coder in the JPEG2000 standard (Part 1) [4].

In EBCOT *Tier 1* block coding, as shown in Fig. 1, bit planes are finely modeled by image context modeling techniques, and each bit in the bit plane is fractional bit-plane coded by a binary adaptive arithmetic coder (the MQ coder) [3]. In *Tier 2*, the post-compression rate-distortion optimization algorithm (PCRD) is applied to organize codestreams in an optimal way to generate the final bitstream according to bit-rate and distortion constraints [3]. As robust transmission of compressed data over error-prone channels has become an increasingly
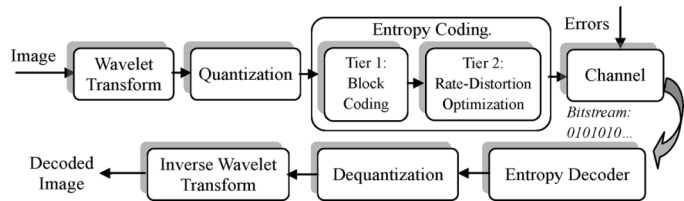
Fig. 1. Image encoding, transmission, and decoding of JPEG2000.

important requirement, error-resilient tools are adopted in JPEG2000 at both the entropy coding level and the packet level. When detecting errors, the decoder replaces the current and the following bit planes by zeros to prevent error propagation [4]. Extension work on error resilience is still ongoing in the new standard JPEG2000 for wireless applications (JPWL) [5].

For wavelet-based bit-plane coding, many coders adopt adaptive coding to capture the bit-plane correlations, but it is still inefficient in certain cases. For example, bits in some lower order bit planes are often uniformly distributed, where the MQ coder sometimes causes expansion rather than compression, i.e., direct transmission is more efficient than adaptive coding. Although EBCOT specifies a bypass mode to output raw bits of the lower bit planes, it has no systematic way to tell from which bit plane direct output is more efficient. In addition, for those low-resolution blocks with few coefficients and those texture-like blocks, the MQ coders sometimes cannot adapt to the local properties well and therefore lose some efficiency. Additionally, error resilience in JPEG2000 can also be improved. Bilgin *et al.* [6] point out there are dependencies among the coding passes of a code block, where partial decoding of a corrupted bitstream can be used to decode the bitstream as much as possible and thus improve the PSNR.

In this brief, we study the use of the context-based bit-plane Golomb coder (CB-BPGC) for image coding. By combining the BPGC [7], an embedded coding strategy for Laplacian distributed sources such as wavelet coefficients in HL, LH, and HH subbands, with image context modeling techniques, CB-BPGC provides better compression performance and is more resilient to errors compared with EBCOT. This brief is organized as follows. Implementation of the CB-BPGC is introduced in Section II. In Section III, we present the details of the error-resilient techniques used in CB-BPGC. Experimental results and the conclusions are then discussed in Sections IV and V, respectively.

II. PROPOSED ENTROPY CODER

The embedded coding strategy BPGC is first presented in [7], which is now successfully implemented in the MPEG-4 Audio

Scalable Lossless Coding (SLS) Standard [8]. We start this section with a brief review of the algorithm, followed by our implementation details for scalable image coding.

A. Bit-Plane Golomb Coding

Consider a Laplacian distributed source X, which has the probability density function given by

$$f_X(x) = e^{-|x|\sqrt{2/\sigma^2}}/\sqrt{2\sigma^2} \qquad (1)$$

where the magnitude of each sample $X_i(i = 1, 2, \ldots N)$ is binary represented by bit planes and the sign is uniformly distributed. If the source X is independent and identically Laplacian distributed (i.i.d.), the approximate probability of the bit $b_{j,i}$ in bit plane $B_j(j = 1, 2, \ldots m)$ can be described as

$$Q_j^L = \begin{cases} 1/(1 + 2^{j-L}), & j \geq L \\ 1/2, & j < L \end{cases} \qquad (2)$$

$$L = \min\left\{L' \in Z | 2^{L'+1}N \geq A\right\} \qquad (3)$$

where m is the most significant bit plane, N is the number of samples, and A is the absolute sum of the samples. By calculating N and A, parameter L can be easily obtained. L divides the bit planes into two parts: *lazy bit planes* where bits 0 and 1 have a probability of 1/2 and can be directly output, and *non-lazy bit planes* (the m bit plane to the L bit plane), whose skew probabilities are specified by the distance to the lazy bit plane $D2L : j - L$, and these bits are coded by the static arithmetic coder [7].

B. Context Modeling

Although BPGC is designed for Laplacian distributed data, it is not directly practical for coding wavelet coefficients. For audio scalable coding, the good performance of BPGC is based on the constraint that the coding source is i.i.d. and the fact that the audio signal is a one-dimensional (1-D) signal. As spatial dependencies of image wavelet coefficients are quite high, the BPGC static probability model where probability is specified only by $D2L$ would obviously lose some coding efficiency. Bits in the wavelet coefficient bit planes are significantly affected by the neighbors. For example, it is more likely for the current bit to be "1" when most bits of the corresponding neighbor bit planes are "1."

Fortunately, BPGC can be easily combined with image context modeling techniques, by which bit probabilities differ from each other according to both $D2L$ and neighborhood contexts $ctxt$. The seven $D2L$ contexts used in CB-BPGC are listed in Table I. Context 0 is for the lazy bit planes where raw bits are output without compression. There are also bit-plane coding examples in the table listing the corresponding $D2L$ context for each bit plane. For the neighborhood contexts $ctxt$, part of the contexts in EBCOT, which is widely used for image coding [9], is adopted [4]: nine contexts for the coefficients which are about to be significant in the current bit plane; three contexts for those already significant in previous coded bit planes. To reduce complexity, the sign bits are output directly while five contexts are

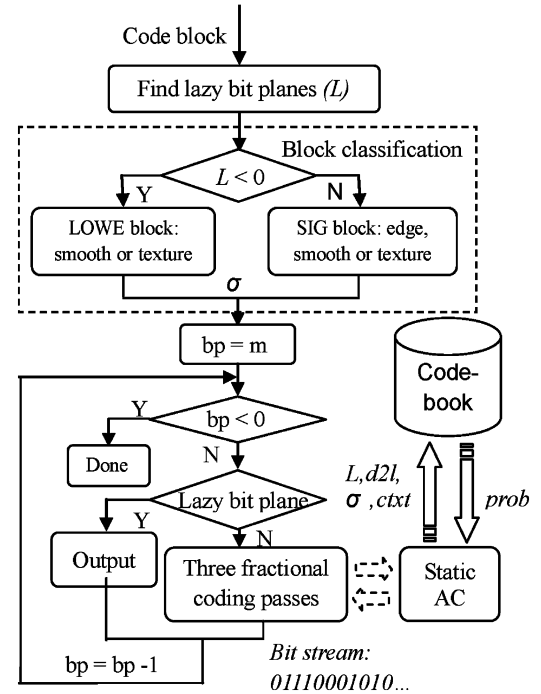| D2L contexts ($j-L$)	0 (≤-3)	1 (-2)	2 (-1)	3 (0)	4 (1)	5 (2)	6 (≥3)
Example 1: m:8 L:6	0,1,2,3	4	5	6	7	8	
Example 2: m:9 L:6	0,1,2,3	4	5	6	7	8	9
Example 3: m:7 L:4	0,1	2	3	4	5	6	7

Fig. 2. CB-BPGC encoding a block.

used for sign coding in EBCOT. Codebooks of bit probabilities can be trained offline and presaved in both the encoder and decoder.

C. CB-BPGC

CB-BPGC uses the same PCRD algorithm as in EBCOT *Tier 2* to pack the bitstream after block coding. Fig. 2. illustrates the block encoding process of CB-BPGC.

As shown in Fig. 2, after finding parameter L, there is a block classification procedure which is applied to model the coefficient local properties in a better way. Observation shows that blocks with $L < 0$ (LOWE blocks), most of which are in the very low-entropy high-frequency subbands, have quite different $D2L$ and $ctxt$ related bit probabilities compared with those blocks with $L \geq 0$ (SIG blocks). Table II tabulates the different bit probabilities of SIG and LOWE blocks for the nine significant coding $ctxts$ when $D2L = 3$. In addition, for the SIG blocks, three different classes appear with distinct bit probabilities as illustrated in Fig. 3. The three 64×64 blocks have the same most significant bit plane $m = 6$ and the same lazy bit plane $L = 3$, but the left one is smooth, the middle one seems more textural, and the right one contains an obvious edge. If each block is divided to smaller 8×8 subblocks (the number in the subblock area $subm_{x,y}$ indicates the most significant bit plane in the current subblock, and x and y are the subblock horizontal and vertical indices), we can see that the smooth block

TABLE II
PROBABILITY COMPARISON OF SIG AND LOWE BLOCKS FOR NINE
SIGNIFICANT CODING CONTEXTS WHEN $D2L = 3$

ctxt.	0	1	2	3	4	5	6	7	8
SIG	0.006	0.066	0.233	0.054	0.148	0.222	0.214	0.200	0.500
LOWE	0.011	0.118	0.263	0.076	0.152	0.333	0.242	0.341	0.300

Class 0: σ = 0.487   *Class 1: σ = 1.333*   *Class 2: σ = 2.254*
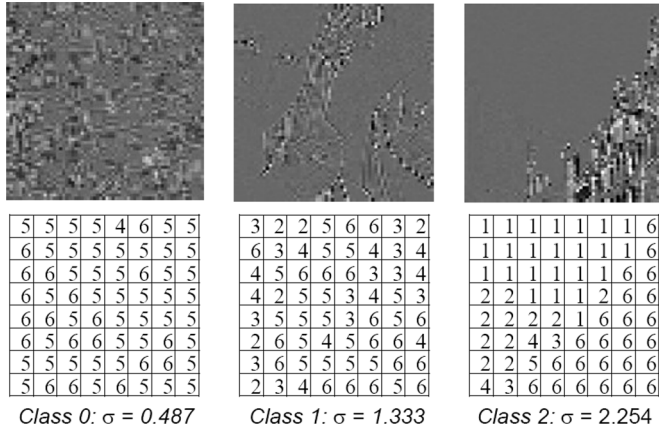
Fig. 3. SIG block classification example (three SIG blocks, 64×64, $m = 6$, $L = 3$. Left: class 0, smooth; middle: class 1, textual; right: class 2, edge).

has a smaller σ, the textual block has a median σ, and the edge block has a larger σ, where σ is the standard deviation of the *subm* array given by

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{x,y} (subm_{x,y} - \overline{subm})^2} \qquad (4)$$

where $\overline{subm}$ is the mean value of the *subm* array. Similarly, LOWE blocks can also be divided to two classes (smooth and edge) according to σ. We can train the thresholds of the parameter σ to classify all of the blocks. Blocks with the same class share the same codebook. A memory of about 3.7 KB is used for the codebooks in CB-BPGC.

After classification, CB-BPGC applies the three fractional bit-plane coding passes to these nonlazy bit planes. The static binary arithmetic coder compresses these bits with the lookup probabilities from the codebooks according to the parameters L, $D2L$, σ, and *ctxt*. In order to decode correctly, the parameters L and σ are transmitted as side information in the bitstream packet header.

III. ERROR RESILIENCE

Similar with JPEG2000, CB-BPGC hierarchically organizes the codestreams by subbands, blocks, and bit planes for resynchronization at the packet level to prevent error propagation. Fig. 4 illustrates the error-resilient strategies used in CB-BPGC. The static arithmetic coder terminates at each fractional bit plane to stop error propagation, followed by the segment marker "1010" after the coding pass 3, which is also the end of the current bit plane, and segment marker "10" after the coding passes 1 and 2. Whenever a mistake appears in decoding these markers, an error is detected.

1: Significant Propagation Coding Pass
2: Magnitude Refinement Coding Pass
3: Clear-up Coding Pass
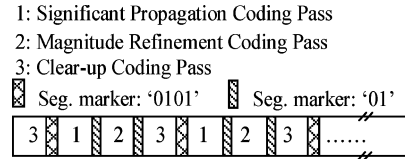⊠ Seg. marker: '0101'   ⧅ Seg. marker: '01'

Fig. 4. Error-resilient segment markers.

In the decoder, CB-BPGC conducts the partial decoding [6] for the following bit planes of the corrupted blocks.

Case 1) Error detected in coding pass 1.
No further coding passes 1 and 3 can be decoded, but coding pass 2 in the current bit plane can proceed.

Case 2) Error detected in coding pass 2.
No further coding pass 2 can be decoded, but coding passes 1 and 3 in the current and the following bit planes can proceed.

Case 3) Error detected in coding pass 3.
No further coding passes 1, 2, and 3 can be decoded.

Note that the error-resilient PSNR gain reported in [6] is based on the assumption that there is an external error-detection mechanism to tell the decoder from which byte in a fractional bit plane is corrupted, which leads to more complicated partial decoding applied on the fractional bit-plane level instead of the bit-plane level, i.e., additional information outside of the decoder helps to guide the decoding of the corrupted codestream. Our test results show that, by only using the internal error-detection method in CB-BPGC, substantial PSNR improvement can be obtained when the image is transmitted through a Rayleigh channel.

IV. EXPERIMENTAL RESULTS

The proposed coder is implemented with the Java implementation of JPEG2000 ($JJ2000$). The codebooks are trained from a large set of grayscale natural images taken by the authors. Ten typical grayscale JPEG2000 test images (such as *café*, *fruits*, and *Lena*) are used to evaluate the coding and error resilience performance.

A. Coding Performance

Table III shows the lossless compression ratio of five-level decomposition at different block sizes (64×64 and 16×16) with a reversible 5/3 filter. The numbers of bits per pixel for losslessly compressed images by JPEG2000 and the proposed CB-BPGC are listed. The positive numbers in the *Perc.* column are the percentages of CB-BPGC better than JPEG2000 and the negative ones are inverse. The average results show that CB-BPGC is more efficient than JPEG2000, especially for those images which seem harder to compress, e.g., *baboon* and *café*, where the adaptive coder probably fails to learn the complicated texture-like blocks well. In addition, JPEG2000 loses more efficiency in the case of smaller code block size where the number of the wavelet coefficients is small and the adaptive coder may end coding before it adapts to the context well, e.g., when the code block size is 16×16, CB-BPGC is 2.56% better on average. Fig. 5 gives an example for the bit-plane coding com-

[Online]. Available: http://jj2000.epfl.ch/

TABLE III
LOSSLESS PERFORMANCE FOR 5/3 REVERSIBLE DWT (bpp)

Image	Size	64×64			16×16		
		J2K	CB-B PGC	Perc.	J2K	CB-B PGC	Perc.
baboon	500×480	6.166	6.020	2.36%	6.626	6.412	3.22%
barb	720×576	6.249	6.143	1.69%	6.728	6.553	2.61%
fruits	640×512	4.149	4.168	-0.46%	4.538	4.451	1.91%
goldhill	720×576	4.645	4.609	0.78%	5.058	4.937	2.39%
lena	512×512	4.620	4.568	1.12%	5.022	4.871	3.01%
monarch	768×512	3.845	3.894	-1.28%	4.237	4.150	2.04%
woman	512×640	4.238	4.234	0.10%	4.619	4.532	1.89%
café	1024×1280	5.673	5.570	1.80%	6.148	5.966	2.95%
tool	1280×1024	4.402	4.414	-0.28%	4.826	4.708	2.44%
actors	1280×1024	5.408	5.320	1.62%	5.873	5.690	3.12%
average		4.957	4.894	0.75%	5.390	5.227	2.56%

level 0~4, 31 blocks (Café, 512*640, 5 level wavelet decomposition, blk size 64*64)

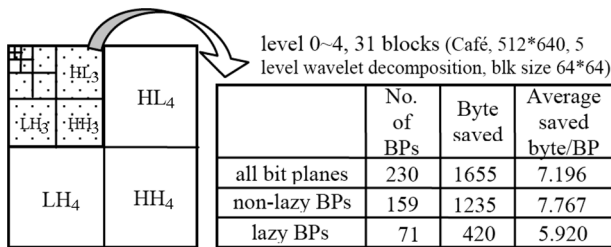|	No. of BPs	Byte saved	Average saved byte/BP
all bit planes	230	1655	7.196
non-lazy BPs	159	1235	7.767
lazy BPs	71	420	5.920

Fig. 5. Lossless compression comparison of image *café* for all of the code blocks in levels 0~4 between JPEG2000 and CB-BPGC.

TABLE IV
AVERAGE LOSSY PSNR PERFORMANCE (dB)

bpp	0.125	0.25	0.5	1	2	4
JPEG2000	25.92	28.25	31.25	35.18	40.51	47.69
CB-BPGC	25.90	28.28	31.32	35.31	40.76	48.09

parison results of levels 0~4 coefficients of image *café*. For the nonlazy bit planes, CB-BPGC removes more redundancy and therefore saves some bytes. The lazy bit-plane performance also shows that it is more efficient to output raw bits in these bit planes instead of adaptive coding them.

The average scalable coding performances of the images (9/7 filter, 5 level decomposition, block size: 16 × 16) are given in Table IV, which shows that CB-BPGC outperforms JPEG2000 in terms of PSNR except at very low bit rates. The PSNR increase is about 0.1 dB for a bi trate of 1 bpp and about 0.25 dB for a bit rate of 2 bpp on average. However, at low bit rates, JPEG2000 is better. This is probably because these LL subband coefficients, whose coding results affect the lossy coding performance a lot at the lower bit rates, are more like Rayleigh distributed, whereas BPGC is suitable for Laplacian distribution and cannot model them well. Another reason may be that we sacrifice some coding efficiency for the sign bits by direct transmission where in low-frequency subbands sign bits are compressible.

B. Error Resilience Performance

Fig. 6 shows the CB-BPGC average PSNR improvements of the ten images compared to JPEG2000 (5 level 9/7 filter, block size: 64 × 64) for different bit rates. Both of them are

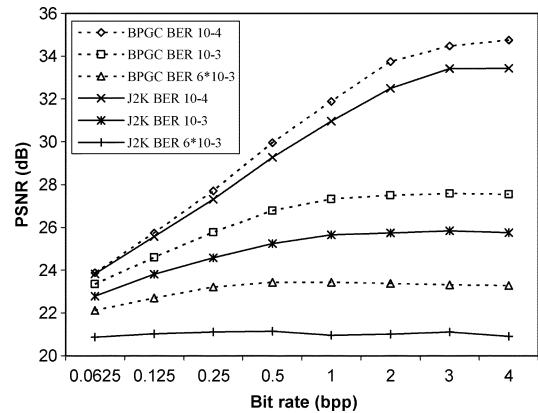

Fig. 6. Comparison of error resilience performance between JPEG2000 (solid lines) and CB-BPGC (dashed lines) at channel BERs of 10^{-4}, 10^{-3}, and 6×10^{-3}.

simulated though a wireless Rayleigh fading channel at bit error rates (BERs) of 10^{-4}, 10^{-3}, and 6×10^{-3} (the corresponding channel Eb_No parameters are 0.6, 1.3, and 2.1 dB, respectively; a detailed description of the channel conditions can be found in [10]). Each image is simulated over the Rayleigh channel 1000 times. CB-BPGC uses the error-resilient tools described in Section III, and EBCOT is set with the entropy level properties BYPASS, RESET, CAUSAL, ERTERM, SEGMARK, and RESTART. For both bitstreams, LL subband layers are protected from error corruption, which is the most important information and often assumed to be transmitted through a more reliable channel. As shown in Fig. 6, CB-BPGC is more resilient to errors with improved PSNR for all the bit rates on averages at 0.731, 1.514, and 2.097 dB for BERs at 10^{-4}, 10^{-3}, and 6×10^{-3}, respectively. It should be noted that, at higher BERs, both CB-BPGC and JPEG2000 have almost constant decoded image quality across all bit rates. This is probably because, when BER is high enough, almost all of the blocks are damaged from the most significant bit planes and, thus, the bit-rate increase cannot improve image quality [11]. Subjective results of the images *woman*, *Lena*, and *peppers* at a BER of 10^{-3} and 1 bpp are shown in Fig. 7. Comparing the decoded images, we can see that CB-BPGC gains not only a better PSNR in decibels but also a substantial improvement of subjective visual effect.

The improvement of CB-BPGC error-resilient performance is not only gained by partial bit-plane decoding, but also by the more efficient scalable coding. As the PCRD algorithm organizes codestreams according to the contribution of reducing distortion, i.e., in decreasing order, more efficient compression enables CB-BPGC to consume less bytes to embed the codestream while still providing the equivalent distortion reduction. Hence, when a transmission error occurs, it corrupts the less important bitstream of CB-BPGC and, thus, the PSNR result is better. Direct output of lazy bit planes also improves error resilience performance. Although errors may occur in certain lazy bit plane, we can further decode the remaining because the errors are isolated to certain coefficients instead of propagating to the others. In addition, CB-BPGC uses partial decoding to fully decode the corrupted codestream. Further improvement of error resilience

Fig. 7. Subjective results of (a)–(c) Lena, (d)–(f) bike, and (g)–(i) peppers at a bit rate of 1 bpp and a channel BER of 10^{-3}. (a) Error-free *woman* (256×256). (b) JPEG2000 (34.065 dB). (c) CB-BPGC (34.997 dB). (d) Error-free *Lena* (256×256). (e) JPEG2000 (32.153 dB). (f) CB-BPGC (34. 080 dB). (g) Error-free *peppers* (256×256). (h) JPEG2000 (27.251 dB). (i) CB-BPGC (29.254 dB).

can be achieved by applying some error concealment techniques such as edge-based concealment [12].

C. Complexity Analysis

CB-BPGC also has a lower complexity than JPEG2000. As mentioned in Section II, CB-BPGC and EBCOT differ mainly in the entropy coding *Tier 1* block coding, which is the most time-consuming part in JPEG2000 (the runtime percentages of grayscale image lossless and lossy encoding are about 71.63% and 52.26%, respectively [13]). Experimental results show that CB-BPGC consumes about 84.58% of the runtime of JPEG2000 *Tier 1* block coding for the lossless mode and 82.74% for the lossy mode, which means that about 11.04% of the lossless encoding time and 9.02% of the lossy encoding time are saved for grayscale images. The possible reasons for the lower complexity are as follows.

Direct output of the sign bits and bits in the lazy bit planes reduces some burden of the context modeling in CB-BPGC. As shown in the *cafe* example in Fig. 5, for the levels 0~4 code blocks, 30.9% of the bit planes are lazy bit planes ($D2L \leq 3$). We can further let more bit planes be lazy bit planes, for example, bit planes with $D2L \leq -1$, where 57.8% of the bit planes can be directly output. Experiments show that the average lossless coding performance is still better than JPEG2000 by 0.73% and 2.49% for block sizes 64×64 and 16×16, respectively. This tells us that, by sacrificing a little coding ef-

ficiency, computation complexity could be further reduced on these bit planes.

In addition, the static arithmetic coder is always simpler with lower computational complexity than the adaptive arithmetic coders by avoiding the probability adaptive procedure. A recent arithmetic coding complexity study [14] shows that the encoding time of a static arithmetic coder is about 58.6% of the MQ coder. Since a static arithmetic coder is used in CB-BPGC while JPEG2000 uses the adaptive MQ coder, the reduction of computation complexity of CB-PBGC is also achieved.

V. CONCLUSION

In this brief, we proposed a wavelet-based scalable image entropy coder, namely, CB-BPGC. By judiciously combining the embedded bit-plane coder BPGC, which explores the statistical characteristics of the wavelet coefficients, with image context modeling techniques, CB-BPGC outperforms JPEG2000 for both lossless and lossy coding performance. In addition, a lower complexity is also achieved. In addition, because of the partial decoding, direct transmission of *lazy bit planes*, and better compression ratio leading to corruptions of less important bitstreams, CB-BPGC is more resilient to transmission errors compared with the JPEG2000 standard.

REFERENCES

[1] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

[2] A. Said and W. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 243–250, Jun. 1996.

[3] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Trans. Image Process.*, vol. 9, no. 7, pp. 1158–1170, Jul. 2000.

[4] D. Taubman and M. W. Marcellin, *JPEG2000 Image Compression Fundamentals, Standard and Practice*. Boston, MA: Kluwer, 2002.

[5] *JPEG2000 Image Coding System—Part 11: Wireless JPEG2000—Committee Draft*, ISO/IEC JTC1/SC29/WG1 N3386, 2004.

[6] A. Bilgin, Z. Wu, and M. W. Marcellin, "Decompression of corrupt JPEG2000 codestreams," in *Proc. Data Compression Conf.*, 2003, pp. 123–132.

[7] R. Yu, C. C. Ko, S. Rahardja, and X. Lin, "Bit-plane Golomb coding for sources with Laplacian distributions," in *Proc. IEEE Int. Conf. Accoust., Speech, Singal Process.*, 2003, vol. 4, pp. 277–280.

[8] R. Yu, X. Lin, S. Rahardja, and H. Huang, Technical Description of I2R's Proposal for MPEG-4 Audio Scalable Lossless Coding (SLS): Advanced Audio Zip (AAZ) , ISO/IEC JTC1/SC29/WG11, M10035, Oct. 2003.

[9] L.-J. Kau and Y.-P. Lin, "Adaptive lossless image coding using least squares optimization with edge-look-ahead," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 52, no. 11, pp. 751–755, Nov. 2005.

[10] G. Wang and X. Lin, "Qos-guarantee error control for scalable image over wireless fading channel," in *Proc. VCIP*, 2002, pp. 37–45.

[11] S.-C. Diego and T. Ebrahimi, "An analytical study of JPEG2000 functionalities," in *Proc. Int. Conf. Image Process.*, 2000, vol. 2, pp. 49–52.

[12] S. Ye, Q. Sun, and E.-C. Chang, "Edge directed filter based error concealment for wavelet based image," in *Proc. IEEE Int. Conf. Image Process.*, 2004.

[13] C. Lian, K. Chen, H. Chen, and L. Chen, "Analysis and architecture design of block-coding engine for EBCOT in JPEG2000," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 3, pp. 219–230, Mar. 2003.

[14] A. Said, Comparative Analysis of Arithmetic Coding Computational Complexity Hewlett-Packard, HP Lab. Tech. Rep. HPL-2004-75, 2004.