Lossy to lossless compressions of hyperspectral images using three-dimensional set partitioning algorithm

Jiaji Wu[a], Zhensen Wu and Chengke Wu
School of Science, Xidian University, Xi'an, 710071, P.R.China

ABSTRACT

In this paper, we present a three-dimensional (3D) hyperspectral image compression algorithm based on zeroblock coding and wavelet transforms. An efficient Asymmetric 3D wavelet Transform (AT) based on the lifting technique and packet transform is used to reduce redundancies in both the spectral and spatial dimensions. The implementation via 3D integer lifting scheme allows to map integer-to-integer values, enabling lossy and lossless decompression from the same bit stream. To encode these coefficients after Asymmetric 3D wavelet transform, a modified 3DSPECK algorithm – Asymmetric Transform 3D Set Partitioning Embedded bloCK (AT-3DSPECK) is proposed. According to the distribution of energy of the transformed coefficients, the 3DSPECK's 3D set partitioning block algorithm and the 3D octave band partitioning scheme are efficiently combined in the proposed AT-3DSPECK algorithm. Several AVIRIS images are used to evaluate the compression performance. Compared with the JPEG2000, AT-3DSPIHT and 3DSPECK lossless compression techniques, the AT-3DSPECK achieves the best lossless performance. In lossy mode, the AT-3DSPECK algorithm outperforms AT-3DSPIHT and 3DSPECK at all rates. Besides the high compression performance, AT-3DSPECK supports progressive transmission. Clearly, the proposed AT-3DSPECK algorithm is a better candidate than several conventional methods.

Keywords: hyperspectral images,3D image compression, wavelet transform,3DSPECK, zeroblock coding

1. INTRODUCTION

Hyperspectral images are of interest for a large number of applications, including environmental monitoring, geology, and meteorology. This is because hyperspectral imagery provides the potential for more accurate and detailed information extraction than possible with any other type of remotely sensed data. However, hyperspectral images result in large sized data sets. The storage and transmission of large volumes of hyperspectral data have become significant concerns. Therefore efficient compression is required for storage and transmission.

Hyperspectral images include both spatial and spectral redundancies. Most popular image coding algorithms attempt to transform the image data so that the transformed coefficients are largely uncorrelated. Then these coefficients can be quantized and coded. In many applications, the 3D transform employs the Karhunen-Loeve transform (KLT) to decorrelate spectral redundancies while 2D wavelet transform is used to decorrelate spatial redundancies[1]. The well-known Karhunen-Loeve transform (KLT) is the optimal linear transformation for decorrelating the data. Unfortunately, KLT is too computationally complex, and also KLT can rarely achieve lossless compression.

The 3D wavelet transform (3DWT) has a much lower computational complexity than the KLT/WT. Therefore many 3D image coding methods based on 3DWT are proposed in [2-7]. The classical 3D wavelet image coding algorithm is 3DSPIHT proposed by Kim *et al.* [2]. It is an extension of original 2DSPIHT[8] and has a 3D tree structure. Tang *et al.* [3] extended 2DSPECK[9] to 3DSPECK for hyperspectral image compression. In [3] the transformed coefficients are partitioned into several three-dimensional zeroblocks with different sizes. Initially, each wavelet subband is treated as a zeroblock. Next, when the zeroblock is significant against the threshold n, it is split into several smaller subblocks. In general, 3DSPECK is very close to 3DSPIHT considering rate-distortion performance. However, 3DSPECK has been found to be better performance in hyperspectral images with high spatial frequency content and complex texture[3].

[a] E-mail: luckydog73@sina.com (Jiaji Wu), wuzhs@mail.xidian.edu.cn (Zhensen Wu), ckwu@ns2.xidian.edu.cn (Chengke Wu).

Electronic Imaging and Multimedia Technology IV, edited by Chung-Sheng Li, Minerva M. Yeung, Proc. of SPIE Vol. 5637 (SPIE, Bellingham, WA, 2005)
0277-786X/05/$15 · doi: 10.1117/12.575437

Table 1. Lossless integer lifting filters for forward transform

Filter name	Lifting Steps
5x3	$d[n] = d_0[n] - \lfloor 1/2(s_0[n] + s_0[n+1]) + 1/2 \rfloor$ $s[n] = s_0[n] + \lfloor 1/4(d[n-1] + d[n]) + 1/2 \rfloor$
5x11	$d_1[n] = d_0[n] - \lfloor 1/2(s_0[n] + s_0[n+1]) + 1/2 \rfloor$ $s[n] = s_0[n] + \lfloor 1/4(d_1[n-1] + d_1[n]) + 1/2 \rfloor$ $d[n] = d_1[n] - \lfloor 1/16(-s[n-1] + s[n] + s[n+1] - s[n+2]) + 1/2 \rfloor$
9x3	$d[n] = d_0[n] - \lfloor 1/2(s_0[n] + s_0[n+1]) + 1/2 \rfloor$ $s[n] = s_0[n] + \lfloor 19/64(d[n-1] + d[n]) - 3/64(d[n-2] + d[n+1]) + 1/2 \rfloor$
9x7	$d[n] = d_0[n] - \lfloor 9/16(s_0[n] + s_0[n+1]) - 1/16(s_0[n-1] + s_0[n+2]) + 1/2 \rfloor$ $s[n] = s_0[n] + \lfloor 1/4(d[n-1] + d[n]) + 1/2 \rfloor$
13x11	$d[n] = d_0[n] - \lfloor 75/128(s_0[n] + s_0[n+1]) - 25/256(s_0[n-1] + s_0[n+2]) + 3/256(s_0[n-2] + s_0[n+3]) + 1/2 \rfloor$ $s[n] = s_0[n] + \lfloor 1/4(d[n-1] + d[n]) + 1/2 \rfloor$

Recently a new asymmetric 3DWT method has been proposed. For 3D image data compression, the asymmetric 3DWT is more efficient than the symmetric 3DWT. The new asymmetric transform method differs from symmetric transform mainly in a wavelet packet transform applied spectrally. At-3DSPIHT[5], 3D-ESCOT[6] and optimal 3D coefficient tree structure[4] based on the asymmetric 3DWT are presented. They exhibited a better performance than conventional algorithms based on symmetric 3DWT.

In this paper, a progressive lossy-to-lossless hyperspectral image compression technique is developed. For most remote sensing applications, it is unwise to discard any information that may be useful later, thus the original quality of the data must be thoroughly preserved after compression/decompression. In order to obtain progressive lossy-to-lossless compression with a single embedded bitstream, we apply asymmetric 3DWT based on integer lifting scheme to eliminate much of the redundancy. Finally these transformed coefficients are quantized and encoded. Considering the distribution of the coefficients of the subbands, we apply 3DSPECK's 3D set partitioning block algorithm and the 3D octave partitioning scheme to code the transformed coefficients, where 3D octave partitioning is a modification and extension of 2DSPECK's octave partitioning.

The rest of this paper is organized as following: in section 2, after several common reversible integer wavelet transforms based on lifting scheme are reviewed, the asymmetric 3DWT and the AT-3DSPECK are described in detail. Experiments and results are provided in section 3, in which the performances of AT-3DSPECK, AT-3DSPIHT, 3DSPECK, JPEG2000, 2DSPECK and 2DSPIHT are compared for hyperspectral image compression. Finally, section 4 concludes this paper.

2. THREE-DIMENSIONAL WAVELET TRANSFORM

2.1. Lifting-based wavelet transform

Wavelet transforms can be implemented using finite impulse response filter banks. Wim Sweldens[11] introduced the lifting scheme allowing to compute the discrete wavelet transform with a reduced computational complexity. Any discrete wavelet transform or two band subband filtering with finite filters can be decomposed into a finite sequence of simple filtering steps, which we call lifting steps[12], thus any integer wavelet transform can be achieved using lifting scheme. The lifting algorithm can be described in three steps: lazy wavelet transform, predication and update. Let $x[n]$ be the input signal. Then "the lazy wavelet transform" is given by

$$s_0[n] = x[2n], \tag{1}$$

$$d_0[n] = x[2n+1]. \tag{2}$$

Next, alternating "predication" and "update" steps are applied using

$$d_i[n] = d_{i-1}[n] - \left\lfloor \left(\sum_k p_i[k] s_{i-1}[n-k] \right) + 1/2 \right\rfloor, \tag{3}$$

$$s_i[n] = s_{i-1}[n] + \left\lfloor \left(\sum_k u_i[k] d_i[n-k] \right) + 1/2 \right\rfloor, \tag{4}$$

for $i = 0, \cdots, M-1$. Finally, scaling factors are applied to get the wavelet coefficients:

$$s[n] = s_M[n]/k, \tag{5}$$

$$d[n] = k d_M[n], \tag{6}$$

where $s[n]$ and $d[n]$ are, respectively, the low-pass and the high-pass wavelet coefficients and $k = 1$ is the corresponding scaling factors in our work. Table 1 shows several integer filters.

2.2. Three-Dimensional integer Wavelet Transform

A commonly used 3DWT is the symmetric 3DWT which has been used in 3DSPIHT[2] and 3DSPECK[3]. It is performed by three separate 1D dyadic wavelet transforms along the spectral directions, the rows and the columns of the image data. However, many experiments show that 3D hyperspectral images (or 3D video sequences) do not have symmetric statistical properties along all the directions[4], [5]. The average standard deviation of the spectral dimension is much smaller than the average standard deviations of other two spatial dimensions. The asymmetric transform just solves these asymmetric statistics, and can efficiently reduce the redundancy between spectral bands. The 3D asymmetric wavelet transform has been applied in [4-7] for 3D image compression. Fig.1 illustrates a 3D asymmetric wavelet structure which has three levels of decomposition. Three levels of dyadic wavelet decomposition are applied in the spatial domain, followed by three levels of wavelet decomposition on the spectral domain.

Usually, the non-unitary transform does not affect the performance of lossless compression. However, this will decrease the performance of lossy compression. If the transform is unitary, the distortion in the transform domain can be directly related to the distortion in the pixel domain. Thus, when the non-unitary transform is employed, appropriate scaling of the subband coefficients is necessary to correct the mismatch of distortion estimation between the pixel domain and transform domain. In our study, we follow a simple bit shifting of wavelet coefficients proposed by Xiong[6] to make the transform unitary. Fig.2 gives a 3-level asymmetric wavelet transform structure with the subband scaling. Although [6] applies wavelet packet technology on the spectral dimension to approximate a unitary transformation and to improve the performance of 3D image compression, but our experiments show that the applied wavelet packet on the high-frequency subbands of spectral dimension of the

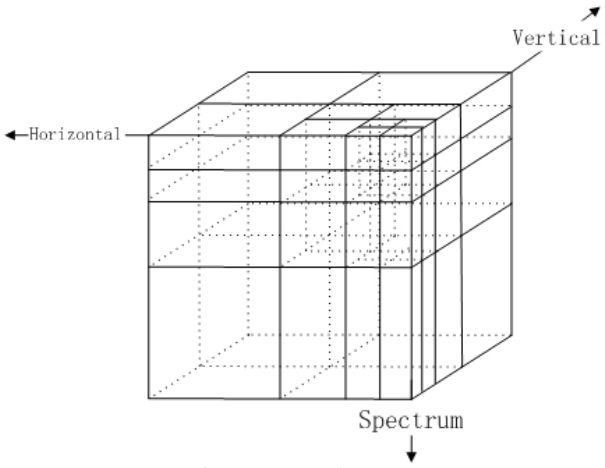


Fig. 1. Asymmetric 3DWT structure

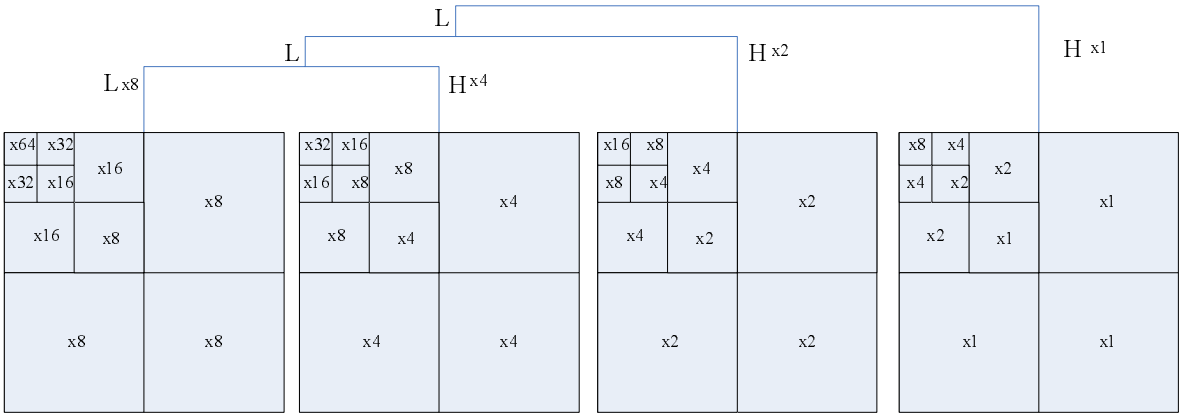


Fig. 2. The subband scaling parameters used in Asymmetric 3DWT structure

hyperspectral images will decrease the performance for lossy compression. In [6], right shift will cause the loss of the precision on the highest-frequency subband. In our case, all coefficients on the highest-frequency subband will be not shifted.

2.3. Asymmetric Transform 3D Set Partitioning Embedded block (AT-3DSPECK)

In the zerotree algorithm[8], once zerotree tested significant, it is partitioned into several sub-zerotrees. If the zerotree is not significant, outputs "0" to indicate this zerotree. The AT-3DSPECK is different from the 3D zerotree algorithm mainly in its treatment of the zeroblock (a large region of insignificant coefficients). If all of coefficients are not significant in a block, the block is a zeroblock[10]. Whenever a given set in 2DSPECK[9] tests significant against the threshold n for 2D image, it is partitioned into four approximately equal subsets. 3DSPECK is a modification of 2DSPECK, where each set tested significant is partitioned into eight/four approximately equal subsets for 3D images. As in traditional 3DSPECK approaches, l levels of dyadic wavelet decomposition are applied along all directions, the transformed images will have $7l+1$ 3D subbands. When the 3DSPECK algorithm starts, each 3D subband is defined as a zeroblock and is added to the LIS – List of Insignificant Sets. We treat each of these subsets as type S set, and test their significance. If any set of type S is significant, this set is partitioned into eight/four subsets with a smaller size. As we see, in asymmetric way, 2D dyadic wavelet decomposition in the spatial domain and 1D cascaded wavelet packet decomposition on the spectral domain form more 3D subbands. After asymmetric 3D wavelet decomposition, there are $(3l+1)(z+1)$ 3D subbands, and l is the number of decomposition levels in the spatial domain and z is the number of decomposition levels on the spectral domain. Our experiments show that the original 3DSPECK applied to code the coefficients after the asymmetric wavelet transform is not the best for hyperspectral images. Obviously, many transform coefficients are zero or low amplitude in the high frequency subbands (finer subbands) where the probability of finding significant coefficients against a big threshold (the high bit plane) is very little. When the 3DSPECK algorithm starts, several finer subbands which have low energy are treated as insignificant sets in LIS, and they will repeatedly output "0" against the first several bit planes. That means the more level of wavelet decomposition and insignificant set in the LIS, the more "0" bit will be generated against the first several bit planes. An important problem is how to reduce the output produced by these obviously insignificant finer subbands. We consider that whenever possible, these finer subbands should be indicated to use few sets.

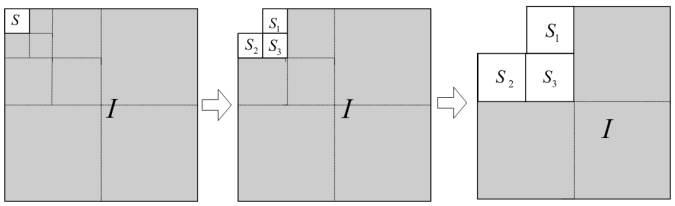


Fig. 3. Partitioning of set I in the spatial domain

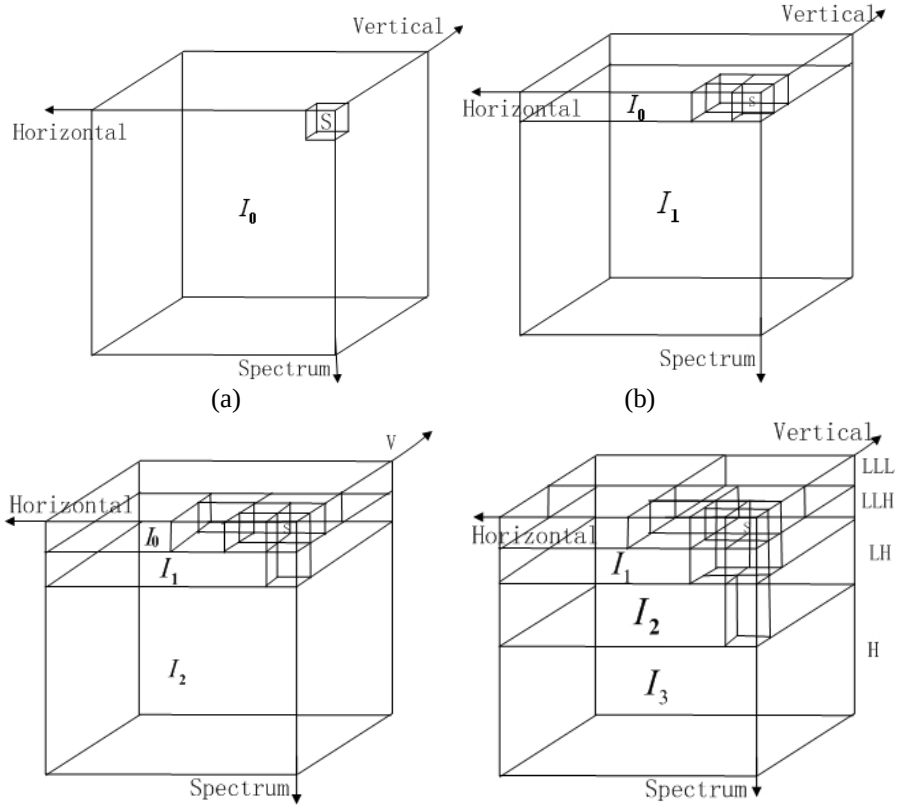


Fig. 4. Illustration of 3D octave band partitioning of AT-3DSPECK

In the 2DSPECK algorithm, the octave band partitioning is proposed. Fig. 3 gives an illustration of this partitioning scheme in the spatial domain. Because energy is concentrated at the topmost levels of wavelet decomposition, the transformed image is partitioned into several sets of type S and one set of type I. If a set of I is significant against some threshold n, it is more likely that the pixels which cause I to be significant lie in the top left regions of I [9], as is shown in Fig. 3. These regions are decomposed into three sets of type S and one set of type I. The main goal of the octave band partitioning is to use one set of type I to represent these finer subbands which have a little probability of insignificant.

We extend the octave band partitioning of 2DSPECK to 3D version in our 3D applications. As is illustrated in Fig.4, at the beginning, the transformed image is partitioned into two 3D sets – one set of type S and one set of type I. They are added to the LIS and tested against the threshold n. If set S is significant, it will be partitioned into eight/four approximately equal subsets with a smaller size and outputs bit "1". If set I is significant, it is partitioned into three new sets of type S and one/two new set of type I, and also outputs bit "1". If set S or I is insignificant, it stays in the LIS and outputs "0". In Fig.5, t, h and v represent the length of three sides of set S respectively. $P(k,i,j)$ denotes the coefficient which is the closest to $(0,0,0)$ position in set S. So set S can be expressed as $S[(k,i,j)t,h,v]$. After set S is partitioned, the length of three sides of these subsets is defined as

$$\begin{aligned} t' &= \lfloor t/2 \rfloor, \quad t'' = t - \lfloor t/2 \rfloor, \\ h' &= \lfloor h/2 \rfloor, \quad h'' = h - \lfloor h/2 \rfloor, \\ v' &= \lfloor v/2 \rfloor, \quad v'' = v - \lfloor v/2 \rfloor \end{aligned} \tag{7}$$

where each subset is treated as a new set of type S and tested recursively. We use $I(k,i,j)$ to represent the position of set I, as is showed in Fig.6. For example, I_0 is represented as $I(0,R,C)$ in Fig.4 (a), I_1 is represented as $I(B,0,0)$ in Fig. 4 (b), and I_3 is represented as $I(B,2R,2C)$ in Fig. 4 (d). The simple partitioning procedure for set I and set S is summarized as the following:

$I(k,i,j)$: set I at band k, row i and column j.
$S[(k,i,j)t,h,v]$: set of type S at band k, row i and column j; t, h and v represent the length of three sides of set S respectively.
B : total bands of the lowest subband.
R : total rows of the lowest subband.
C : total columns of the lowest subband.
$O(x)$: set x is partitioned into several smaller subsets.

if (set $I(k,i,j)$ is significant)
{
  if (k =0 and $i = R$ and $j = C$)

$$O(I) = \left\{ \begin{matrix} S[(0,0,C)B,R,C], S[(0,R,0)B,R,C], \\ S[(0,R,C)B,R,C], I(0,2R,2C), I(B,0,0) \end{matrix} \right\}$$

  elseif (i =0 and j =0)

$$O(I) = \left\{ S[(k,0,0)k,R,C], I(k,R,C), I(2k,0,0) \right\}$$

  else

$$O(I) = \left\{ \begin{matrix} S[(k,0,j)B,i,j], S[(k,i,0)B,i,j], \\ S[(k,i,j)B,i,j], I(k,2i,2j) \end{matrix} \right\}$$

}
if (set $S[(k,i,j)t,h,v]$ is significant)

$$O(S) = \left\{ \begin{matrix} S[(k,i,j)\lfloor t/2 \rfloor, \lfloor h/2 \rfloor, \lfloor v/2 \rfloor], \\ S[(k,i,j+\lfloor v/2 \rfloor)\lfloor t/2 \rfloor, \lfloor h/2 \rfloor, v-\lfloor v/2 \rfloor], \\ S[(k,i+\lfloor h/2 \rfloor,j)\lfloor t/2 \rfloor, h-\lfloor h/2 \rfloor, \lfloor v/2 \rfloor], \\ S[(k,i+\lfloor h/2 \rfloor,j+\lfloor v/2 \rfloor)\lfloor t/2 \rfloor, h-\lfloor h/2 \rfloor, v-\lfloor v/2 \rfloor], \\ S[(k+\lfloor t/2 \rfloor,i,j)t-\lfloor t/2 \rfloor, \lfloor h/2 \rfloor, \lfloor v/2 \rfloor], \\ S[(k+\lfloor t/2 \rfloor,i,j+\lfloor v/2 \rfloor)t-\lfloor t/2 \rfloor, \lfloor h/2 \rfloor, v-\lfloor v/2 \rfloor], \\ S[(k+\lfloor t/2 \rfloor,i+\lfloor h/2 \rfloor,j)t-\lfloor t/2 \rfloor, h-\lfloor h/2 \rfloor, \lfloor v/2 \rfloor], \\ S[(k+\lfloor t/2 \rfloor,i+\lfloor h/2 \rfloor,j+\lfloor v/2 \rfloor)t-\lfloor t/2 \rfloor, h-\lfloor h/2 \rfloor, v-\lfloor v/2 \rfloor] \end{matrix} \right\}$$

The minimum size of type S is a 1×1×1 block. That means the 1×1×1 set of type S only indicates one coefficient. So if the length of any side of set is 0, it will be defined as empty and not processed. As we can see, in AT-3DSPECK, a set of type S can be partitioned into x subsets. If a set S is significant and its first $x-1$ subsets are insignificant, the x th subset must be significant, and we do not have to send the significance test result of the last subset[9]. Our experiments show that this step can significantly improve the coding performance.

To optimize the rate-distortion performance of the embedded bit stream, during the sorting pass, an important step is that sets should be tested in the increasing order of size[3], [9]. The reason is that if some coefficients are significant, their neighboring coefficients in the same energy cluster have close magnitudes. So it is likely that those not found to be significant in the current pass will be significant against some nearby lower threshold. However, any sorting algorithm will increase the coding time and the encoder/decoder would become very slow[3], [9]. Our

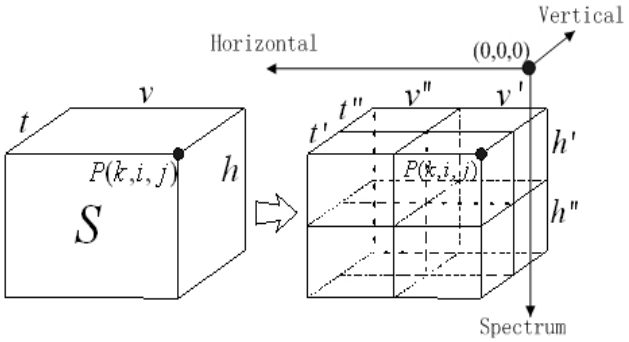


Fig. 5. Partitioning of set S

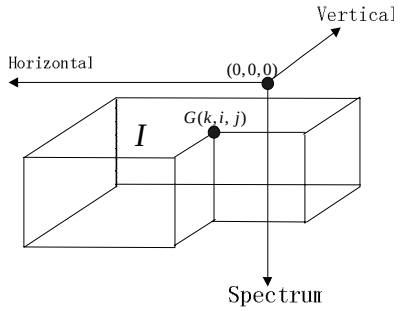


Fig. 6. Illustrate of set I

experiments show that if we did sorting procedure, we would not tolerate the coding time for hyperspectral images with high resolution at high bit rates. To avoid the sorting procedure, all one need to do is to search the LIS several times against the different partitioning depth.

Assume that the hyperspectral images consist of T bands, where each band consists of $H \times V$ pixels in the spatial domain. The 3D set with $T \times H \times V$ pixels has the minimum partitioning depth 0. Let d be the partitioning depth of any set of type S. The lengths of three sides of set of type S are: t, h and v. To describe the partitioning depth simply, we consider that these sets with the same $h \times v$ have the same partitioning depth. Then the partitioning depth of any set is given by $d = \log_2(H/h)$ or $d = \log_2(V/v)$. When a set with the depth d is partitioned, the partitioning depth of its subsets is $d+1$. Clearly, the size of set of type S in LIS is in inverse proportion to its partitioning depth. In each bit plane, all we need to do is to search the LIS several times against the different partition depth from the maximum depth to the minimum depth. The simple partitioning procedure for set I and set S is summarized as the following:

$D_{\max}$, $D_{\min}$: the maximum partitioning depth and the minimum partitioning depth among all sets.

$d(S)$: the partition depth of set S.

$O(S)$: set S is partitioned into several smaller subsets.

for $l = D_{\max}$: $D_{\min}$

for each set S in LIS, do

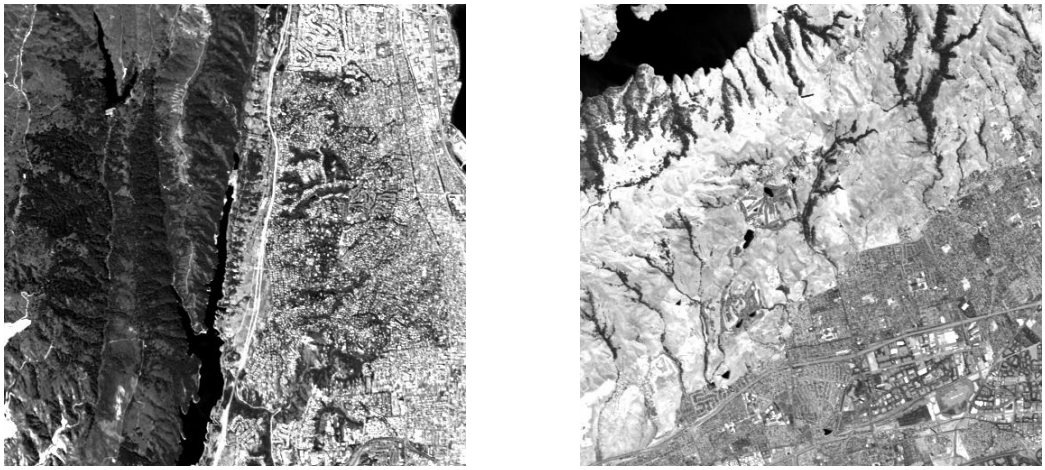
if $d(S) = l$ do $O(S)$, else skip

The above partitioning procedure and sorting rule satisfy the condition where the width is not equal to the height in the spatial domain for 3D images. If we skip the sorting rule, the computational time will decrease proportionally. But at the same time it is likely that the lossy compression performance will decrease proportionally.

3. NUMERICAL RESULTS

We use two hyperspectral images "Jasper1" and "Moffett1" to evaluate the compression performance. All test images are typical AVIRIS images and are provided by NASA. There are 224 spectral bands, each consisting of 614 lines of 512 pixels quantized at 12 bit per pixel per band (bpppb). For our experiments, a cube set of size $512 \times 512 \times 224$ is selected and used as the test set. Fig.7 (a) shows band 188 of the "Jasper1" images, and Fig.7 (b) shows band 120 of the "Moffett1" images.

The performance improves with the coding unit size in general, but the gain is smaller and smaller as the coding unit size increase[1], [4], [6]. So the test images are coded with the unit size of 16 bands in our implementation. We use four-level wavelet transforms with the 5/3, 5/11, 9/3, 9/7, 13/11[6] and 9/7-F[13] tap biorthogonal filters along all the directions. The implementation of the AT-3DSPIHT follows the algorithm presented in [5]. All algorithms apply the arithmetic coding to further improve the coding performance.



(a) (b)

Fig. 7. Band 188 of the (a) "Jasper1" and band 120 of (b) "Moffett1" used in the experiments

3.1. Lossless compression

Table 2 gives the average lossless compression results (bpppb) with the subband scaling for the test images. In addition, different filters perform differently for the test images. The results show that AT-3DSPECK is better coder than AT-3DSPIHT. We notice that especially the 5/3 filter are performing well, closely followed by the 9/3 filter.

The results in Table 3 are the lossless compression bitrate using the 5/3 filter without the subband scaling. Clearly, these results are better than that in Table 2. This is because the subband scaling by bit shifts causes a bit

Table 2. Lossless compression comparisons using different filters with the subband scaling

	Jasper1		Moffett1	
bpppb	AT-3DSPIHT	AT-3DSPECK	AT-3DSPIHT	AT-3DSPECK
5/3	7.30	7.22	7.40	7.31
5/11	7.52	7.42	7.63	7.53
9/3	7.32	7.23	7.40	7.32
9/7	7.48	7.38	7.59	7.49
13/11	7.55	7.45	7.66	7.57
Average	7.43	7.34	7.54	7.44

Table 3. Lossless compression comparisons using the 5/3 filter without the subband scaling

bpppb	3DSPECK	AT-3DSPIHT	AT-3DSPECK	2DSPIHT	2DSPECK	JPEG2000
Jasper1	7.35	6.87	6.71	8.72	8.65	8.83
Moffett1	7.44	6.96	6.79	8.77	8.71	8.89

Table 4. SNR (dB) performance comparisons using the 5/3 integer filter with the subband scaling and 9/7 floating-point filter for lossy coding

	Jasper1					Moffett1				
bpppb	0.2	0.5	1	2	4	0.2	0.5	1	2	4
3DSPECK(5/3)	19.03	23.26	28.68	37.10	48.07	22.15	26.34	31.96	40.55	51.24
AT-3DSPIHT(5/3)	24.59	32.33	36.60	41.38	46.60	27.19	35.08	39.57	43.36	49.90
AT-3DSPECK(5/3)	25.62	32.42	37.32	42.20	50.54	28.51	35.89	40.46	44.93	53.55
3DSPECK(9/7F)	19.28	23.41	28.73	37.05	48.20	22.40	26.49	31.97	40.44	51.38
AT-3DSPIHT(9/7F)	26.03	33.09	37.30	41.51	47.18	28.77	35.80	39.96	44.07	50.19
AT-3DSPECK(9/7F)	26.40	33.14	37.87	43.17	53.16	29.41	36.52	40.92	45.92	56.25

growth and hence a data expansion, inefficient for lossless compression. On the other hand, as mentioned earlier, the subband scaling ensures higher compression ratios for lossy coding. To gain higher lossy compression ratios, these codecs are equipped with the subband scaling. Compared to the 2DSPECK, 2DSPIHT and JPEG2000, obviously, the 3D versions (3DSPECK, AT-3DSPIHT and AT-3DSPECK) guarantee at lest 16% improvement for lossless compression.

3.2. Lossy compression

Since the codec is embedded in our applications, lossy coding results can be generated easily from decoding the same encoded bitstream with different target bit rates. The various bit rates and corresponding reconstruction Signal-to-Noise Ratio (SNR) for coding the test images "Jasper1" and "Moffett1" over a wide bit rate range using AT-3DSPECK, 3DSPECK[3] and AT-3DSPIHT[5] are shown in Table 4, where SNR is averaged over all bands. We see that AT-3DSPECK outperforms 3DSPECK and AT-3DSPIHT at all rates once again. AT-3DSPECK is 1.6dB better than AT-3DSPIHT on average. AT-3DSPIHT achieves higher SNR than 3DSPECK at very low and middle bit rate, but at high rates 3DSPECK performs slightly better than AT-3SPIHT. We notice that 9/7 floating-point wavelet for lossy compression provides slightly better compression ratios than integer wavelet. Although the lossy performance using integer wavelet is not quite as good as that for floating-point wavelet, integer wavelets are less complex and offer excellent performance for lossless image compression applications.

4. CONCLUSION

In this paper, a modified 3DSPECK algorithm (AT-3DSPECK) based on 3D integer wavelet packet transforms has been proposed for lossy to lossless compression of hyperspectral images. The two powerful techniques: the asymmetric 3D wavelet transform and 3D set partitioning block algorithm are efficiently combined in the proposed algorithm. The asymmetric 3D wavelet transform can efficiently reduce the correlation along the spatial and spectral dimensions and concentrate much more energy into a single band. To gain good lossy compression performance, our method use a 3D integer wavelet packet transforms with the subband scaling to approximate a 3D unitary transforms. When comparing the AT-3DSPECK to other three-dimensional coder like AT-3DSPIHTand 3DSPECK, AT-3DSPECK gave good results at any bit rate, especially at the higher bit rates. For lossless coding, which is a very important feature in hyperspectral image compression, AT-3DSPECK is outperforming the other coders. Thus, the proposed AT-3DSPECK algorithm is a better candidate than AT-3DSPIHT and several others methods based on 3D wavelet transform for hyperspectral image compression. Moreover, AT-3DSPECK can be applied in compression of medical volumetric data. In our future work, we plan to pay increased attention to the video coding part, using 3D wavelet transform and motion compensation.

ACKNOWLEDGMENTS

This paper is supported by the National Natural Science Foundation of China (No. 60371020).

REFERENCES

1. P.L. Dragotti, G. Poggi, and A.R.P. Ragozini, "Compression of multispectral images by three-dimensional SPIHT algorithm", *IEEE Trans. Geosci. Remote sensing*, 38, pp.416-428, 2000
2. B.-J. Kim and W.A. Pearlman, "An Embedded Wavelet Video Coder Using Three-Dimensional Set Partitioning in Hierarchical Trees (SPIHT)", *Proceedings of the IEEE Data Compression Conf.*, pp.251-260, 1997.
3. X. Tang, W. A. Pearlman, and J. W. Modestino, "Hyperspectral Image Compression Using Three-Dimensional Wavelet Coding", *Proceedings of SPIE/IS&T Electronic Imaging 2003*, 5022, Jan. 2003.
4. C. He, J. Dong, and Y.F. Zhang, "Optimal 3-D Coefficient Tree Structure for 3-D Wavelet Video Coding", *IEEE Trans. on Circuits Syst. Video Technol.*, 13, pp.961-972, 2003.
5. X. Tang, S. Cho, and W. A. Pearlman, "3d set partitioning coding methods in hyperspectral image compression", *Proceedings of the IEEE International Conference on Image Processing, ICIP2003*, pp.239-242, 2003.
6. Z. Xiong, X. Wu, S. Cheng and J. Hua, "Lossy-to-Lossless Compression of Medical Volumetric Data Using Three-Dimensional Integer Wavelet Transforms", *IEEE Trans. on Medical Imaging*, 22, pp.456-470, 2003.
7. P. Schelkens, A. Munteanu, J. Barbarien, M. Galca, X. Giro-Nieto, and J. Cornelis, "Wavelet coding of volumetric medical datasets", *IEEE Trans. on Medical Imaging*, 22, pp. 441-458, 2003.

8. A. Said and W.A. Pearlman, “A new, fast and efficient image codec based on set partitioning in hierarchical trees”, *IEEE Trans. on Circuits Syst. Video Technol.*, 6, pp.243-250, 1996.
9. Asad Islam and W.A. Pearlman, “An embedded and efficient low-complexity hierarchical image coder”, *Proceedings of SPIE Visual Communications and Image Processing*, 3653, pp.284-305, 1999.
10. S-T. Hsiang and J. W. Woods, “Embedded Image Coding Using Zeroblock of Subband/Wavelet Coefficients and Context Modeling”, *Proceedings of IEEE Int. Conf. on Circuits and Systems (ISCAS2000)*, 3, pp.662-665, 2000.
11. W. Sweldens, “The Lifting Scheme: A New Philosophy in Biorthogonal Wavelet Constructions”, *SPIE Conference 1995*, 2569, pp.68-79, 1995.
12. I. Daubechies and W. Sweldens, “Factoring wavelet transforms into lifting steps,” *J. Fourier Anal. Appl.*, 4, pp. 247–269, 1998.
13. M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, “Image Coding using wavelet transform”, *IEEE Trans. Image Processing*, 1, pp.205-220, 1992.