Digital Signal Processing 64 (2017) 96–106

Contents lists available at ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

Remote sensing image compression based on binary tree and optimized truncation [☆]

Ke-Kun Huang [a], Hui Liu [a], Chuan-Xian Ren [b,*], Yu-Feng Yu [b], Zhao-Rong Lai [c]

[a] School of Mathematics, JiaYing University, Meizhou, Guangdong, 514015, China
[b] Intelligent Data Center and Department of Mathematics, Sun Yat-Sen University, Guangzhou 510275, China
[c] Department of Mathematics, College of Information Science and Technology, JiNan University, Guangzhou 510632, China

ARTICLE INFO

Article history:
Available online 28 February 2017

Keywords:
Remote sensing image compression
On-board compression
Binary tree coding
Rate-distortion optimization
Wavelet application

ABSTRACT

The remote sensing image data is so vast that it requires compression by low-complexity algorithm on space-borne equipment. Binary tree coding with adaptive scanning order (BTCA) is an effective algorithm for the mission. However, for large-scale remote sensing images, BTCA requires a lot of memory, and does not provide random access property. In this paper, we propose a new coding method based on BTCA and optimize truncation. The wavelet image is first divided into several blocks which are encoded individually by BTCA. According the property of BTCA, we select the valid truncation points for each block carefully to optimize the ratio of rate-distortion, so that a higher compression ratio, lower memory requirement and random access property are attained. Without any entropy coding, the proposed method is simple and fast, which is very suitable for space-borne equipment. Experiments are conducted on three remote sensing image sets, and the results show that it can significantly improve PSNR, SSIM and VIF, as well as subjective visual experience.

© 2017 Elsevier Inc. All rights reserved.

1. Introduction

Remote sensing images have wide applications such as object detection [1], environmental monitoring [2], and urban planning [3], etc. With the rapid development of sensor technology, high spatial resolution images are more easily acquired. Since the images will take up a great deal of storage space, and the hardware of space-borne equipments is limited, great efforts are made to seek low-complexity image compression algorithms.

Because wavelet transform expresses the local time-domain and frequency-domain characterization and provides excellent multiresolution analysis features, most of the state-of-the-art coding systems are based on wavelet transform. The Embedded Zerotree Wavelet (EZW) [4] is the first successful compression algorithm based on wavelet transform, which constructs zero-tree by the correlations across scales in each bit-plane, so that a large number of no-significant coefficients are predicted successfully by a root of zerotree. Set Partitioning in Hierarchical Tree (SPIHT) [5] is another famous algorithm, which can significantly improve the performance with lower complexity compared with EZW. In [6], the Set Partitioned Embedded Block (SPECK) coder was proposed to extend SPIHT.

EZW, SPIHT and SPECK exploit self-similarity across scales, while there are some algorithms that utilize the clustering characteristic within each subband. In [7], the QuadTree Coding (QTC) algorithm for wavelet image compression was proposed. It splits each node into four descendent nodes once it is significant with the current threshold. QTC produces an embedded data stream, supports quality scalability, and permits region-of-interest coding. The Embedded Block Coding with Optimized Truncation (EBCOT) algorithm [8], adopted in the JPEG2000 standard [9], combines layered block coding, rate-distortion optimization, and context-based arithmetic coding in an efficient and highly scalable way.

Presently, there are some methods which are specifically designed for remote sensing images. In [10], Li et al. proposed the two-dimensional oriented wavelet transform for remote sensing images based on JPEG2000. Kulkarni et al. [11] presented a scan-based JPEG2000 for large-scale images. However, JPEG2000 is too complex, which hinders JPEG2000 to be a compression standard for space-borne missions where the capacity of storage and transmission is limited.

[☆] This work is supported in part by National Science Foundation of China under Grants 61403164 and 61572536, in part by the Fundamental Research Funds for the Central Universities under Grant 16lgzd16, and in part by the Foundation for Distinguished Young Talents in Higher Education of Guangdong, China (2013LYM_0085).

[*] Corresponding author.
E-mail addresses: kkcocoon@163.com (K.-K. Huang), imlhxm@163.com (H. Liu), rchuanx@mail.sysu.edu.cn (C.-X. Ren), yuyufeng220@163.com (Y.-F. Yu), laizhr@jnu.edu.cn (Z.-R. Lai).

http://dx.doi.org/10.1016/j.dsp.2017.02.008
1051-2004/© 2017 Elsevier Inc. All rights reserved.

In [12], the Consultative Committee for Space Data Systems (CCSDS) [13] published an image data compression standard (CCSDS-IDC) based on wavelet transform. The recommendation specifically targets space-borne missions, and focuses more on complexity and less on compression performance. It has a limited set of options, supporting its successful application without in-depth algorithm knowledge. In [14], some extensions of CCSDS-IDC have been proposed. In [15], the directional lifting wavelet transform was used to improve the performance of CCSDS-IDC.

Besides JPEG2000 and CCSDS-IDC, the other compression algorithms were also applied for remote sensing image. In [16], a modified listless strip based SPIHT was proposed to reduce system complexity and minimize processing time and memory usage. In [17], an improved SPIHT was proposed for multispectral image compression for various band images with high resolution. In [18], a three dimensional SPECK was proposed for hyperspectral image compression. In [19], a SAR complex image data compression algorithm based on QTC in wavelet transform domain was proposed, showing that QTC achieves the best performance for SAR complex image compression. In [20], the cubic B—spline fuzzy transform was proposed for an efficient and secure compression in wireless sensor networks.

Recently, compressed sensing has been successfully used in many fields of computer vision [21,22], which also has been proved to perform well on image compression. In [23], a SAR image compression using multiscale dictionary learning and sparse representation was proposed, which is better for preserving the important features of SAR images with a competitive compression performance. In [24], an adaptive spectral-spatial compression of hyperspectral image with sparse representation has been proposed, which outperforms some state-of-the-art HSI compression methods in terms of the rate-distortion and spectral fidelity performances. For remote sensing image compression, the complexity of algorithm is very important. A new fast algorithm for image and video compression based on discrete Tchebichef transforms DTT has shown good performance [25]. The proposed DTT-based method is multiplication-free and only requires a reduced number of additions and bit-shifting operations, which is suitable for remote sensing image compression for its lower complexity.

In 2012, a fast remote sensing image coder, named binary tree coding with adaptive scanning order (BTCA) [26], was presented. It used the binary tree for coding remote sensing image in wavelet domain, and then developed an adaptive scanning order to traverse the binary tree, so that better performance and visual effect are attained. Unlike JPEG2000 or CCSDS-IDC, BTCA is very fast because it does not use any entropy coding. BTCA is very easy to implement in hardware and very suitable for on-board compression. Since BTCA was proposed, some improvements have been developed. In [27], an embedded image compression based on fractal and wavelet is proposed, where the lowest frequency subband of wavelet domain is coded by fractal first, and the coding error and other sub-bands are coded by BTCA. In [28], a Human vision-based Adaptive Scanning (HAS) for the compression of remote sensing image was proposed, which generates an importance weighting mask according to the human visual characteristics before applying BTCA. In [29], a content-based adaptive scanning scheme was proposed for remote sensing image compression, which provides different scanning orders among and within subbands based on BTCA.

However, BTCA needs a lot of memory to store the binary tree for large-scale remote sensing images, and does not provide random access property. In [26], BTCA is processed with a scan-based mode, named BTCAS, to deal with this problem. Nevertheless, BTCAS sacrifices in compression efficiency. In this paper, we propose a new coding method based on **B**inary **T**ree and **O**ptimized **T**runcation (**BTOT**). Compared with BTCA, the proposed BTOT method significantly improves the PSNR, while it requires less memory and has the random access property.

The main contributions of this paper are listed as follows:

- We propose an improved BTCA algorithm by optimized truncation. The wavelet image is first divided into several blocks which are encoded individually by BTCA. Then we apply rate-distortion optimization to get the higher compression ratio, less memory requirement and the random access property.
- We find that the distortion-rate almost decreases as the adaptive scanning level increases, so we record the bit rate and corresponding distortion as a candidate truncation point after each adaptive scanning level to get finer embedded bitstreams.

The remainder of the paper is organized as follows: In section 2, we introduce the BTCA algorithm briefly. In section 3, we describe the proposed method in detail. In section 4, we give the analyses about time complexity, memory requirement and the feature of the coding stream. The experimental results are presented in section 5. Finally, we provides the conclusion in section 6.

2. The BTCA algorithm

Because the coefficients in the high-frequency wavelet subbands are sparse, we can divide the subbands into several blocks and then check whether they contain significant coefficients. The tree structure is widely used in many fields of information sciences [30–32], and is also a good model for compression [33]. In [7], each block is split into four sub-blocks once it tests as significant with respective to the current threshold. However, the binary tree is more efficient than quadtree [34], so the binary tree coding in wavelet domain is proposed in [26]. Because our method is based on BTCA, we briefly describe BTCA as follows.

Suppose a wavelet image is of size $2^N \times 2^N$. We first transform it into an one-dimensional vector using Morton scanning order, denoted by V. Then we construct the binary tree $\Gamma(t)$ for $1 \le t < 2 \times 2^N \times 2^N$. The bottom level of the binary tree consists of the absolute value of each coefficient:

$$\Gamma(t) = \left| V(t - 2^N \times 2^N) \right| \text{ for } 2^N \times 2^N \le t < 2 \times 2^N \times 2^N. \tag{1}$$

The upper levels of the tree are defined iteratively:

$$\Gamma(t) = \max\{\Gamma(2t), \Gamma(2t+1)\} \text{ for } 1 \le t < 2^N \times 2^N. \tag{2}$$

After constructing the binary tree, we can traverse the tree by depth-first, which can be expressed as a function $code = BTC(\Gamma, t, T_k)$ as Algorithm 1, where t is a tree node index of the binary tree, and T_k is a threshold where $T_0 = 2^{\lfloor \log_2 \Gamma(1) \rfloor}$ and $T_k = T_0/2^k$.

The coefficients on the edges are often large, so we can scan the neighbor of the previous significant coefficients before other regions are scanned. BTCA uses an adaptive scanning order, namely, from the bottom level to the top, if a coefficient's brother is a previous significant coefficient, then we traverse the coefficient and its descendants by depth-first. Based on Algorithm 1, the detail steps of BTCA can be described as a function $code = BTCA(\Gamma, T_k)$ as Algorithm 2.

3. Compression based on binary tree and optimized truncation

BTCA achieves state-of-the-art performance, but it needs a lot of memory to store the binary tree for a large remote sensing image, and does not provide random access property. In this section, we propose a new method based on binary tree and optimized

Algorithm 1 Function $code = BTC(\Gamma, j, T_k)$.

```
if $\Gamma(j) \geq T_{k-1}$ then
  % If it has been coded with significant in larger threshold.
  if $j < 2^N \times 2^N$ then
    $l = BTC(\Gamma, 2j, T_k)$;
    $r = BTC(\Gamma, 2j+1, T_k)$;
    $code = l \cup r$;
  else
    $code = \{sign(V(j - 2^N \times 2^N))\}$.
  end if
else if $j > 1$ and $j$ mod $2 = 1$ and $\Gamma(j-1) < T_k$ then
  % If it has a significant parent and its brother has just been coded with insignificant.
  if $j < 2^N \times 2^N$ then
    $l = BTC(\Gamma, 2j, T_k)$;
    $r = BTC(\Gamma, 2j+1, T_k)$;
    $code = l \cup r$;
  else
    $code = \{sign(V(j - 2^N \times 2^N))\}$.
  end if
else if $\Gamma(j) \geq T_k$ then
  % If it is significant with current threshold.
  if $j < 2^N \times 2^N$ then
    $cl = BTC(\Gamma, 2j, T_k)$;
    $cr = BTC(\Gamma, 2j+1, T_k)$;
    $code = \{1\} \cup l \cup r$;
  else
    $code = \{1, sign(V(j - 2^N \times 2^N))\}$.
  end if
else
  $code = \{0\}$;
end if
```

Algorithm 2 Function $code = BTCA(\Gamma, T_k)$.

```
$d = 2N + 1$;
while $d > 1$ do
  % Traverse the tree level by level from the bottom to the top.
  for $j = \sum_{i=0}^{d-1} 2^i + 1$ to $\sum_{i=0}^{d} 2^i$ do
    if $\Gamma(j) \geq T_{k-1}$ then
      if $j$ mod $2 = 0$ and $\Gamma(j+1) < T_{k-1}$ then
        % If the left subtree is significant, then coding the right subtree.
        $c = BTC(\Gamma, j+1, T_k)$;
      else if $j$ mod $2 = 1$ and $\Gamma(j-1) < T_{k-1}$ then
        % If the right subtree is significant, then coding the left subtree.
        $c = BTC(\Gamma, j-1, T_k)$;
      end if
    end if
    $code = code \cup c$;
  end for
  $d = d - 1$;
end while
```

truncation, in order to get higher compression ratio and less memory requirement, as well as the random access property.

The wavelet image is first divided into several blocks. Then the blocks are encoded by BTCA independently. To apply rate-distortion optimization, the most important step is how to get the valid truncation points. We find that the distortion-rate almost decreases as the adaptive scanning level increases, so we can record the bit rate and corresponding distortion as a candidate truncation point after each adaptive scanning level. The number of the proposed candidate truncation points is more than EBCOT [8] in which there are only 4 candidate truncation points for each bit-plane. The detail process can be described as follows.

Suppose the bit rate R_j and the corresponding distortion D_j at the jth candidate truncation point of a block are obtained. Let

$$S_j = \frac{D_{j-1} - D_j}{R_j - R_{j-1}} \tag{3}$$

denote the local distortion-rate. We hope that S_j decreases as j increases. Fig. 1 shows the variation of S_j when using the proposed method to encode a block of a remote sensing image with four bit-

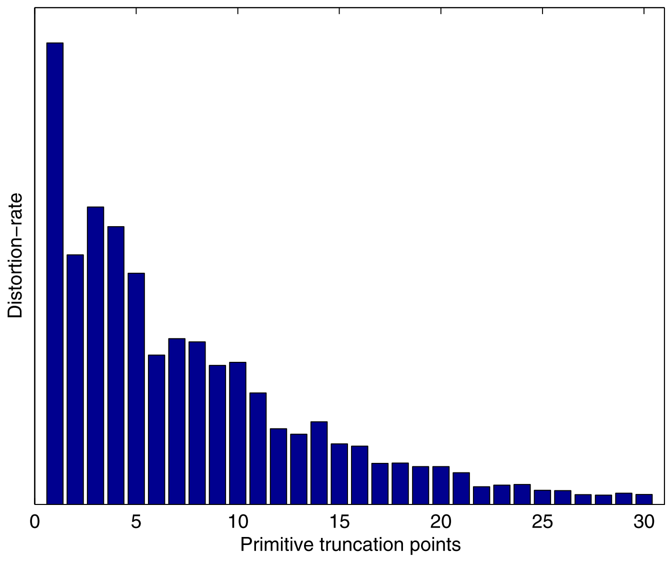


Fig. 1. Distortion-rates of the candidate truncation points of the proposed method.

planes whose size is $2^6 \times 2^6$. Except the reduplicative truncation points, there are 30 candidate truncation points in the proposed method, while there are $1 + 4 \times 3 = 13$ points in EBCOT [8]. So the proposed method can achieve finer embedded bit-streams than EBCOT.

We use the Post-Compression Rate Distortion (PCRD) optimization algorithm [8] to select the candidate truncation points of the independent block such that the distortion is minimized at a specified bit rate. If $S_{j+1} > S_j$, then jth candidate truncation point cannot be selected, and we need to update

$$S_{j+1} = \frac{D_{j-1} - D_{j+1}}{R_{j+1} - R_{j-1}}. \tag{4}$$

Then we compare it with S_{j-1}. If $S_{j+1} > S_{j-1}$, then $(j-1)$th candidate truncation point cannot be selected, and we need to update

$$S_{j+1} = \frac{D_{j-2} - D_{j+1}}{R_{j+1} - R_{j-2}}. \tag{5}$$

This process is performed until $S_{j+1} < S_{j_k}$, where

$$S_{j+1} = \frac{D_{j_k-1} - D_{j+1}}{R_{j+1} - R_{j_k-1}}. \tag{6}$$

We can select the strictly decreasing $\{S_{j_k}\}$ as a series of valid truncation points. Fig. 2 shows distortion-rates of the valid truncation points of the candidate truncation points in Fig. 1. Because S_j is almost monotonically decreasing, it remains most of the candidate truncation points in Fig. 2. Using PCRD optimization algorithm, a final truncation point can be attained for each block at a specified rate. The number of the final truncation points is equal to the number of blocks, so there are only a little extra storage to record the truncation points. Applying the rate-distortion optimization, the proposed method gets better performance and requires less memory, as well as random access property.

The proposed method based on **B**inary **T**ree and **O**ptimization **T**runcation is called **BTOT**. The detail encoding algorithm of the proposed method can be summarized as Algorithm 3.

To better illustrate the proposed method, we give a simple example. Suppose that there is a 4×4 block whose coefficients are shown in the left side of Fig. 3. The block is transformed into one-dimensional vector with Morton scanning order, resulting $V = \{10, 5, 4, 3, 3, 1, 2, 5, 0, 0, 1, 1, 2, 3, 6, 3\}$. Then we can construct the binary tree according to Eq. (1) and Eq. (2), as shown

Algorithm 3 The detail encoding algorithm of the proposed BTOT method.

Input: The original image I; The coding length *bits*.
Output: The encoded information *code* and the code length of each block R_t.

```
1:  Perform wavelet transform: $I_w = DWT(I)$;
2:  Divide $I_w$ into blocks $\{B_m, m = 1, \cdots, M\}$ where $B_m$ is of size $2^n \times 2^n$;
3:  % Coding each block alone as follows.
4:  for $m = 1$ to $M$ do
5:      Construct binary tree $\Gamma_m$ for $B_m$;
6:      Initialize truncation points $R_m = \{\}$ and $D_m = \{\}$.
7:      Traverse the binary tree by depth-first with $T_0$:
          $c_m = BTC(\Gamma_m, 1, T_0)$;
8:      Record truncation points: $R_m = \{R_m, r\}$ and $D_m = \{D_m, MSE\}$, where $r = length(c_m)$ and $MSE$ denotes the mean square errors between original image and the image reconstructed by the current code $c_m$.
9:      % Coding with each threshold $T_k$ as follows.
10:     for $k = 1$ to $K$ do
11:         $d = 2n + 1$; % The depth of binary tree.
12:         while $d > 1$ do
13:             % Traverse the tree level by level from the bottom to the top.
14:             for $j = \sum_{i=0}^{d-1} 2^i + 1$ to $\sum_{i=0}^{d} 2^i$ do
15:                 if $\Gamma_m(j) \geq T_{k-1}$ then
16:                     if $j$ mod $2 = 0$ and $\Gamma_m(j+1) < T_{k-1}$ then
17:                         % If the left subtree is significant, then coding the right subtree.
18:                         $c = BTC(\Gamma_m, j+1, T_k)$;
19:                     else if $j$ mod $2 = 1$ and $\Gamma_m(j-1) < T_{k-1}$ then
20:                         % If the right subtree is significant, then coding the left subtree.
21:                         $c = BTC(\Gamma_m, j-1, T_k)$;
22:                     end if
23:                 end if
24:                 $c_m = c_m \cup c$;
25:             end for
26:             $d = d - 1$;
27:             Record truncation points;
28:         end while
29:         Perform magnitude refinement pass and record truncation points;
30:     end for
31: end for

32: Initialize $R_v = \{\}$, $S_v = \{\}$. % Valid truncation points of all blocks.
33: for $m = 1$ to $M$ do
34:     Initialize $D = \{0\}$, $R = \{0\}$, $S = \{\infty\}$, $j = 1$; % Valid truncation points of the current block.
35:     for $i = 1$ to $length(R_m)$ do
36:         $S_t = \frac{D(j) - D_m(i)}{R_m(i) - R(j)}$;
37:         while $S_t > S(j)$ do
38:             $j = j - 1$;
39:             $S_t = \frac{D(j) - D_m(i)}{R_m(i) - R(j)}$;
40:         end while
41:         $j = j + 1$;
42:         $D(j) = D_m(i)$;
43:         $R(j) = R_m(i)$;
44:         $S(j) = S_t$;
45:     end for
46:     Delete the first element in $R$ and $S$, and append some zeros to $R$ and $S$ such that $length(R) = L$ and $length(S) = L$;
47:     $R_v = R_v \cup R$; $S_v = S_v \cup S$;
48: end for

49: Sort $S_v$ in descending order and get the permutation vector $S_{ind}$ of the corresponding element;
50: Initialize $R_t(m) = 0$ for $m = 1, \cdots, M$;
51: $i = 1$;
52: while $\sum_m (R_t(m)) < bits$ do
53:     $s_i = S_{ind}(i)$;
54:     $m = mod(s_i, L)$;;
55:     $R_t(m) = R_v(s_i)$;
56:     $i = i + 1$;
57: end while
58: Initialize $code = \{\}$;
59: for $m = 1$ to $M$ do
60:     $code = code \cup c_m(1 : R_t(m))$;
61: end for
```

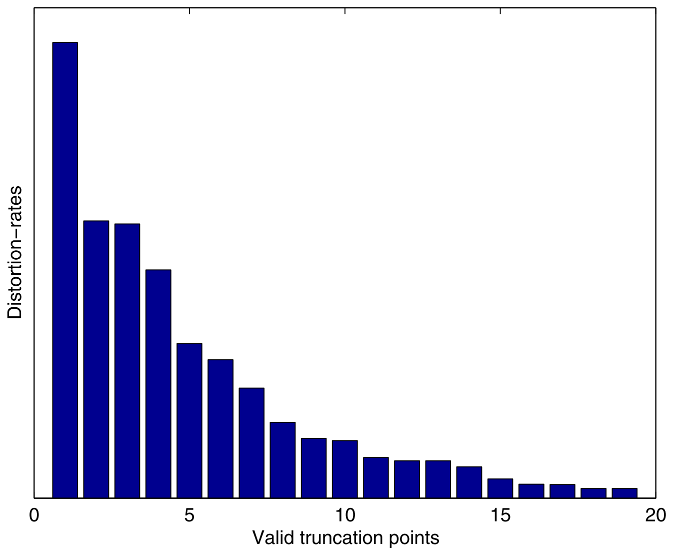


Fig. 2. Distortion-rates of the valid truncation points of the proposed method.

Table 1
The candidate truncation points of the proposed method for the example in Fig. 3.

No.	T	Level	D	R	S
0	–	–	249	0	inf
1	8	–	153	10	9.6
2	4	4	129	12	12.0
3	4	3	129	13	0.0
4	4	2	93	21	4.5
5	4	1	57	27	6.0
6	4	0	53	28	4.0
7	2	4	32	34	3.5
8	2	3	23	39	1.8
9	2	2	11	45	2.0
10	2	0	8	50	0.6

in the right side of Fig. 3. Table 1 shows the candidate truncation points for coding the block. We initialize the distortion by $D_0 = \sum_{i,j} x_{i,j}^2 = 249$, the rate $R_0 = 0$ and the distortion-rate $S_0 = \text{inf}$. Then we first traverse the binary tree by depth-first with $T = 8$. The resulting code is "1111100000", and the reconstructed coefficients are 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, respectively, so $R_1 = 10$, $D_1 = 153$ and $S_1 = \frac{D_0 - D_1}{R_1 - R_0} = 9.6$.

Then we traverse the binary tree from bottom level to top level with $T = 4$. For each level, if the brother of a coefficient is significant with $T = 8$, then we prefer to traverse the coefficient and its descendants, i.e., with $T = 4$, we perform $BTC(\Gamma, 17, 4)$ for level $= 4$, $BTC(\Gamma, 9, 4)$ for level $= 3$, $BTC(\Gamma, 5, 4)$ for level $= 2$ and $BTC(\Gamma, 3, 4)$ for level $= 1$ in turn. After traversing with $T = 4$, we perform magnitude refinement pass for all the previous significant coefficients, as No. 6 candidate truncation point in Table 1.

With $T = 2$, we perform $BTC(\Gamma, 19, 2)$, $BTC(\Gamma, 22, 2)$ and $BTC(\Gamma, 31, 2)$ for level=4, $BTC(\Gamma, 10, 2)$ and $BTC(\Gamma, 14, 2)$ for level=3, $BTC(\Gamma, 6, 2)$ for level=2 in turn, i.e., we traverse the brothers of the yellow coefficients in Fig. 3 first. After traversing with $T = 4$, we get 10 candidate truncation points, as shown in Table 1.

After the candidate truncation points are attained, we need to calculate the valid truncation points. Because $S_2 > S_1$, the No. 1 candidate truncation point cannot be selected as a valid one, and we need to update $S_2 = \frac{D_0 - D_2}{R_2 - R_0} = \frac{249 - 129}{12 - 0} = 10$. Because $S_4 > S_3$, we need to update $S_4 = \frac{D_2 - D_4}{R_4 - R_2} = \frac{129 - 93}{21 - 12} = 4$. This process can be repeated and we can get all the valid truncation points, as shown in Table 2.

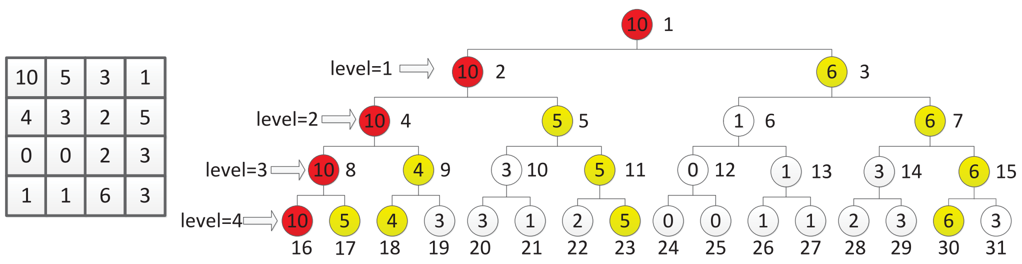


Fig. 3. A simple example to illustrate the proposed method. Left: a block to be encoded. Right: The corresponding binary tree.

Table 2
The valid truncation points of the proposed method for the example in Fig. 3.

No.	T	Level	D	R	S
0	–	–	249	0	inf
2	4	4	129	12	10.0
5	4	1	57	27	4.8
6	4	0	53	28	4.0
7	2	4	32	34	3.5
9	2	2	11	45	1.9
10	2	0	8	50	0.6

4. Analysis and discussion

4.1. Time complexity

Suppose an entire wavelet image is of size $2^N \times 2^N$, and the blocks are of size $2^n \times 2^n$, where $N = pn$. There are $2^p \times 2^p = 2^{2p}$ blocks. For each block, there are $2^n \times 2^n = 2^{2n}$ coefficients consisting the bottom level of the binary tree. Constructing a tree node of upper level requires a comparison with two descendants. Let $D = \log_2\left(2^{2n}\right) + 1 = 2n + 1$ stands for the tree depth, then the comparison times for constructing all the trees of BTOT is:

$$\begin{aligned} C_{BTOT} &= 2^{2p}2^{2n}\left(\frac{1}{2} + \frac{1}{2^2} + \cdots + \frac{1}{2^{D-1}}\right) \\ &= 2^{2p}2^{2n}\left(1 - \frac{1}{2^{2n}}\right) \\ &= 2^{2p}(2^{2n} - 1) \\ &= (2^{2N} - 2^{2p}). \end{aligned} \tag{7}$$

While the comparison times for constructing the binary tree of BTCA is:

$$\begin{aligned} C_{BTCA} &= 2^{2N}\left(\frac{1}{2} + \frac{1}{2^2} + \cdots + \frac{1}{2^{2N}}\right) \\ &= 2^{2N}\left(1 - \frac{1}{2^{2N}}\right) \\ &= 2^{2N} - 1. \end{aligned} \tag{8}$$

We can find that $C_{BTOT} < C_{BTCA}$, which means that BTOT does not increase the complexity for constructing the binary tree, but decreases a little.

After constructing the binary tree, we need to traverse the tree for each bit-plane. According to Algorithm 1, we can find that the coding process of BTCA and that of BTOT are similar, except that BTOT records some truncation points. However, in order to perform rate distortion optimization, we have to do some extra coding process for each blocks. On the other hand, after recoding the truncation points, we need to apply PCRD optimization to get a final truncation point in each block at a specified bit rate. So, BTOT will take more encoding process than BTCA. Considering BTOT requires much less memory and provides resolution and SNR scalability, the extra coding process is valuable.

In fact, most of the computation cost in entropy coding [26]. For instance, the coding time for SPIHT with arithmetic coding is 0.33 s, while it is 0.14 s without arithmetic coding [5]. So the methods using entropy coding, such as EBCOT [8], JPEG2000 [9], are too complex to be a standard for space-borne tasks.

4.2. Memory requirement

Most of coding methods need to store significant/insignificant coefficients and sets. For example, when coding an image of size 512×512 at 0.5 bpp for SPIHT, there are about 25000 significant coefficients, 50000 insignificant coefficients, and 32000 insignificant sets. If we use linked list to store them, then each one needs 8 bytes. Hence, the memory for recoding significant/insignificant coefficients and sets is about 0.4M bytes for SPIHT. Besides, it needs to store the original wavelet image, whose size is 1M bytes, and needs to store the final coding stream which needs 0.0625M bytes, so the total memory requirement is about 1.5M for SPIHT.

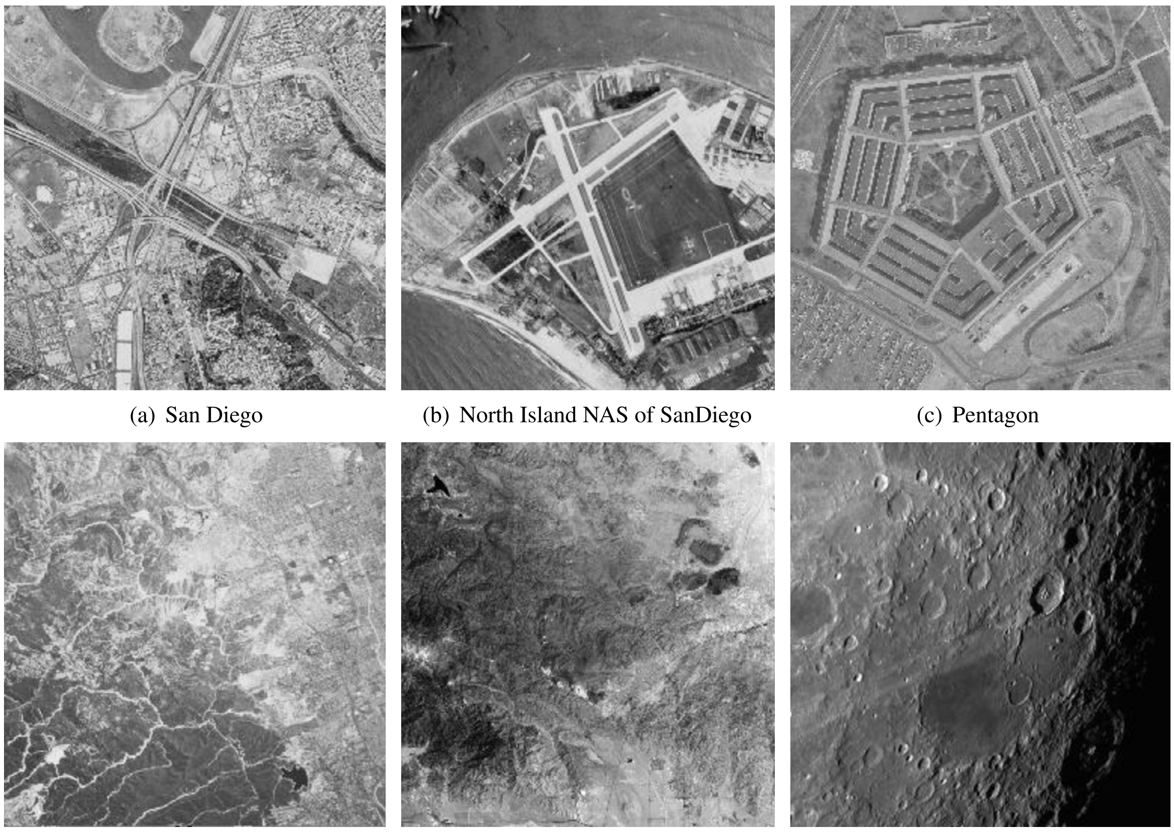
If the block size of BTOT is set to 64×64, then there are $2 \times 64 \times 64$ elements in the binary tree for each block of BTOT, which needs about 0.03M bytes with each element 4 bytes. Besides, it needs to store the final coding stream and some truncation points which need about 0.063M bytes, so the total memory requirement for BTOT is about 0.1M bytes. For BTCA, there are $2 \times 512 \times 512$ elements in the binary tree which needs 2M bytes, so the total memory requirement for BTCA is about 2.07M bytes. We can find that the memory requirement for BTOT is much less than that for BTCA.

4.3. The features of the coding stream

BTCA provides the quality scalability, because BTCA is an embedded coding method and it can stop coding or decoding at any specified rate. However, BTCA cannot provide the random access property, because the decoder of BTCA has to decode each region in turn. On the contrary, BTOT not only provides the quality scalability, but also has the random access property. Because BTOT exports the code length of each block, the decoder of BTOT can decode any specified blocks. Moreover, if we restrict each block within a wavelet subband, then BTOT can provide resolution scalability, which means the decoder can decode any specified resolution.

5. Experimental results

In this section, we conduct some experiments on three remote sensing image sets: the USC-SIPI Image Database [35], the Landsat images [36] and the CCSDS image test corpus [37], to evaluate the performance of the proposed BTOT method. The USC-SIPI image database contains four volumes. The Aerials volume of the database consists of 38 images. We select three images, i.e., San Diego, North Island NAS of San Diego and Pentagon, for exper-



(a) San Diego (b) North Island NAS of SanDiego (c) Pentagon

(d) Hills (e) Fires (f) Lunar

Fig. 4. Remote sensing images for experiment.

Table 3
The mean PSNRs of the proposed BTOT method at five bit rates for different block sizes.

Block size	PSNR				
	1 bpp	0.5 bpp	0.25 bpp	0.125 bpp	0.0625 bpp
16×16	32.29	28.79	26.32	24.50	22.88
32×32	**32.44**	28.96	26.52	24.74	23.41
64 × 64	32.43	**28.97**	26.56	**24.79**	**23.44**
128×128	32.43	28.92	**26.72**	24.78	23.35

iment, as shown in Fig. 4(a), (b), (c). The Landsat images were produced by the Landsat-7 flight, including eight spectral bands and one panchromatic band. Experiments are conducted on two images: Hills and Fires, as shown in Fig. 4(d), (e). The CCSDS reference test image set includes a variety of space imaging instrument data such as solar, stellar, planetary, earth observations, etc. We use the Lunar image for experiment, as shown in Fig. 4(f).

The images for experiment include different kinds of remote sensing images. All these images are converted into gray ones and size of $512 \times 512 \times 8$ before compression. In addition, we use 9/7-tap biorthogonal wavelet filters [38] for wavelet transform. The size of each block of the proposed BTOT method is 64×64. The Peak-Signal-to-Noise-Ratios (PSNR) for different Compression Ratios (CR) of our method are compared to those of SPIHT [5], SPECK [6], CCSDS [12], JPEG2000 [11], QTC [39], BTCA [26], BTCAS [26] and HAS [28]. The MATLAB source codes for BTCA and the proposed BTOT method are available.[1]

The PSNR and CR are expressed by the following relations [40]:

$$PSNR(\text{dB}) = 10\log_{10}\frac{255^2}{MSE}, \tag{9}$$

$$MSE = \frac{\sum_{i=1}^{W}\sum_{j=1}^{H}(x_{ij}-y_{ij})^2}{W \times H}, \tag{10}$$

$$CR(\text{bpp}) = \frac{\text{number of coded bits}}{W \times H}, \tag{11}$$

where x and y denote the original image and reconstructed one, respectively, and the images are of size $W \times H$.

Besides PSNR, we also use the Structural Similarity Index Measure (SSIM) [41] and Visual Information Fidelity (VIF) [42] criterion to evaluate the performance, which are considered to be correlated with the quality perception of the human visual system. The SSIM is defined as:

$$SSIM = l(x,y)c(x,y)s(x,y), \tag{12}$$

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2\mu_y^2 + C_1}, \tag{13}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2\sigma_y^2 + C_2}, \tag{14}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}, \tag{15}$$

where the detail parameters can be found in [41].

The model of VIF is defined as follows.

$$Y(i,k) = \beta(i,k)X(i,k) + V(i,k) + W(i,k) \tag{16}$$

$$\chi(i,k) = X(i,k) + W(i,k) \tag{17}$$

where $X(i,k)$ is the original image, $\chi(i,k)$ is the image captured by human, $W(i,k)$ is the noise of human vision system

[1] **MATLAB code:** http://www.mathworks.com/matlabcentral/profile/authors/5133554-ke-kun-huang/.

Table 4
PSNRs of the proposed BTOT method and other algorithms at five bit rates for different remote sensing images.

Image	Algorithm	PSNR				
		1 bpp	0.5 bpp	0.25 bpp	0.125 bpp	0.0625 bpp
SanDiego	SPIHT	26.61	23.37	21.40	20.08	18.89
	SPECK	26.73	23.57	21.80	20.44	19.45
	CCSDS	26.58	23.42	21.42	19.67	14.71
	JPEG2000	27.07	23.68	21.84	20.39	19.23
	QTC	26.83	23.63	21.83	20.45	19.45
	BTCA	26.93	23.76	21.87	20.53	19.56
	BTCAS	26.76	23.64	21.87	20.48	19.54
	HAS	26.93	23.79	21.88	20.50	19.52
	BTOT	**27.09**	**24.00**	**21.97**	**20.76**	**19.82**
NorthSanDiego	SPIHT	33.44	29.70	26.76	24.18	21.82
	SPECK	33.45	29.76	27.02	24.84	23.03
	CCSDS	33.21	29.48	26.46	22.70	15.54
	JPEG2000	33.81	29.92	27.01	24.69	22.46
	QTC	33.60	29.86	27.10	24.89	23.05
	BTCA	33.84	30.16	**27.39**	25.09	23.25
	BTCAS	33.47	29.81	27.07	24.82	23.03
	HAS	33.80	**30.42**	27.38	25.06	23.22
	BTOT	**33.95**	30.17	27.38	**25.13**	**23.26**
Pentagon	SPIHT	34.45	30.64	28.01	26.04	24.10
	SPECK	34.39	30.70	28.44	26.65	25.20
	CCSDS	34.47	30.60	28.04	25.36	18.17
	JPEG2000	**35.08**	31.11	28.54	26.57	24.73
	QTC	34.54	30.79	28.49	26.67	25.17
	BTCA	34.85	31.09	28.60	26.80	25.37
	BTCAS	34.81	31.02	28.58	26.70	25.21
	HAS	34.96	**31.13**	28.60	26.81	25.27
	BTOT	34.86	31.08	**28.67**	**27.06**	**25.43**
Hills	SPIHT	30.08	27.08	25.01	23.21	21.68
	SPECK	30.19	27.39	25.27	23.67	22.49
	CCSDS	30.34	27.40	25.13	22.88	17.78
	JPEG2000	**30.91**	27.60	25.32	23.62	22.22
	QTC	30.36	27.49	25.31	23.71	22.50
	BTCA	30.58	27.57	25.44	23.86	**22.60**
	BTCAS	30.50	27.54	25.39	23.78	22.49
	HAS	30.51	27.59	25.58	23.83	22.55
	BTOT	30.63	**27.77**	**25.72**	**23.95**	**22.60**
Fires	SPIHT	31.25	28.38	26.17	24.59	23.22
	SPECK	31.44	28.48	26.50	25.01	23.95
	CCSDS	31.42	28.44	26.34	24.46	19.88
	JPEG2000	**31.91**	28.63	26.58	24.92	23.71
	QTC	31.53	28.56	26.55	25.08	23.97
	BTCA	31.65	28.67	26.68	25.17	24.10
	BTCAS	31.61	28.60	26.60	25.14	24.01
	HAS	31.61	28.70	26.67	25.18	24.02
	BTOT	31.77	**29.02**	**26.85**	**25.22**	**24.12**
Lunar	SPIHT	35.56	31.13	28.04	25.48	23.28
	SPECK	35.63	31.30	28.54	26.26	24.49
	CCSDS	35.46	30.98	27.79	24.17	14.71
	JPEG2000	**36.34**	31.75	28.64	26.10	24.12
	QTC	35.88	31.42	28.61	26.46	24.54
	BTCA	36.15	31.73	**28.78**	26.56	24.74
	BTCAS	36.00	31.62	28.67	26.42	24.55
	HAS	36.08	31.76	28.77	26.51	24.70
	BTOT	36.25	**31.76**	**28.78**	**26.61**	**25.42**

and $Y(i,k)$ is the distorted image. $X(i,k)$ is decomposed into several subbands $X_1(i,k), X_2(i,k), \cdots, X_m(i,k)$ by wavelet transformation. Suppose the variance of $W(i,k)$ is σ_W, $c = 0.01$ and $Z(i,k)^2 = \frac{1}{m}\sum_{j=1}^{M} X_j(i,k)^2$, then

$$VIF = \frac{log_2(\frac{\beta(i,k)^2 Z(i,k)^2 + \sigma_W + c}{\sigma_W + c})}{log_2(\frac{Z(i,k)^2 + c}{c})}. \tag{18}$$

For the proposed method, there is only one parameter, i.e., the block size, need to select. Table 3 shows the mean PSNRs of the proposed BTOT method at five bit rates for different block sizes, where the PSNR at each bit rate is the mean value for six testing image. We can find that the performance for the block size 16×16 is always the lowest. When the block size is too small, there will be many continuous insignificant block, so the compression performance is limited. The results for other block sizes are similar. Considering the position scalability, we select the block size 64×64 for the proposed method.

Table 4 lists the PSNRs of the proposed method and other algorithms at five bit rates for different remote sensing images. From the results, we can draw the following conclusions:

- CCSDS almost attains the worst results for each image and each bit rate, because it specifically targets onboard mission and focuses more on complexity and less on compression performance.

Table 5
SSIMs of the proposed BTOT method and other algorithms at five bit rates for different remote sensing images.

Image	Algorithm	SSIM				
		1 bpp	0.5 bpp	0.25 bpp	0.125 bpp	0.0625 bpp
SanDiego	SPIHT	0.9601	0.9130	0.8578	0.7945	0.7104
	CCSDS	0.9600	0.9140	0.8569	0.7648	0.4152
	BTCA	0.9631	0.9213	0.8746	0.8190	0.7643
	BTCAS	0.9616	0.9183	0.8716	0.8153	0.7591
	HAS	0.9635	0.9211	0.8731	0.8202	0.7606
	BTOT	**0.9645**	**0.9256**	**0.8748**	**0.8296**	**0.7770**
Hills	SPIHT	0.9813	0.9623	0.9375	0.9030	0.8575
	CCSDS	0.9824	0.9648	0.9392	0.8931	0.6929
	BTCA	0.9833	0.9661	0.9438	0.9178	**0.8864**
	BTCAS	0.9829	0.9658	0.9429	0.9149	0.8832
	HAS	0.9830	0.9664	0.9454	0.9166	0.8858
	BTOT	**0.9835**	**0.9676**	**0.9474**	**0.9184**	0.8862
Fires	SPIHT	0.9894	0.9771	0.9650	0.9489	0.9282
	CCSDS	0.9897	0.9794	0.9662	0.9468	0.8503
	BTCA	0.9890	0.9805	0.9689	0.9554	**0.9422**
	BTCAS	0.9901	0.9802	0.9682	0.9550	0.9408
	HAS	0.9902	0.9807	0.9688	0.9557	0.9411
	BTOT	**0.9905**	**0.9820**	**0.9700**	**0.9559**	0.9421
Lunar	SPIHT	0.9960	0.9890	0.9773	0.9585	0.9296
	CCSDS	0.9959	0.9885	0.9758	0.9424	0.5812
	BTCA	0.9965	0.9904	**0.9809**	0.9677	0.9505
	BTCAS	0.9964	0.9901	0.9803	0.9666	0.9479
	HAS	0.9965	0.9904	0.9809	0.9675	0.9503
	BTOT	**0.9966**	**0.9905**	0.9808	**0.9681**	**0.9580**
Average	SPIHT	0.9817	0.9604	0.9344	0.9012	0.8564
	CCSDS	0.9820	0.9617	0.9345	0.8868	0.6349
	BTCA	0.9830	0.9646	0.9421	0.9150	0.8858
	BTCAS	0.9828	0.9636	0.9407	0.9130	0.8827
	HAS	0.9833	0.9647	0.9421	0.9150	0.8845
	BTOT	**0.9838**	**0.9664**	**0.9433**	**0.9180**	**0.8908**

Table 6
VIFs of the proposed BTOT method and other algorithms at five bit rates for different remote sensing images.

Image	Algorithm	VIF				
		1 bpp	0.5 bpp	0.25 bpp	0.125 bpp	0.0625 bpp
SanDiego	BTCA	0.4481	0.3006	0.1910	0.1163	0.0683
	BTCAS	0.4554	0.2990	0.1891	0.1118	0.0652
	HAS	0.4525	0.2954	**0.1937**	0.1155	0.0638
	BTOT	**0.4635**	**0.3115**	0.1921	**0.1346**	**0.0833**
Fires	BTCA	0.5683	0.3841	0.2420	0.1419	0.0816
	BTCAS	0.5793	0.3862	0.2441	0.1437	0.0799
	HAS	**0.5847**	0.3840	0.2444	**0.1483**	0.0806
	BTOT	0.5789	**0.4106**	**0.2520**	0.1480	**0.0831**
Lunar	BTCA	0.7226	0.4728	0.3087	0.1817	0.1046
	BTCAS	0.7139	0.4753	0.3040	0.1810	0.1021
	HAS	**0.7235**	**0.4794**	0.3083	0.1856	0.1086
	BTOT	**0.7235**	0.4768	**0.3098**	**0.1901**	**0.1677**

- JPEG2000 in general achieves better performance than SPIHT or SPECK, especially at higher bit rates. However, JPEG2000 is too complex to be a standard for airborne missions.
- BTCA gets better performance than QTC, which proves that binary tree provides a more efficient image representation than quadtree.
- The PSNR of BTCAS is lower than that of BTCA, though BTCAS provides random access property and requires less memory.
- HAS generates an importance weighting mask according to the human visual characteristics before applying BTCA, but HAS cannot improve the PSNR compared with BTCA. The PSNRs of both algorithms are very similar.
- The proposed BTOT achieves the highest performance, especially significantly improves the performance compared with BTCA. For SanDiego image, the PSNR of BTOT increases an average of 0.64 dB, 0.31 dB, 1.55 dB, 0.27 dB, 0.27 dB, 0.18 dB, 0.25 dB, 0.19 dB than SPIHT, SPECK, CCSDS, JPEG2000, QTC, BTCA, BTCAS and HAS, respectively.

Table 5 lists the SSIMs of the proposed method and other algorithms. We can find that the SSIM results are similar to the PSNR results. At the same compression ratio, the higher the PSNR is, the better the SSIM usually is. For example, at 0.5 bpp for SanDiego image, the PSNR of BTOT is 24.00, which is the highest of all methods, and the SSIM is 0.9256, which is also the highest of all methods. The average SSIM of all images for each method at each bit rate is listed at the end of Table 5. It can be seen that the average of the proposed method is the highest of all methods at each bit rate.

Table 6 lists the VIFs of the proposed method and other algorithms. Except sometimes the VIF of HAS is the better, the performance of the proposed method is the best, which proves that

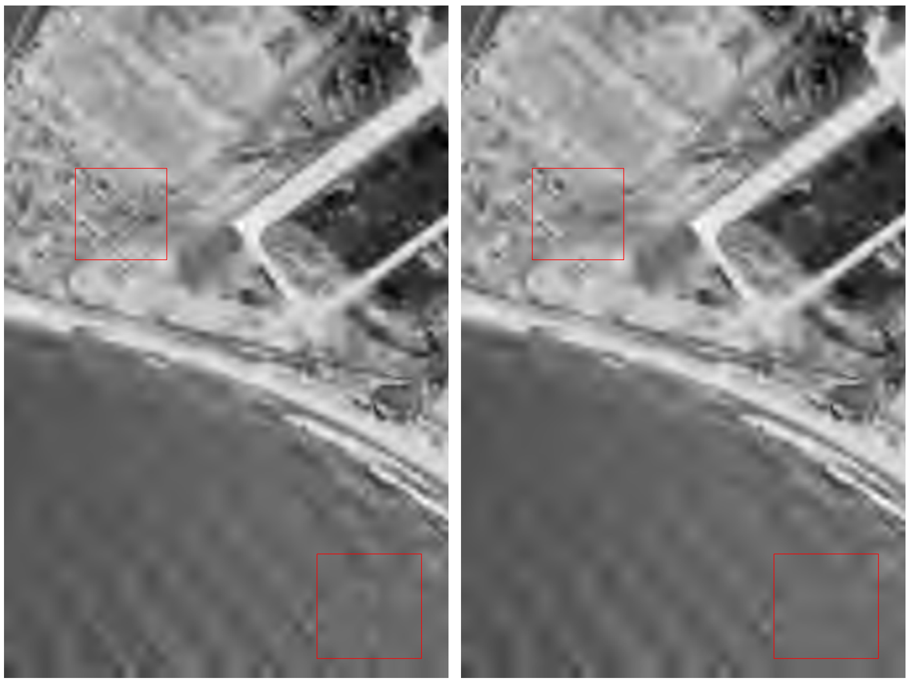

(a) Reconstructed image with BTOT
(b) Reconstructed image with SPIHT

Fig. 5. Comparison of two compression algorithms with the image of North Island NAS of San Diego at 0.25 bpp. The number of coded coefficients with BTOT is 12584, while it is 11627 with SPIHT. The reconstructed image with BTOT is clearer than that with SPIHT. For example, there are more textures in the red rectangles of the reconstructed image with BTOT.

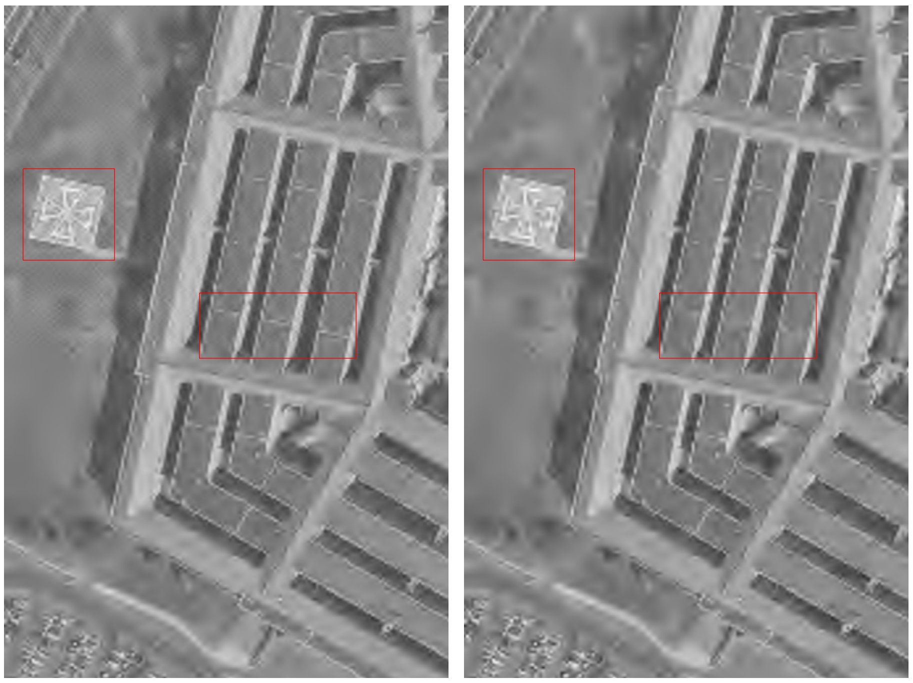

(a) Reconstructed image with BTOT
(b) Reconstructed image with SPIHT

Fig. 6. Comparison of two compression algorithms with Pentagon at 0.5 bpp. The number of coded coefficients with BTOT is 27452, while it is 19548 with SPIHT. The reconstructed image with BTOT is clearer than that with SPIHT. For example, the lines of the reconstructed image with BTOT are more clearer in the right rectangles.

HAS is a state-of-the-art method and the proposed method is the better.

Fig. 5 shows the comparison of two compression algorithms with the image of North Island NAS of San Diego at 0.25 bpp. The number of coded coefficients with BTOT is 12584, while it is 11627 with SPIHT. The reconstructed image with BTOT is clearer than that with SPIHT in the red rectangles. Fig. 6 gets a similar result with the Pentagon image.

Table 7 shows the comparison of PSNR, SSIM, complexity, memory requirement, coding time and other properties for different methods at the same compression ratio. SPIHT is implemented without arithmetic coding [43]. The program of CCSDS can be

Table 7
Comparison of different methods at the same compression ratio.

Method	SPIHT	CCSDS	BTCA	BTOT
CR	0.5 bpp	0.5 bpp	0.5 bpp	0.5 bpp
PSNR	28.38	28.39	28.83	28.97
SSIM	0.9639	0.9645	0.9677	0.9856
Complexity	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Memory requirement	1.5M	1.1M	2M	0.1M
Coding time	0.10	0.75	0.11	0.18
Quality scalability	yes	yes	yes	yes
Position scalability	no	no	no	yes
Resolution scalability	no	no	no	yes

download from [44]. The programs are evaluated on a computer with Intel Core i7-4910MQ CPU and 32 G memory. The PSNR, SSIM and coding time are the corresponding mean values for six testing images, respectively. We can find that the PSNR of BTOT is the highest, so does the SSIM. According to Eq. (7) and Eq. (8), the complexity of BTCA and BTOT is proportional to the number of image pixels n, so do those of SPIHT and CCSDS. BTOT is only a little slower than SPIHT, but is much faster than CCSDS, because CCSDS uses entropy coding. Based on the analysis in section 4.2, the memory requirement for BTOT is much less than that for BTCA. Moreover, BTOT provides quality, position and resolution scalability, while SPIHT, CCSDS and BTCA only provide quality scalability.

6. Conclusion

In this paper, we have proposed an efficient compression method for remote sensing image. We first divide the entire wavelet image to several blocks and encode them individually by BTCA, then apply rate-distortion optimization to get higher compression ratio and lower memory requirement, as well as the random access property. The proposed method is simple and easy to be implemented, so the proposed method is very suitable for remote sensing image. Furthermore, we can use wavelet packet transform to construct binary tree, design entropy coding and parallel algorithm to improve efficiency, which are our future works.

References

[1] A. Gunes, M.B. Guldogan, Joint underwater target detection and tracking with the Bernoulli filter using an acoustic vector sensor, Digit. Signal Process. 48 (1) (2016) 246–258.

[2] S. Uğur, O. Arıkan, A.C. Gürbüz, SAR image reconstruction by expectation maximization based matching pursuit, Digit. Signal Process. 37 (2) (2015) 75–84.

[3] M.M. Nielsen, Remote sensing for urban planning and management: the use of window-independent context segmentation to extract urban features in Stockholm, Comput. Environ. Urban Syst. 52 (7) (2015) 1–9.

[4] J.M. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, IEEE Trans. Signal Process. 41 (12) (1993) 3445–3462.

[5] A. Said, W.A. Pearlman, A new, fast, and efficient image codec based on set partitioning in hierarchical trees, IEEE Trans. Circuits Syst. Video Technol. 6 (3) (1996) 243–250.

[6] W.A. Pearlman, A. Islam, N. Nagaraj, A. Said, Efficient, low-complexity image coding with a set-partitioning embedded block coder, IEEE Trans. Circuits Syst. Video Technol. 14 (11) (2004) 1219–1235.

[7] A. Munteanu, J. Cornelis, G.V.D. Auwera, P. Cristea, Wavelet image compression – the quadtree coding approach, IEEE Trans. Inf. Technol. Biomed. 3 (3) (1999) 176–185.

[8] D. Taubman, High performance scalable image compression with EBCOT, IEEE Trans. Image Process. 9 (7) (2000) 1158–1170.

[9] D. Taubman, M. Marcellin, JPEG2000 Image Compression Fundamentals, Standards and Practice: Image Compression Fundamentals, Standards and Practice, vol. 642, Springer Science & Business Media, 2012.

[10] B. Li, R. Yang, H.X. Jiang, Remote-sensing image compression using two-dimensional oriented wavelet transform, IEEE Trans. Geosci. Remote Sens. 49 (1) (2011) 236–250.

[11] P. Kulkarni, A. Bilgin, M.W. Marcellin, J.C. Dagher, J.H. Kasner, T.J. Flohr, J.C. Rountree, Hyperspectral data compression, Ch. in: Compression of Earth Science Data with JPEG2000, Springer US, Boston, MA, 2006, pp. 347–378.

[12] [Online], Image data compression, Nov. 2005. Ser. Blue Book, CCSDS 122.0-b-1, http://public.ccsds.org/publications/BlueBooks.aspx.

[13] [Online], Consultative committee for space data systems, http://www.ccsds.org.

[14] F. Garcia-Vilchez, J. Serra-Sagrista, Extending the CCSDS recommendation for image data compression for remote sensing scenarios, IEEE Trans. Geosci. Remote Sens. 47 (10) (2009) 3431–3445.

[15] X.S. Hou, J. Yang, G.F. Jiang, X.M. Qian, Complex sar image compression based on directional lifting wavelet transform with high clustering capability, IEEE Trans. Geosci. Remote Sens. 51 (1) (2013) 527–538.

[16] H. ZainEldin, M.A. Elhosseini, H.A. Ali, A modified listless strip based SPIHT for wireless multimedia sensor networks, Computers and Electrical Engineering, http://dx.doi.org/10.1016/j.compeleceng.2015.10.001.

[17] V.B. Raju, K.J. Sankar, C.D. Naidu, S. Bachu, Multispectral image compression for various band images with high resolution improved DWT SPIHT, Int. J. Signal Process., Image Process. Pattern Recognit. 9 (2) (2016) 271–286.

[18] X.L. Tang, W.A. Pearlman Ch, Three-dimensional wavelet-based compression of hyperspectral images, in: Hyperspectral Data Compression, Springer US, Boston, MA, 2006, pp. 273–308.

[19] X.S. Hou, M. Han, C. Gong, X.M. Qian, SAR complex image data compression based on quadtree and zerotree coding in discrete wavelet transform domain: a comparative study, Neurocomputing 148 (2015) 561–568.

[20] M. Gaeta, V. Loia, S. Tomasiello, Cubic b–spline fuzzy transforms for an efficient and secure compression in wireless sensor networks, Inf. Sci. 339 (2016) 19–30.

[21] M. Rostami, O. Michailovich, Z. Wang, Gradient-based surface reconstruction using compressed sensing, in: IEEE International Conference on Image Processing (ICIP), 2012, pp. 913–916.

[22] M. Rostami, O.V. Michailovich, Z. Wang, Surface reconstruction in gradient-field domain using compressed sensing, IEEE Trans. Image Process. 24 (5) (2015) 1628–1638.

[23] X. Zhan, R. Zhang, D. Yin, C. Huo, Sar image compression using multiscale dictionary learning and sparse representation, IEEE Geosci. Remote Sens. Lett. 10 (5) (2013) 1090–1094.

[24] W. Fu, S. Li, L. Fang, J.A. Benediktsson, Adaptive spectral-spatial compression of hyperspectral image with sparse representation, IEEE Trans. Geosci. Remote Sens. 99 (2016) 1–12, http://dx.doi.org/10.1109/TGRS.2016.2613848.

[25] P.A.M. Oliveira, R.J. Cintra, F.M. Bayer, S. Kulasekera, Low-complexity image and video coding based on an approximate discrete Tchebichef transform, IEEE Trans. Circuits Syst. Video Technol. (2016) 1–10.

[26] K.K. Huang, D.Q. Dai, A new on-board image codec based on binary tree with adaptive scanning order in scan-based mode, IEEE Trans. Geosci. Remote Sens. 50 (10) (2012) 3737–3750.

[27] H. Liu, K.K. Huang, Embedded image compression based on fractal and wavelet, Int. J. Digit. Content Technol. Appl. 8 (2) (2014) 133–139.

[28] C.P. Shi, J.P. Zhang, Y. Zhang, A novel vision-based adaptive scanning for the compression of remote sensing images, IEEE Trans. Geosci. Remote Sens. 54 (3) (2016) 1336–1348.

[29] C.P. Shi, J.P. Zhang, Y. Zhang, Content-based onboard compression for remote sensing images, Neurocomputing 191 (2016) 330–340.

[30] S.G. Wang, M.C. Zhou, Z.W. Li, C.Y. Wang, A new modified reachability tree approach and its applications to unbounded Petri nets, IEEE Trans. Syst. Man Cybern. Syst. 43 (4) (2013) 932–940.

[31] J.T. Liu, C.H. Sui, D.W. Deng, J.W. Wang, B. Feng, W.Y. Liu, C.H. Wu, Representing conditional preference by boosted regression trees for recommendation, Inf. Sci. 327 (2016) 1–20.

[32] Y.-L. Chen, C.-C. Wu, K. Tang, Time-constrained cost-sensitive decision tree induction, Inf. Sci. 354 (2016) 140–152.

[33] E. Shusterman, M. Feder, Image compression via improved quadtree decomposition algorithms, IEEE Trans. Image Process. 3 (2) (1994) 207–215.

[34] C.A. Shaffer, R. Juvvadi, L.S. Heath, Generalized comparison of quadtree and bintree storage requirements, Image Vis. Comput. 11 (7) (1993) 402–412.

[35] [Online], The USC-SIPI image database, http://sipi.usc.edu/database/.

[36] [Online], United States geological survey, landsat project website, http://landsat.usgs.gov/.

[37] [Online], Consultative committee for space data systems, CCSDS image test, http://cwe.ccsds.org/sls/docs/sls-dc/.

[38] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies, Image coding using wavelet transform, IEEE Trans. Image Process. 1 (2) (1992) 205–220.

[39] S.T. Hsiang, J.W. Woods, Embedded image coding using zeroblock of subband/wavelet coefficients and context modeling, in: Data Compression Conf., Washington, DC, 2001, pp. 83–92.

[40] H. Liu, K.-K. Huang, Zerotree wavelet image compression with weighted sub-block-trees and adaptive coding order, Int. J. Wavelets Multiresolut. Inf. Process. 14 (4) (2016) 1650021.

[41] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 8 (03) (2004) 600–612.

[42] H.R. Sheikh, A.C. Bovik, Image information and visual quality, IEEE Trans. Image Process. 15 (2) (2006) 430–440.

[43] [Online], Center for image processing research, http://ipl.rpi.edu/.

[44] [Online], University of Nebraska-Lincoln, http://hyperspectral.unl.edu/.

Ke-Kun Huang received the B.Sc., M.Sc. and Ph.D. degrees in applied mathematics from Sun Yat-sen University, Guangzhou, China, in 2002, 2005 and 2016, respectively. He is currently an Associate Professor with the Department of Mathematics, Jiaying University, Meizhou, China. His research interests include image processing and face recognition.

Hui Liu received the B.Sc. degree and the M.Sc. degree in mathematics from South China Normal University, Guangzhou, China, in 2000 and 2006, respectively. She is currently an Associate Professor with the Department of Mathematics, Jiaying University, Meizhou, China. Her current research interests include fractal and image processing.

Chuan-Xian Ren received the Ph.D. degree from the Faculty of Mathematics and Computing, Sun Yat-Sen University, Guangzhou, China, in 2010. He is currently an Associate Professor of the Faculty of Mathematics and Computing, Sun Yat-sen University. His current research interests include image processing and face recognition.

Yu-Feng Yu received the B.Sc. degree in mathematics from Shangrao Normal University, Shangrao, China, in 2011, and the M.Sc. degree in mathematics from Shenzhen University, Shenzhen, China, in 2014. He is currently pursuing the Ph.D. degree in statistics in Sun Yat-Sen University. His research interests include statistical optimization and pattern recognition.

Zhao-Rong Lai received the B.Sc. degree in mathematics, the M.Sc. degree in computational science, and the Ph.D. degree in statistics from Sun Yat-sen University, Guangzhou, China, in 2010, 2012, and 2015, respectively. He is currently with the department of Mathematics, JiNan University. His research interests include face recognition and statistical optimization.