J. Software Engineering & Applications, **2009, 2: 86-95**
doi:10.4236/jsea.2009.22013 Published Online July 2009 (www.SciRP.org/journal/jsea)

Lossy-to-Lossless Compression of Hyperspectral Image Using the 3D Set Partitioned Embedded ZeroBlock Coding Algorithm

Ying Hou

School of Communication and Information Engineering, Xi'an University of Science and Technology, Xi'an, China.
Email: houying@mailst.xjtu.edu.cn

Received December 28th, 2009; revised March 23rd, 2009; accepted April 22nd, 2009.

ABSTRACT

In this paper, we propose a three-dimensional Set Partitioned Embedded ZeroBlock Coding (3D SPEZBC) lossy-to-lossless compression algorithm for hyperspectral image which is an improved three-dimensional Embedded ZeroBlock Coding (3D EZBC) algorithm. The algorithm adopts the 3D integer wavelet packet transform proposed by Xiong et al. to decorrelate, the set-based partitioning zeroblock coding to process bitplane coding and the context-based adaptive arithmetic coding for further entropy coding. The theoretical analysis and experimental results demonstrate that 3D SPEZBC not only provides the same excellent compression performances as 3D EZBC, but also reduces the memory requirement compared with 3D EZBC. For achieving good coding performance, the diverse wavelet filters and unitary scaling factors are compared and evaluated, and the best choices were given. In comparison with several state-of-the-art wavelet coding algorithms, the proposed algorithm provides better compression performance and unsupervised classification accuracy.

Keywords: *Image Compression, Hyperspectral Image, 3D Wavelet Packet Transforms, Zeroblock Coding*

1. Introduction

Hyperspectral images provide high resolution and valuable spectrum information about the Earth's surface, so they are a useful tool and extensively applied in military and civilian fields. However, due to the huge amounts of data that bring about some problems in data transmission, storage and processing, more efficient compression technique becomes an indispensable task and a hot research topic.

In recent years, some hyperspectral image compression algorithms based on three-dimensional wavelet transform [1,2,3,4,5] are particularly interested thanks to their excellent compression performances and many attractive properties, such as the three-dimensional Set Partitioning in Hierarchical Trees (3D SPIHT) [4,5], the three-dimensional Set Partitioned Embedded bloCK (3D SPECK) [1], and the JPEG2000 multi-component (JPEG 2000-MC) [2,3]. The researches on hyperspectral image compression schemes can be generally classified into lossless and lossy techniques [1]. Lossless compression can exactly reconstruct the original images without losing any information. The state-of-the-art lossless compression methods are able to achieve compression ratios of 2 ~ 3.4 : 1, which is not enough to meet the actual compression requirements. Lossy compression can achieve higher compression ratio by discarding some information. Nevertheless, thanks to the extraordinary expense to collect hyperspectral images, sometimes we would not like to lose important data information that may affect the later applications. The lossy-to-lossless compression scheme combines the characteristics of two above-mentioned compression techniques and gives the option of the reconstructed image quality (lossy or lossless coding) according to the practical demands. The lossy compression results are obtained when the decoder truncates the lossless encoded bit stream at a desired rate. If the hyperspectral image is decoded without losing any information, it can be perfectly reconstructed. Recently, the researches in the lossy-to-lossless compression for hyperspectral images have been proposed. Tang and Pearlman [6] proposed a lossy-to-lossless compression solution to support random ROI access for hyperspectral image using the 3D SPECK algorithm. Wu *et al.* [7] present an asymmetric transform 3D SPECK (AT-3D SPECK) algorithm for hyperspectral image lossy-to-lossless compression. In Reference [8], Penna *et al.* propose a unified embedded lossy-to-lossless compression framework based on the JPEG 2000 standard. Zhang, Fowler and Liu [9] present a lossy-to-lossless hyperspectral image compression algorithm by using three-dimensional tarp-based coding with classification for embedding (3D TCE) and integer Karhunen-Loève transform (KLT).

The motion-compensated Embedded ZeroBlock Coding (MC-EZBC) [10] coder proposed by Hsiang and Woods is a successful scalable video compression algorithm and provides higher compression efficiency, lower computational complexity and some attractive features such as quality, resolution and temporal scalability. Hyperspectral image has higher correlation and not motion along spectral direction [1]. Thus, the 3D EZBC algorithm without motion compensation can achieve better coding performance for hyperspectral image compression. Whereas, because it needs to establish a quadtree representation structure for each individual 2D subband before starting the bitplane coding, the amount of memory required for quadtree structure is prominent and disadvantageous for the hyperspectral images compression. For a hyperspectral image with size $512\times 512\times 224$, the memory space of quadtree representation structure needs about 299.52 Mbytes. So, Hou and Liu take into account the characteristics of hyperspectral image, the excellent performance of the 3D EZBC algorithm, as well as the attractive properties of low memory requirements and fast encoding/decoding of the 2D SPECK algorithm [11], and then propose a three-dimensional Set Partitioned Embedded ZeroBlock Coding (3D SPEZBC) algorithm which is more suitable for hyperspectral image compression [12]. Instead of the partitioning coding method based on the quadtree representation structure in 3D EZBC, this algorithm adopts the partitioning coding method based on the set representation structure in 2D SPECK to process each individual 2D subband, so it can save higher memory requirements against 3D EZBC because the quadtree structure can be eliminated. For $512\times 512\times 224$ hyperspectral image, 75.52 Mbytes memory space is economized against 3D EZBC.

In this paper, we present a hyperspectral image lossy-to-lossless compression method based on the 3D SPEZBC algorithm, which adopts the Xiong's 3D integer wavelet packet transform (3D integer WPT) to decorrelate, the set-based quadtree partitioning zeroblock technique to process bitplane coding and the context-based adaptive arithmetic coding for further entropy coding. According to the extensive experiments and theoretical analyses, 3D SPEZBC provides the same excellent compression performances compared with 3D EZBC, saves the considerable memory requirement against 3D EZBC and exhibits the speed performance that is slightly worse than 3D EZBC. Furthermore, for achieving good coding per formance, we also evaluate different wavelet filters and unitary scaling factors based on the 3D integer WPT structure, and make the best choices. Compared with several state-of-the-art wavelet-based coding algorithms, the experimental results demonstrate that our algorithm can provide excellent compression performance and unsupervised classification accuracy. So the 3D SPEZBC algorithm is a good candidate for hyperspectral images lossy-to-lossless compression.

The remainder of this paper is organized as follows: Section 2 presents an overview of wavelet transform and Xiong's 3D integer wavelet packet decomposition structures with unitary scaling. In Section 3, the 3D SPEZBC algorithm for hyperspectral image lossy-to-lossless compression is described in detail. Furthermore, the discussions on the coding characteristics and the comparison between the 3D SPEZBC and 3D EZBC algorithm are given in Section 4. Section 5 provides the comprehensive experimental results for hyperspectral image compression. Finally conclusion is drawn in Section 6.

2. Three-Dimensional Wavelet Transform

The lifting scheme presented by Sweldens is the second generation wavelet transform and provides many attractive advantages. To realize lossy-to-lossless image compression based on wavelet transform, the integer-based lifting scheme [13] is an indispensable tool. It performs the reversible integer-to-integer wavelet transform by rounding and truncating each filter output. Many integer-based lifting wavelet transforms are proposed [14,15]. In this paper, we evaluate and compare the lossy-to- lossless compression performances by using some integer wavelet transforms, such as S+P(B), (2+2, 2), 5/3, etc.

Hyperspectral images can be viewed as 3D data and the image coding performance using 3D wavelet transform (WT) obviously outperforms those using 2D WT in most cases. However, there are diverse 3D WT structures [1,2,3] according to different decomposition order in the spatial-horizontal, spatial-vertical, and spectral-slice directions, namely 3D dyadic wavelet transform (DWT), 3D wavelet packet transform (WPT) and Xiong's 3D integer WPT. In recent years, researches have proven that the statistics of hyperspectral image are not symmetric along three dimensions and that higher correlation is exhibited in the spectral direction [1]. 3D WPT allows different decomposition levels in the spatial and spectral dimensions, and further performs spatial decomposition even in the higher-frequency spectral subbands. So it can achieve more flexible decomposition structure and preferable energy convergence in the space-frequency domain. Moreover, 3D integer WPT proposed by Xiong *et al.* [15] is capable of efficiently utilizing the statistical properties to decorrelate and gaining better compression performance for lossy-to-lossless coding. In our lossy-to-lossless compression coder, Xiong's 3D integer WPT is used to decorrelate hyperspectral images.

2.1 Xiong's 3D Integer WPT Structure

As shown in Figure 1, Xiong's 3D integer WPT first carries out an $L_{spectral}$ levels 1D WPT in the spectral-slice direction, which needs to further decompose the high-frequency component at even decomposition level, and then applies an $L_{spectral}$ levels 2D DWT to each resulting spatial image. If we adopt Xiong's 3D integer WPT of $L_{spectral}$ spatial levels and $L_{spectral}$ spectral levels for the hyperspectral image with F spectral bands, $K = (L_{spatial} \times 3 + 1) \times F$ individual 2D subbands can be generated. For example, Figure 1 shows the Xiong's 3D integer WPT structure with 56 2D subbands, as performing two spatial levels and two spectral levels for a volumetric image with 8 spectral bands.

2.2 Unitary Scaling Factor for 3D Integer Wavelet Transform

Because the integer-based lifting wavelet transform is not unitary, it badly compromises integer-based lossy coding performance [15]. To obtain better lossy coding performance, some researchers have presented a simple approach via bit shifting of wavelet coefficients to make the integer WT approximately unitary. In Reference 1, Tang *et al.* adopt Xiong's 3D integer WPT with unitary scaling [15] for hyperspectral image lossy-to-lossless compression. Nevertheless, thanks to fractional scaling factors, bits will be lost for right shift on the highest-frequency subbands, so all factors must be multiplied by the correctional times in order to make them be the nonnegative powers of 2. The correctional times used by Tang equal to four [1]. Through our experimental evaluations and analysis on the compression performances of several 3D integer WT structures with unitary scaling factor, we found that the Tang's unitary scaling structure with the correctional times can achieve effective integer-based lossy coding performances, but its lossless compression performance is degraded. Furthermore, unitary scaling structure adopted by Wu *et al.* [7] can obtain slightly better lossless performance than Tang's unitary scaling structure, but its integer-based lossy coding performance is worse than that of Tang's method. So we adopt a unitary scaling structure as in Figure 2's for hyperspectral image compression, which can obtain not only better lossless performance, but also excellent integer-based lossy performance.

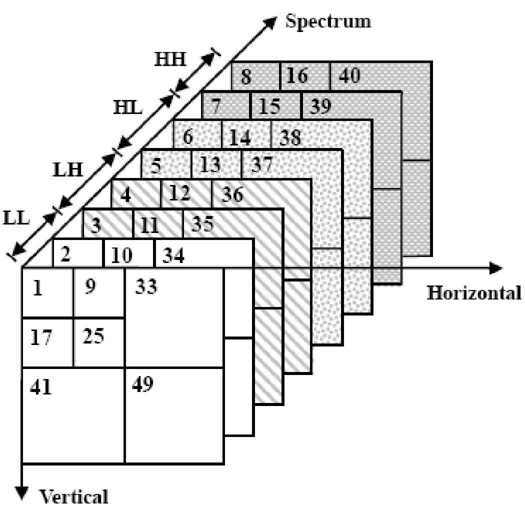


Figure 1. Xiong's 3D integer wavelet packet transform structures of two spectral levels and two spatial levels. The numbers on the front upper left corner of all subbands indicate the list initialization order of the 3D SPEZBC algorithm

3. The 3D SPEZBC Algorithm for Hyperspectral Image Coding

The 3D EZBC algorithm needs to establish a quadtree representation structure with the hierarchical pyramidal model for each individual 2D subband before starting the bitplane coding. This structure provides a fast quadtree splitting scheme, but its price paid needs much memory [10]. Especially, the memory cost is prominent and disadvantageous in the huge volumetric images compression, such as hyperspectral images, 3D medical images, etc. 3D SPEZBC is an embedded zeroblock bitplane coding algorithm by efficiently utilizing the energy clustering nature within subbands and the strong dependency across subbands. It adopts the set-based quardtree partitioning zeroblock coding and the context-based adaptive arithmetic

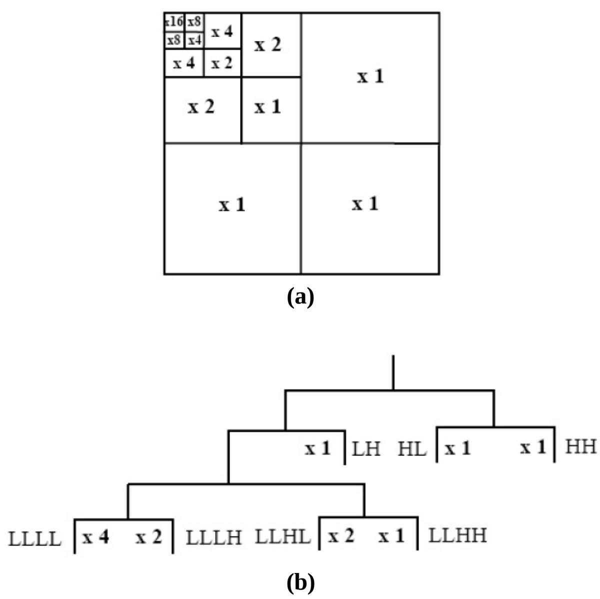


Figure 2. The unitary scaling factors after Xiong's 3D integer WPT of four spatial levels and four spectral levels. (a) The spatial scaling factors. (b) The spectral scaling factors

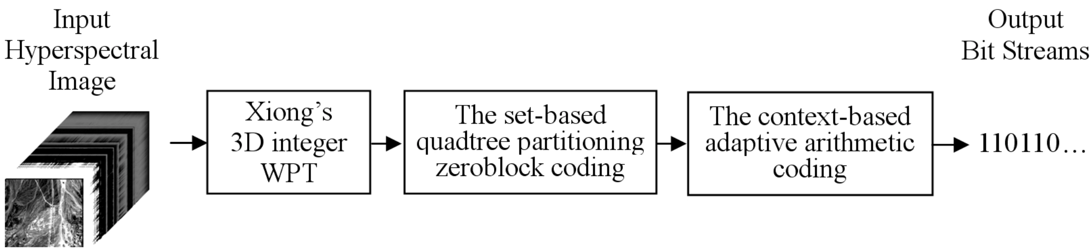


Figure 3. Block diagram of the hyperspectral image lossy-to-lossless coding system based on 3D SPEZBC

coding techniques. The block diagram of the hyperspectral image lossy-to-lossless compression coder based on 3D SPEZBC is illustrated in Figure 3 and the complete coding procedure is summarized as the following three steps.

1) Firstly, for the hyperspectral image, a hierarchical pyramidal structure is obtained by Xiong's 3D integer WPT with Figure 2's unitary scaling factor. In this structure, many rectangular 2D subbands with different sizes are generated, and each individual 2D subband is treated as a code block and defined as an initialization set. Whereafter, the code blocks are split and the significant coefficients are located via the set-based quadtree partitioning zeroblock coding technique.

2) Before starting the coding process, we define a set $S_{k,b}[l]$ to represent the code block of size $2^l \times 2^l$ at the spectral band b, subband k and set partitioning level l, as shown in Table 1. Where the set partitioning level l denotes the splitting depth from current code set to pixel-level sets (namely single pixel). For a set of size $M \times M$, it is defined by

$$l = \log_2 M$$

Substantively, l plays the same role as the quadtree level in 3D EZBC. Moreover, L_k denotes the set partitioning level of initialization set (namely k^{th} 2D subband), where k is the subband index ordeir (k = 0, 1, …, K-1) and K is the total number of the 2D subbands in wavelet decomposition image. At the same time, we define $L_{\max}$ to be the maximum set partitioning level among all initialization subbands. Table 1 lists the relationship among the code block, set and set partitioning level.

The 3D SPEZBC algorithm adopts the same list strategy used in 3D EZBC, and maintains two arrays of lists:

◆ *LIS*: List of Insignificant Sets.

◆ *LSP*: List of Significant Pixels.

In order to effectively use the statistical characteristics within individual subbands and set partitioning levels, some lists are separately established, namely $LIS_k[l]$ (*LIS* of the subband k and set partitioning level l) and LSP_k (*LSP* of the subband k), where k = 0, 1, …, K-1 and l = 0, 1, …, $L_{\max}$. Initially, k^{th} 2D subband at spectral band b is treated as a $S_{k,b}[L_k]$ set and added into $LIS_k[L_k]$ list according to the subband index order k of the marked numbers in Figure 1. In the coding procedure, the sets are successively added into corresponding $LIS_k[l]$ or LSP_k list in terms of their significance status.

3D SPEZBC adopts the set-based partitioning bitplane coding to progressively encode the wavelet coefficients of each subband from the Most Significant Bit (MSB) plane toward the Least Significant Bit (LSB) plane. In every bit-plane pass, all sets in LIS_k (*LIS* of the subband k) list are tested and coded from the bottom set partitioning level (l = 0 level) to the maximum set partitioning level (l = $L_{\max}$ level). Therefore, the sets of size 1×1 (single pixels) are coded first, and the sets of size 2×2 are coded next, and so on. If the set $S_{k,b}[l]$ contains the significant coefficients, it is tested significant against the current threshold. So set $S_{k,b}[l]$ of size $2^l \times 2^l$ at the set partitioning level l is partitioned into four approximately equal subsets $O(S_{k,b}[l]) = \{S^0_{k,b}[l-1],\ S^1_{k,b}[l-1],\ S^2_{k,b}[l-1],\ S^3_{k,b}[l-1]\}$ of size $2^{l-1}\times 2^{l-1}$ at the set partitioning level l-1. Subsequently, each subset is treated as a new set, and in turn these new sets $O(S_{k,b}[l])$ are further tested and processed in the same way above, as shown in Figure 4(b). Whole partitioning process is recursively executed until the pixel-level sets are reached, so that all significant pixels in subband are located and then added into LSP_k list for further refinement coding. The 3D SPEZBC coding algorithm is described later in detail.

3) Finally, in order to further improve the coding performance, 3D SPEZBC makes use of the context-based adaptive arithmetic coding approach in 3D EZBC to encode the significance map, signs and refinement bitstreams. Although our algorithm gets rid of the quadtree structure, it can also make use of the set-based partitioning process to build upon the similar context models with hierarchical pyramidal structure as 3D EZBC. Nevertheless, unlike the 3D EZBC context models which are built for the quadtree nodes from different subbands and quadtree levels [10], 3D SPEZBC build the independent context models for all sets within individual subbands and set partitioning levels. And it effectively employs two statistical dependencies — the intra-band correlation among sets at the same set partitioning level within subband and the inter-band correlation among set across subbands. For entropy coding of significance testing bitstreams, the 3D SPEZBC context models registers the significance testing status of each set, and the context of every set $S_{k,b}[l]$ is located as node of the

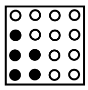
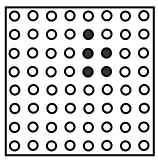
pyramidal context models at the spectral band b, subband k and set partitioning level l, as illustrated in Figure 4(c). Moreover, the sign coding employs the similar scheme of JPEG 2000, namely the output sign bitstream of the significant coefficient is coded according to its sign and significance status of its eight neighboring pixels. Finally, the entropy coding of refinement bitstreams utilizes the same context models of the significance map coding. The details about the context models and look-up tables can refer to Reference 10.

The significance testing function of the set $S_{k,b}[l]$ against a certain threshold 2^n is defined as follows:

$$\Gamma_n(S_{k,b}[l]) = \begin{cases} 1, & \text{if } 2^n \leq \max\limits_{(i,j,b)\in S_{k,b}[l]} |c(i,j,b)| < 2^{n+1} \\ 0, & \text{else} \end{cases}$$

where $c(i, j, b)$ denotes the transformed wavelet coefficient at the coordinate (i, j, b)

Table 1. The relationship of code block, set and set partitioning level

Code block	Set	Set size	Set partitionin level l
	$S_{k,b}[0]$	1 x 1	0
	$S_{k,b}[1]$	2 x 2	1
	$S_{k,b}[2]$	4 x 4	2
	$S_{k,b}[3]$	8 x 8	3

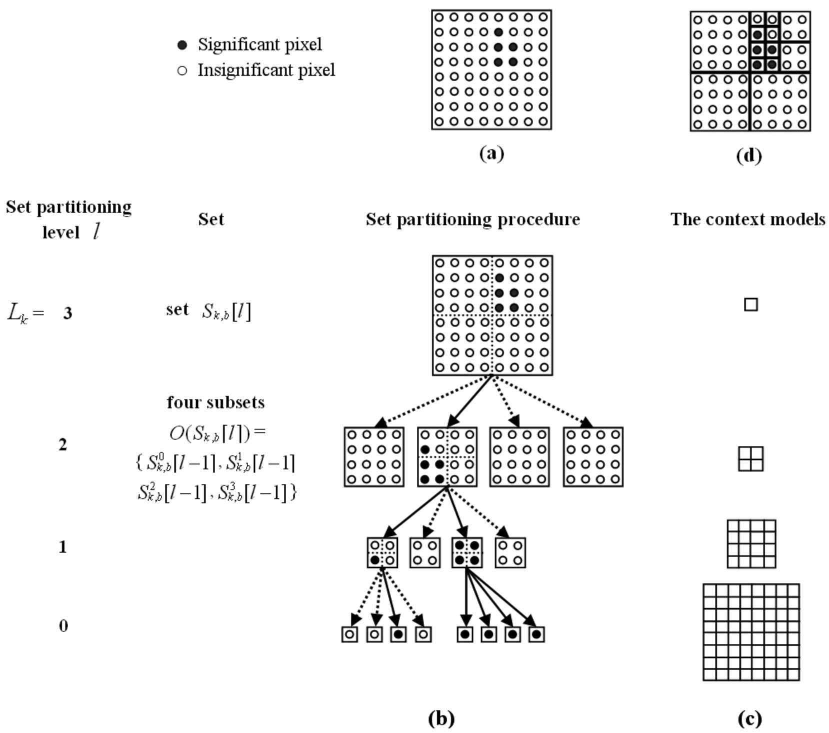


Figure 4. Illustration of the set-based quadtree partitioning procedure and the context models of the 3D SPEZBC algorithm. (a) The original k^{th} subband of b^{th} spectral band. Initially, two arrays of lists (LIS_k and LSP_k) is maintained for subband k. (b) The set-based partitioning procedure. (c) The context models for subband k. (d) The Partitioning result for subband k of spectral band b

3.1 Initialization

- Output $n = \left\lfloor \log_2\{\max_{(i,j,b)} |c(i,j,b)|\} \right\rfloor$
- for $k = 0 : K-1$
  - Set $LIS_k[l] = \begin{cases} \{S_{k,b}[l], \text{namely } k^{th} \text{ subband}\}, & l = L_k \\ \phi, & \text{otherwise} \end{cases}$
  - Set $LSP_k = \phi$.

3.2 Sorting Pass

for $l = 0 : L_{\max}$
  for $k = 0 : K-1$
    CodeLIS (k, l);

CodeLIS (k, l)
{
For each set $S_{k,b}[l] \in LIS_k[l]$,
- Output $\Gamma_n(S_{k,b}[l])$;
- if $\Gamma_n(S_{k,b}[l]) = 1$
  - if l = 0 (namely the set $S_{k,b}[l]$ is a pixel)
    Output sign of $S_{k,b}[l]$, and remove $S_{k,b}[l]$ to LSP_k;
    else
    CodeSubSets($S_{k,b}[l]$), and remove $S_{k,b}[l]$ from $LIS_k[l]$.

}

CodeSubSets ($S_{k,b}[l]$)
{
- Partition $S_{k,b}[l]$ into four approximately equal sizes of subsets O ($S_{k,b}[l]$) ={ $S^0_{k,b}[l-1]$, $S^1_{k,b}[l-1]$, $S^2_{k,b}[l-1]$, $S^3_{k,b}[l-1]$ }, where $S'_{k,b}[l-1] \in O(S_{k,b}[l])$.
- For each $S'_{k,b}[l-1]$
  - Output $\Gamma_n(S'_{k,b}[l-1])$;
  - if $\Gamma_n(S'_{k,b}[l-1]) = 1$
    - if l = 1 (namely the set $S'_{k,b}[l-1]$ is a pixel)
      Output sign of $S'_{k,b}[l-1]$, and add $S'_{k,b}[l-1]$ to LSP_k;
      else
      CodeSubSets ($S'_{k,b}[l-1]$).

    else
    add $S'_{k,b}[l-1]$ to $LIS_k[l-1]$.

}

3.3 Refinement Pass

for $k = 0 : K-1$
  CodeLSP (k);

  CodeLSP (k)
{
- For each pixel set $S_{k,b}[l] \in LSP_k$ which correspond to pixel $c(i, j, b)$, output the n^{th} MSB of | $c(i, j, b)$|except those included in the last sorting pass.

}

3.4 Quantization Step

Decrement n by 1 and go to step 2.

4. Discussion

The difference between two partitioning mechanism, the partitioning zeroblock coding based on set representation structure in 3D SPEZBC and the partitioning zeroblock coding based on quadtree representation structure in 3D EZBC, are only the different representation structures and testing mode for splitting the code block, but their splitting and coding results are same to each other. Moreover, our experimental results also show that the compression performances of 3D SPEZBC and 3D EZBC are totally the same. However, 3D SPEZBC saves considerable memory requirement in comparison with 3D EZBC due to the fact that the quadtree representation structure can be eliminated. For the k^{th} subband of size $M \times M$, the quadtree depth D_k of 3D EZBC is equal to $\log_2 M$. If the transformed wavelet coefficients are stored as the binary floating-point numbers (4 bytes), its quad-tree representation structure needs to be allotted $\sum_{i=0}^{D_k}(\frac{1}{4})^i \times M \times M \times sizeof(float) = \sum_{i=0}^{D_k}(\frac{1}{4})^i \times M^2 \times 4 \quad (bytes)$ memory usage. The quadtree nodes at the bottom quad-tree level 0 (namely i = 0) consist of the magnitudes of the wavelet coefficients in subband, so it can't be deleted. Therefore, when using the 3D SPEZBC algorithm, $\sum_{i=1}^{L_k}(\frac{1}{4})^i \times M^2 \times 4 \quad (bytes)$ memory can be saved for this subband.

If four spatial levels and four spectral levels 3D WPT is employed for the hyperspectral image of size 512×512×224, the pyramidal wavelet structure has 224 bands, and each band has 4 subbands of size 32×32, 3 subbands of size 64×64, 3 subbands of size 128×128, and 3 sub-bands of size 256×256. So the saved memory space in our algorithm against 3D EZBC is computed as

$$224 \times [4\times\sum_{i=1}^{5}(\frac{1}{4})^i \times 32^2 \times 4 + 3\times\sum_{i=1}^{6}(\frac{1}{4})^i \times 64^2 \times 4$$

$$+ \ 3\times\sum_{i=1}^{7}(\frac{1}{4})^i \times 128^2 \times 4 \ + \ 3\times\sum_{i=1}^{8}(\frac{1}{4})^i \times 256^2 \times 4] \quad (bytes)$$

$$\approx \ 79185344 \quad (bytes) \ \approx \ 75.52 \quad (Mbytes)$$

The quadtree structure of 3D EZBC provides a fast quadtree splitting scheme by reducing the number of significance test. Nevertheless, the set-based partitioning zeroblock coding method of 3D SPEZBC is very simple, and its 2D code sets are smaller and are processed according to the increasing order of set size using the multi-list structure at the particular set partitioning level, as well as the 3D SPEZBC algorithm does not need time to establish quadtree structure, so it also exhibits excellent speed performance that is slightly worse than 3D EZBC. When our coder compresses four 512×512×224 AVIRIS hyperspectral images (such as Cuprite, Jasper Ridge, Low Altitude and Lunar Lake) on a AMD Athlon 3800+ CPU 2GHz machine, there are averagely 87.36 s for encoding and 125.23 s for decoding at 1.0 bpppb, 105.88 s for encoding and 170.74 s for decoding at 2.0 bpppb, as well as 135.10 s for encoding and 209.42 s for decoding at 3.0 bpppb, respectively.

5. Experimental Results

We performed coding experiments on four signed 16-bit radiance AVIRIS hyperspectral images [16], namely Cuprite scene 1, Jasper Ridge scene 1, Low Altitude scene 1 and Lunar Lake scene 1. In our experiments, we extracted the 256×256 lower left corner, so that the dimensions of the test image were 256×256×224 pixels. For lossy compression the rate distortion performance was compared by means of the signal-to-noise (SNR) values for a variety of bit rates in bits per pixel per band (bpppb), and for lossless compression performance we used those rates to evaluate the size of the compressed data streams. SNR is defined as $10\log_{10}\frac{\sigma_x^2}{MSE}$, where σ_x^2 is the average squared value (power) of the original AVIRIS image and MSE is the mean squared error over the entire sequence.

5.1 Lossless Compression Performance

Table 2 presents the lossless compression results for the 3D SPEZBC algorithm using various integer wavelet filters, which adopts the four spatial levels and four spectral levels Xiong's 3D integer WPT with Figure 2's unitary scaling factor. We can see that no certain wavelet filter is optimal for all test images. The 5/11-A, 13/7-C and 5/3 integer filters all provide good compression performances. Furthermore, Adams *et al.* [14] have found that the 5/3 filter evidently required the least computation, and experimental results in Subsection 5.2 further show that the integer-based lossy compression performance using the 5/3 integer filter clearly outperform that using the 5/11-A and 13/7-C integer filters at the medium and high bit rates. Therefore, Figure 5 displays the lossless compression ratios using the 5/3 integer filter in comparison with several state-of-the-art wavelet-based algorithms. In our experiments, JPEG2000-MC used the four spatial levels and four spectral levels 3D integer WPT and other algorithms used the four spatial levels and four spectral levels Xiong's 3D integer WPT with the unitary scaling factor in Figure 2. For all test images, the results show that 3D SPEZBC outperforms 3D SPECK, 3D SPIHT and AT-3D SPIHT, and it is worse than JPEG2000-MC. The average compression ratio of 3D SPEZBC is 5.70 % lower than 3D SPECK, 7.14 % lower than 3D SPIHT, 4.96 % lower than AT-3D SPIHT, and 1.07 % higher than JPEG2000-MC.

5.2 Integer-Based Lossy Compression Performance

The integer-based lossy compression results can be obtained when the decoder truncates the lossless encoded bitstreams in Subsection 5.1 at a desired bit rate. If we decode the hyperspectral image without losing any information, it is perfectly reconstructed. Table 3 shows the rate distortion results of the 3D SPEZBC algorithm using various integer wavelet filters for "Cuprite" image. We can see that these wavelet filters exhibit different coding performance at various bit rates. The 5/3 integer filter requires the least computation proved by Adams *et al.* [14] and provides excellent compression performance at medium and high bit rates. Moreover, when we apply the ISODATA and K-means unsupervised classification methods in comparison further (in Subsection 5.3), at more than 1.0 bpppb (16:1 compression ratio) the classification accuracy is higher than 99%. The experimental results on "Jasper Ridge", "Low Altitude" and "Lunar Lake"

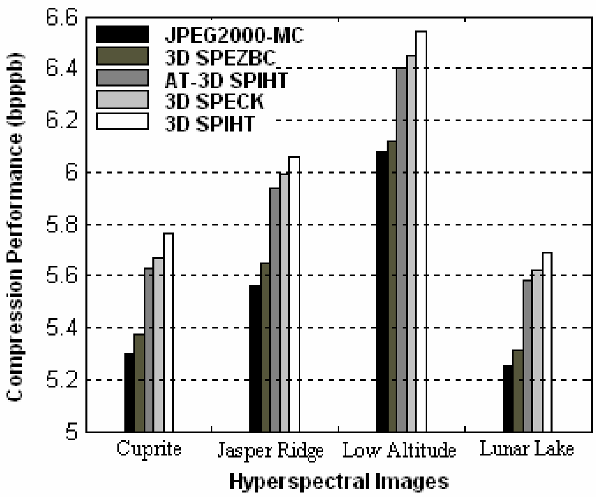


Figure 5. Lossless compression results (bpppb) in comparison with the state-of-the-art wavelet-based algorithms using 5/3 integer filter

Table 2. Lossless compression results (bpppb) for 3D SPEZBC using the various wavelet filters

Wavelet	Commpression Performance (bpppb)				
	Cuprite	Jasper Ridge	Low Altitude	Lunar Lake	Average
	5.44	5.66	6.17	5.39	5.67
(2+2,2)	5.39	5.63	6.11	5.35	5.62
(2,4)	5.38	5.67	6.14	5.32	5.63
(6,2)	5.42	5.66	6.14	5.38	5.65
5/3	5.37	5.65	6.12	**5.31**	5.61
2/6	5.42	5.69	6.21	5.36	5.67
2/10	5.46	5.70	6.21	5.40	5.69
9/7-M	5.39	5.63	6.11	5.35	5.62
9/7-F	5.40	5.67	6.12	5.34	5.63
5/11-A	5.37	**5.62**	**6.09**	5.32	**5.60**
13/7-C	**5.36**	5.63	6.10	**5.31**	**5.60**

Table 3. Integer-based lossy compression performance (SNR, in dB) in comparison with the various integer-based wavelet filters for the 3D SPEZBC algorithm

hyperspectral Image	Wavelet	Bit Rates (bpppb)								
		0.1	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
Cuprite	**S+P (B)**	39.33	47.95	51.95	55.05	56.69	59.01	60.38	62.27	63.17
	(2+2,2)	40.79	49.01	52.99	55.62	57.37	59.65	61.22	63.14	64.09
	(2,4)	40.50	48.90	53.13	55.99	57.94	**60.21**	61.75	63.60	64.49
	(6,2)	40.82	48.78	52.75	55.39	57.05	59.29	60.70	62.52	63.43
	5/3	40.71	49.19	**53.28**	**56.00**	**57.95**	**60.21**	**61.81**	**63.68**	**64.64**
	2/6	39.40	48.22	52.22	55.40	57.26	59.58	60.98	62.80	63.65
	2/10	39.28	47.95	51.82	55.03	56.84	59.09	60.57	62.43	63.36
	9/7-M	40.92	48.94	52.95	55.61	57.32	59.60	61.02	62.87	63.74
	9/7-F	40.55	48.96	52.90	55.55	57.26	58.73	59.72	60.84	61.37
	5/11-A	40.84	49.13	53.17	55.83	57.69	59.98	61.55	63.47	64.38
	13/7-C	**41.04**	**49.28**	**53.28**	55.96	57.74	59.93	61.23	63.05	63.75

hyperspectral images demonstrate the similar conclusions. Taking into these causes consideration, we think that the 5/3 integer filter is a very good choice for hyperspectral image integer-based lossy compression by using the 3D SPEZBC algorithm. For four hyperspectral images, Table 4 shows that the rate-distortion performance of the proposed algorithm is better than several state-of-the-art wavelet-based coding algorithms by using the 5/3 integer filter. We can see that the 3D SPEZBC algorithm outperforms the 3D SPECK, 3D SPIHT, AT-3D SPIHT and JPEG2000-MC algorithms at various bit rates. For all of the four hyperspectral images at 2 bpppb (8:1 compression ratio), 3D SPEZBC averagely overcomes 3D SPECK by 0.76 dB, 3D SPIHT by 1.22 dB, AT-3D SPIHT by 0.40 dB and JPEG2000-MC by 0.18 dB, respectively.

5.3 Classification Performance Comparison

In order to measure the influence of the aforementioned compression algorithms on the application performance of the reconstructed hyperspectral images, we applied the ISODATA and K-means unsupervised classification methods for comparison further, where we set the maximal number as ten classes and the maximal iterations as three. For the hyperspectral images, Table 5 gives the results of ISODATA and k-means unsupervised classification. The accuracy of the classification on 3D SPEZBC outperforms those of 3D SPECK, 3D SPIHT and AT-3D SPIHT, and is very close to those of JPEG2000-MC. For 3D SPEZBC at 1.0 bpppb (16:1 compression ratio), the classification accuracy is higher than 99%.

6. Conclusions

In this paper, we propose the 3D SPEZBC algorithm for hyperspectral image lossy-to-lossless compression, which is an improved 3D EZBC algorithm. It adopts the partitioning coding technique based on the set representation structure so as to avoid the problem with higher memory requirements for establishing the quadtree representation structure. According to the theoretical and experimental

Table 4. Integer-based lossy compression performance (SNR, in dB) in comparison with the state-of-the-art wavelet-based coding algorithms using 5/3 integer filter

hyperspectral Image	Coding Methods	Bit Rate (bpppb)						
		0.1	0.5	1.0	1.5	2.0	2.5	3.0
Cuprite	3D SPEZBC	**40.71**	**49.19**	**53.27**	**56.00**	**57.95**	**60.21**	**61.81**
	3D SPECK	39.92	48.62	52.61	55.49	57.57	59.71	61.45
	3D SPIHT	38.59	47.79	51.79	54.96	57.35	59.37	61.15
	AT-3D SPIHT	40.23	48.93	52.89	55.73	57.80	59.94	61.66
	JPEG2000-MC	40.56	49.02	53.18	55.85	57.89	60.01	61.53
Jasper Ridge	3D SPEZBC	**30.74**	**41.03**	**46.35**	**50.10**	**52.91**	**55.08**	**57.00**
	3D SPECK	30.11	40.13	45.50	49.25	52.18	54.27	56.39
	3D SPIHT	29.07	39.42	45.14	48.91	51.60	53.86	56.03
	AT-3D SPIHT	30.26	40.65	46.05	49.49	52.52	54.51	56.64
	JPEG2000-MC	30.62	40.89	46.14	49.78	52.81	54.65	56.58
Low Altitude	3D SPEZBC	**27.33**	**37.94**	**44.33**	**48.38**	**51.11**	**53.52**	**55.68**
	3D SPECK	26.74	37.05	43.28	47.42	50.12	52.78	54.69
	3D SPIHT	25.84	36.56	42.57	46.82	49.82	52.37	54.40
	AT-3D SPIHT	26.76	37.59	43.81	47.82	50.68	53.21	55.10
	JPEG2000-MC	27.18	37.72	44.06	47.99	50.75	53.15	54.91
Lunar Lake	3D SPEZBC	**43.20**	**50.88**	**54.76**	**57.33**	**59.25**	**61.35**	**62.94**
	3D SPECK	42.41	50.44	54.30	56.94	58.77	60.92	62.73
	3D SPIHT	41.11	49.70	53.49	56.31	58.58	60.55	62.39
	AT-3D SPIHT	42.59	50.64	54.51	57.14	58.94	61.08	62.81
	JPEG2000-MC	43.02	50.76	54.68	57.20	59.07	60.98	62.69

Table 5. Overall classification accuracy comparison (in %) based on ISODATA and K_mean at the various bit rates (bpppb) for lossy compression based on integer wavelet transform

hyperspectral Image	Coding Methods	K_mean					ISODATA				
		0.1	0.5	1.0	1.5	2.0	0.1	0.5	1.0	1.5	2.0
Cuprite	3D SPEZBC	**90.88**	**99.07**	**99.53**	**99.72**	**99.79**	**92.97**	**99.28**	**99.62**	**99.76**	**99.78**
	3D SPECK	90.58	99.03	99.49	99.70	99.78	92.72	99.20	99.58	99.75	99.77
	3D SPIHT	87.63	98.83	99.45	99.69	99.75	90.30	99.13	99.56	99.74	99.76
	AT-3D SPIHT	90.83	99.04	99.50	99.71	**99.79**	92.90	99.25	99.59	99.75	99.77
	JPEG2000-MC	90.86	99.06	99.52	99.71	**99.79**	92.93	99.27	99.61	**99.76**	99.77
Jasper Ridge	3D SPEZBC	**89.74**	**98.77**	**99.48**	**99.68**	**99.71**	**92.23**	**99.08**	**99.72**	**99.80**	**99.86**
	3D SPECK	89.63	98.70	99.43	99.53	99.66	92.07	98.97	99.65	99.75	99.84
	3D SPIHT	87.53	98.58	99.27	99.51	99.65	90.38	98.91	99.54	99.76	99.83
	AT-3D SPIHT	89.69	98.73	99.45	99.65	99.68	92.10	98.99	99.69	99.79	99.84
	JPEG2000-MC	89.72	98.74	99.46	99.66	**99.71**	92.20	99.04	99.70	**99.80**	99.85

analysis, our algorithm not only provides the same excellent compression performance as 3D EZBC, but also can save considerable memory requirements against 3D EZBC. For hyperspectral image lossy-to-lossless compression based on 3D SPEZBC, Xiong's 3D integer WPT with unitary scaling factor in Figure 2 and the 5/3 integer filter are good options. Compared with several state-of-the-art wavelet-based coding algorithms, the experimental results indicate that our coder provides better compression performance and unsupervised classification accuracy.

7. Acknowledgments

This work is supported by the Engagement Foundation of Xi'an University of Science and Technology (Project No.200722).

REFERENCES

[1] X. Tang and W. A. Pearlman, "Three-dimensional wavelet-based compression of hyperspectral images," Hyperspectral Data Compression, MA: Kluwer Academic Publishers, pp. 273−308, 2006.

[2] J. E. Fowler and J. T. Rucker, "3D wavelet-based compression of hyperspectral imagery," Hyperspectral Data Exploitation: Theory and Applications, John Wiley & Sons Inc., Hoboken, NJ, pp. 379−407, 2007.

[3] B. Penna, T. Tillo, E. Magli, and G. Olmo, "Transform coding techniques for lossy hyperspectral data compression," in the Proceedings of IEEE Transactions on Geoscience and Remote Sensing, Vol. 45, No. 5, pp. 1408 −1421, 2007.

[4] T. W. Fry and S. Hauck, "Hyperspectral image compression on reconfigurable platforms," in the Proceedings of IEEE Symposium on Field-Programmable Custom Computing Machines, pp. 251−260, April 2002.

[5] X. Tang, S. Cho, and W. A. Pearlman, "3D set partitioning coding methods in hyperspectral image compression," in the Proceedings of IEEE International Conference on Image Processing, pp. 239−242, September 2003.

[6] X. Tang and W. A. Pearlman, "Lossy-to-lossless block-based compression of hyperspectral volumetric data," in the Proceedings of IEEE International Conference on Image Processing, pp. 1133−1136, 2006.

[7] J. J. Wu, Z. S. Wu, and C. K. Wu, "Lossy to lossless compressions of hyperspectral images using three-dimensional set partitioning algorithm," SPIE Optical Engineering, Vol. 45, No. 2, pp. 0270051−0270058, 2006.

[8] B. Penna, T. Tillo, E. Magli, and G. Olmo, "Embedded lossy-to-lossless compression of hyperspectral images using JPEG 2000," in the Proceedings of IEEE International Geoscience and Remote Sensing Symposium, Vol. 1, pp. 25−29, 2005.

[9] J. Zhang, J. E. Fowler, and G. Z. Liu, "Lossy-to-lossless compression of hyperspectral imagery using 3D-TCE and an integer KLT," IEEE Geoscience and Remote Sensing Letters, Vol. 4, No. 2, pp. 201−205, 2008.

[10] S. T. Hsiang, "Highly scalable subband/wavelet image and video coding," Ph.D dissertation, Rensselaer Polytechnic Institute, Troy, 2002.

[11] A. Islam and W. A. Pearlman, "An embedded and efficient low-complexity hierarchical image coder," in the Proceedings of SPIE Conference on Visual Communications and Image Processing, Vol. 3653, pp. 294−305, 1999.

[12] Y. Hou and G. Z. Liu, "3D set partitioned embedded zero block coding algorithm for hyperspectral image compression," in the Proceedings of SPIE Symposium on MIPPR, Vol. 6790, pp. 561−567, 2007.

[13] A. R. Calderbank, I. Daubechies, W. Sweldens, and B. L. Yeo, "Wavelet transforms that map integers to integers," Applied and Computational Harmonic Analysis, Vol. 5, No. 3, pp. 332−369, 1998.

[14] M. D. Adams and F. Kossentini, "Reversible integer-to-integer wavelet transforms for image compression: Performance evaluation and analysis," IEEE Transactions on Image Processing, Vol. 9, No. 6, pp. 1010−1024, 2000.

[15] Z. X. Xiong, X. L. Wu, S. Cheng, and J. P. Hua, "Lossy-to-lossless compression of medical volumetric data using three-dimensional integer wavelet transforms," IEEE Transactions on Medical Imaging, Vol. 22, No. 3, pp. 459−470, 2003.

[16] http://aviris.jpl.nasa.gov/html/aviris.overview.html.

