Lossless Compression of Bilevel ROI Maps of Hyperspectral Images Using Optimization Algorithms

Reetu Hooda and W. David Pan, *Senior Member, IEEE*

***Abstract*— In this letter, we propose a novel scheme for the compression of the bilevel region of interest (ROI) maps of hyperspectral images. The scheme can achieve high compression ratios by novel ways of partitioning the intricate regions of the image into smaller blocks. We first analyzed the effect of a low-order linear predictive model on the original image. The analysis showed that linear prediction results in lower-entropy residual images compared to the original images, which in turn helps in improving the compression efficiency. We then use the optimization algorithm that finds the best combination of scan directions for the nonzero blocks. It is shown to offer increasingly better compression with additional iterations. The simulation results on various data sets show that our scheme outperforms the international standard for binary image compression [joint bi-level image expert (JBIG2)].**

***Index Terms*— Bilevel images, discrete particle swarm optimization (DPSO), lossless compression, region of interest (ROI) maps.**

I. Introduction

HYPERSPECTRAL imaging is used in a wide range of applications such as remote sensing, medical diagnose, biotechnology, and surveillance. A hyperspectral image is a 3-D cube with spectral and spatial information providing discrimination capabilities. The high volume of hyperspectral data collected by the sensors put a strain on storage and transmission. Therefore, efficient compression of hyperspectral data has become an important concern [1], [2]. To address the main challenges of limited storage space and transmission bandwidth, we can achieve significant size reduction by preserving the region of interest (ROI) [3]. An ROI map indicates the location of pixels that belong to the region of specific interest to a certain applications such as medical imaging and nuclear medicine [4], [5]. We can define an ROI by creating a bilevel map, which is a binary image that is of the same size as the original image. An entry of “1” in the map means that the colocated pixel in the original hyperspectral image belongs to the ROI. A “0” entry means otherwise [6]. Due to the irregular nature of the ROI maps, achieving high compression on such images can be challenging and very critical in applications with limited bandwidth.

Although entropy coding such as Huffman coding, run-length coding, and context modeling form the basic steps employed by most state-of-the-art lossless compression techniques, in recent years, neural network (NN)-based solutions have become popular in the hyperspectral image compression field achieving results that are comparable to conventional methods. To reduce the training time complexity shallow NNs have been used to perform adaptive filtering for better prediction [7]. Long short-term memory (LSTM) could also be used to accommodate long-term data dependence in hyperspectral imagery for reducing prediction errors [8]. Among conventional methods, incorporation of deep belief network has been used for parameter estimation of Golomb-rice coding that offers a reduction in computations over brute force search methods [2]. Whereas pure conventional method involves band reordering and regrouping to introduce spectral redundancy [9]. And since ROI maps fall under the category of binary images, international standards include joint bi-level image expert (JBIG2) and JPEG2000 [10]. Other popular schemes include rectangular partitioning-based algorithm [11], [12] designed for images containing graphs and tables, and prediction algorithms that are used to improve the redundancy to achieve higher compression gains [13].

To the best of our knowledge, the proposed scheme is a first attempt tailored specifically for compression of ground truth, classification maps, and ROI maps, which are employed in remote sensing applications and saliency detection. The main contributions of the proposed method are as follows:

1) A fixed prediction algorithm is used as a preprocessing step to exploit the spatial relationship of adjacent pixels in ROI maps to offer more compressibility.
2) Discrete Particle swarm Optimization (DPSO) is used. It adapts to the nonuniform nature of the ROI maps and finds the absolute optimal scanning direction for efficient compression.

Although the full search technique is the most simplistic search method that guarantees to find the absolute optimal solution, it is an exhaustive method with large computational complexity, which has a limited effect when compressing a single image [14], [15].

The rest of the letter is organized as follows. In Section II, we illustrate the prediction scheme with details of the image grid patterns and scanning directions employed. Section III

Manuscript received September 30, 2020; revised November 12, 2020; accepted December 28, 2020. *(Corresponding author: Reetu Hooda.)*

The authors are with the Electrical and Computer Engineering Department, The University of Alabama in Huntsville, Huntsville, AL 35899 USA (e-mail: rh0059@uah.edu).

Color versions of one or more figures in this letter are available at https://doi.org/10.1109/LGRS.2021.3049164.

Digital Object Identifier 10.1109/LGRS.2021.3049164

$x(\mathrm{i}-1,j-1)$	$x(\mathrm{i}-1,j)$	$x(\mathrm{i}-1,j+1)$
$x(\mathrm{i},j-1)$	$\hat{x}(\mathrm{i},j)$	

Fig. 1. Ordering of pixels based on euclidean distance.

discusses the problem statement, and Section IV comprises all the steps of the optimization algorithm used. In Section V, the experimental results are provided. The letter is concluded in Section VI.

II. Preprocessing

The correlation amongst the neighboring pixel leads to redundancy, which is a common characteristics shared by most ROI images. Binary image compression techniques are concerned with exploiting these redundancies present in the image, which can be quantitatively described by "Entropy." An image with a higher entropy typically means the image is harder to compress [16]. Pixel prediction leads to residuals that tend to have lower entropy than the original image. Further compression is achieved by partitioning the image using an optimization algorithm. The preprocessing step is described in two sections as follows:

A. Predictive Model

Although it is advantageous to have high order predictor, for practical implementation we have limited the order to a small number (e.g., $N = 4$), to maintain lower complexity and less overhead [17]. The ordering of the pixels is based on Euclidean distance from the target pixel as shown in Fig. 1. We use $x(i, j-1)$, $x(i-1, j)$, $x(i-1, j-1)$ and $x(i-1, j+1)$ (neighboring pixels) to predict the target pixel $x(i, j)$. The predicted pixel $\hat{x}(i, j)$ is a linear combination of its causal pixels and a bias. In the case of Nth order predictor, we define $N+1$-dimensional vector. From Fig. 1, for $N = 4$ we consider 5-D vector as $\{1, x(i, j-1), x(i-1, j), x(i-1, j-1), x(i-1, j+1)\}$.

The predicted pixel can be expressed as: $\hat{x}(i, j) = \alpha_0 + \alpha_1 x(i, j-1) + \alpha_2\, x(i-1, j) + \alpha_3\, x(i-1, j-1) + \alpha_4\, x(i-1, j+1)$. where $\alpha's$ are real numbers called *prediction coefficients*.

B. Numerical Solution

The least square (LS) optimization is used to calculate the optimal prediction coefficient [18], [19]. Given the size of the image being $m \times n$, our objective is to find LS solution for the system

$$P\alpha = y \tag{1}$$

where P is an $(mn) \times (N+1)$ matrix shown in Fig. 2, with each row corresponding to 4 neighboring pixels and a bias associated with each target pixel. $\alpha = [\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4]^T$ are prediction coefficients to be determined, and $y = [x(1, 1) \ldots x(i, j) \ldots x(m, n)]^T$ is an mn-dimensional vector with target pixels.

From 1 we have:

$$P^T P\alpha = P^T y. \tag{2}$$

$$P = \left[\begin{array}{ccccc} 1 & x(1,0) & x(0,1) & x(0,0) & x(0,2) \\ 1 & x(1,1) & x(0,2) & x(0,1) & x(0,3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x(1,j-1) & x(0,j) & x(0,j-1) & x(0,j+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x(1,n-1) & x(0,n) & x(0,n-1) & x(0,n+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x(i,j-1) & x(i-1,j) & x(i-1,j-1) & x(i-1,j+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x(m,n-1) & x(m-1,n) & x(m-1,n-1) & x(m-1,n+1) \end{array}\right] \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \\ \\ \end{array}\right\} 1^{st}\ Row \\ \\ \}\ i^{th}\ Row \\ \\ \}\ m^{th}\ Row \end{array}$$

Fig. 2. P matrix.

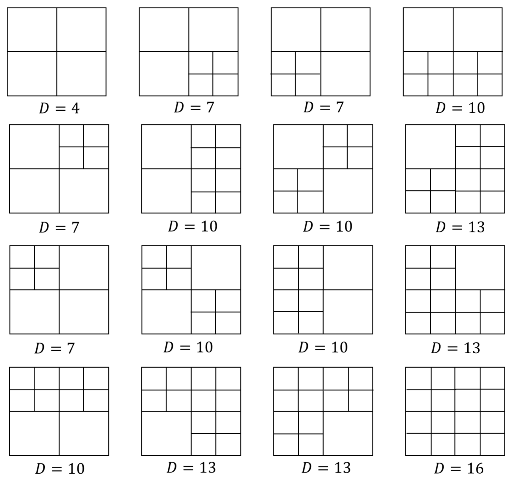

Fig. 3. 16 possible grid patterns for partitioning an image.

Equation 2 has a unique solution

$$\alpha = (P^T P)^{-1} P^T y. \tag{3}$$

Using the optimal solution, $\hat{x}(i, j)$ where $1 \le i \le m, 1 \le j \le n$ is calculated, a predicted value for $x(i, j)$. Thereafter, the output $\hat{x}(i, j)$ of the predictor module is compared with the original value $x(i, j)$ to construct an error (residual) image as follows:

$$E(i, j) = \hat{x}(i, j) \oplus x(i, j) \tag{4}$$

where $\oplus$ is modulo 2 addition, which is realized by an XOR operation. Finally, the lower entropy error image $E(i, j)$ with optimal predictor coefficients α is fed to the optimization algorithm discussed in Section II-C for further compression.

C. Grid Patterns

To take advantage of long-range and short-range correlations, the error image is divided into variable-size blocks based on its statistics. The idea is to partition highly actively changing portion of the image into smaller blocks and have larger blocks for less complicated regions in the image to divide pixel-concentration. First, the image is divided into 4 equally sized blocks and then further division takes place by partitioning each nonzero subblock into 4 subblocks. We examine a total of 16 patterns as shown in Fig. 3. An image

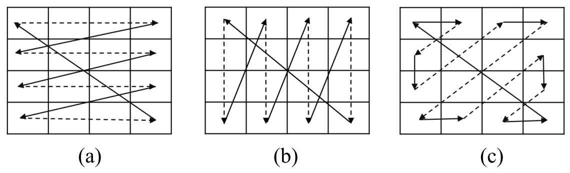

Fig. 4. Scanning directions. (a) Horizontal scan. (b) Vertical scan. (c) Zig-zag scan.

can be segmented, at least into 4 blocks while the maximum number of blocks can go to 16.

D. Scanning Directions

Image pixels are scanned following a certain direction to convert the image block to a 1-D vector. As the distribution of pixels is nonhomogeneous in ROI maps, three different scanning directions are used [20]. An image block can be scanned in horizontal, vertical, or zig-zag raster scan direction, generating different intervals for different scanning path as shown in Fig. 4.

Interval Generation: The interval sequence generated affects the degree of compression. It is calculated as the distance between the previous and next occurrence of the same bit "1" (referred to as *Intervals*) following a scanning direction mentioned above.

III. Problem Statement

An image of size $m \times n$ is divided into D blocks where $D \in \{4, 7, 10, 13, 16\}$ as shown in Fig. 3 and each block can be scanned in one of the three scanning directions to exploit horizontal, vertical, and diagonal redundancy in an image. Therefore, the search complexity increases as 3^D. The value of D determines the dimensions of the search space as $S \in \{3^4, 3^7, 3^{10}, 3^{13}, 3^{16}\}$. A full search method to find the optimal combination of scanning direction out of $S \in \{81, 2187, 59\,049, 1\,594\,323, 43\,046\,721\}$ combinations for maximum compression is an exhaustive process and time consuming. Discrete particle swarm optimization (PSO) algorithm can be employed to find the best combination of the scanning directions to obtain the largest compression on the intervals.

Let us consider an illustrative example, where an image of the size of 12×12 takes up one of the patterns mentioned in Section II-C, by being divided into $D = 13$ blocks as depicted in Fig. 5. The scan path for all the blocks is referred to as *Direction Bits*. Now, let us assume the direction bits shown in Fig. 6(a), as the optimal one for this image. The direction bit is set to "0," if the horizontal direction is chosen for a block; a direction bit of "1" is set for vertical scanning patterns; otherwise, a direction bit of "2" is set, implying zig-zag direction. The interval sequence is generated by scanning the blocks accordingly.

IV. Algorithm

The goal is to search for the best combination of scanning direction using DPSO to minimize the compressed file size. It was inspired by bird flocking or fish schooling behavior. In a DPSO system, each particle represents the candidate solution to the optimization problem and a swarm of particles is considered to fly through the search space. The position or movement of the particles is guided by the experience of the neighboring particles and experience of the entire swarm [21]. A particle's best position is termed as P_{best} and the global best of the entire swarm is termed as G_{best}. Fitness functions are used to measure the performance of each particle [22].

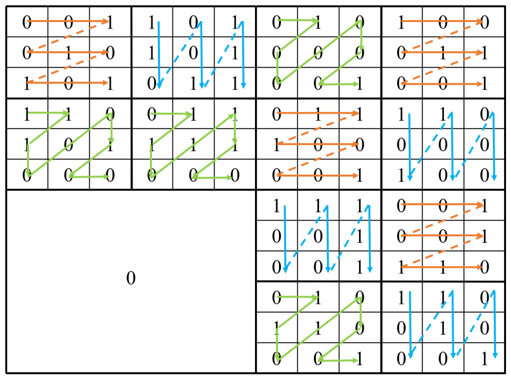

Fig. 5. Image (12×12) divided into smaller blocks.

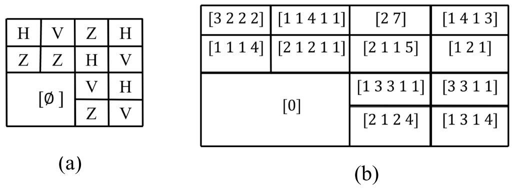

Fig. 6. Bitmap and interval generation. (a) Direction bits. (b) Interval sequences.

A. Parameter Initialization

The particles in the DPSO model are initialized randomly with a vector representing the scanned path which is discrete in nature, since a block can be scanned in only three directions, i.e., H, V, or Z. The dimension of the vector depends on the total number of blocks an image is partitioned into.

B. DPSO Model

DPSO is initialized with a swarm size of N_{pop} particles. Each particle is treated as a point in the D-dimensional space, where D is the number of blocks the image is divided into. The ith particle is represented as $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, where $x_{id} \in \{0, 1, 2\}$ and $1 \le i \le N_{\text{pop}}$, $1 \le d \le D$. The velocity of particle i is represented as $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. The previous best position of the ith particle is represented as $p_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$ and the best among the entire swarm is represented by $p_g = (p_{g1}, p_{g2}, \ldots, p_{gD})$.

At each step, DPSO changes the velocity of each particle toward its P_{best} and G_{best} location. The velocity of a particle can be updated according to the following equation:

$$v_{id} = \omega v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \quad (5)$$

where ω represents the inertial weight used to control the search capability of the DPSO algorithm. Thus, in order to improve both global and local search capability of a particles,

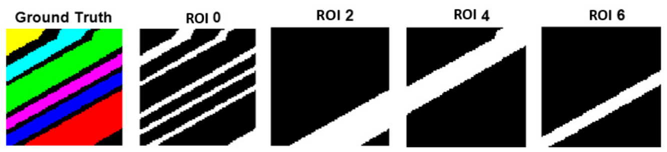

Fig. 7. SalinasA data set.

we choose a value of $\omega = 0.998$ for the momentum term after some tests and tuning.

And the position of the particle can be updated using

$$x_{id} = \begin{cases} 0, & \text{if } \sigma(v_{\text{id}}) < 0.5 \\ 1, & \text{if } \sigma(v_{\text{id}}) > 0.5 \\ 2, & \text{if } \sigma(v_{\text{id}}) = 0.5 \end{cases} \tag{6}$$

where

$$\sigma(v_{\text{id}}) = \frac{1}{1 + e^{v_{\text{id}}}}. \tag{7}$$

In 5, c_1 and c_2 are called acceleration coefficients that controls the effect of cognitive $(c_1\ r_1(p_{\text{id}} - x_{\text{id}}))$ and social component $(c_2\ r_2(p_{gd} - x_{\text{id}}))$ on the overall velocity, which are updated using the method shown in [23]. r_1, r_2 are random numbers uniformly distributed in the range of [0, 1]. t is the current iteration, whereas $\text{Iter}_{\max}$ is the maximum number of iterations.

C. Fitness Function

In the proposed method, a pattern is selected based on the statistics of the input image, for which we divide into nonuniform blocks. The blocks are scanned to generate intervals employing the directions recommended by the DPSO algorithm in its current iteration. If the direction bit is "0," "1," or "2," the block is scanned in horizontal, vertical, or zigzag direction, respectively. Finally, the sequence of intervals and direction bits are combined into a data file and fed to an entropy coder (arithmetic coder in our implementation) to output the compressed file size as the output of the fitness function, i.e., mapping the search space into a function space.

V. Results

To examine the effectiveness and performance of the proposed scheme, we tested our method on six various data sets listed in Table I. Fig. 7 shows SalinasA data set with ground truth and a few ROI maps.

The proposed algorithm was applied to the ROI maps and compared against five other lossless image compression scheme which includes Consultative Committee for International Telegraphy And Telephony (CCITT), Fax3, Fax4, and JPEG2000 and an international standard for bilevel images (JBIG2). The simulation results are shown in Fig. 8. It shows that our method achieves significantly higher compression for biased ROI maps (classes 1–6) and outperforms all the other methods on average. For a nonbiased ROI map (class 0), it gains compression efficiency comparable to JBIG2.

To more fully evaluate the performance of the proposed scheme, we have included the average bitrate and computing time in Table I with respect to the second best method (JBIG2).

TABLE I
Average Bitrate in Bits Per Pixel (BPP)

Dataset	ROIs	Proposed	Time(s)	JBIG2	Time(s)
IP 144 × 144	0:16 1:16	**0.0166** **0.0093**	∼ **0.21**	0.0644 0.0574	∼ 0.61
SalinasA 80 × 80	0:6 1:6	**0.0740** **0.0423**	∼ **0.22**	0.1841 0.1735	∼ 0.62
Salinas 512 × 208	0:16 1:16	**0.0141** **0.0063**	∼ 0.73	0.0154 0.0133	∼ **0.63**
PU 608 × 336	0:9 1:9	0.0397 0.0217	∼ 2.47	**0.0284** **0.0192**	∼ **0.64**
KSC 512 × 608	0:13 1:13	**0.0056** **0.0013**	∼ 3.78	0.0055 0.0046	∼ **0.65**
Botswana 1472 × 256	0:14 1:14	**0.0021** **0.0011**	∼ 5.02	0.0046 0.0038	∼ **0.66**

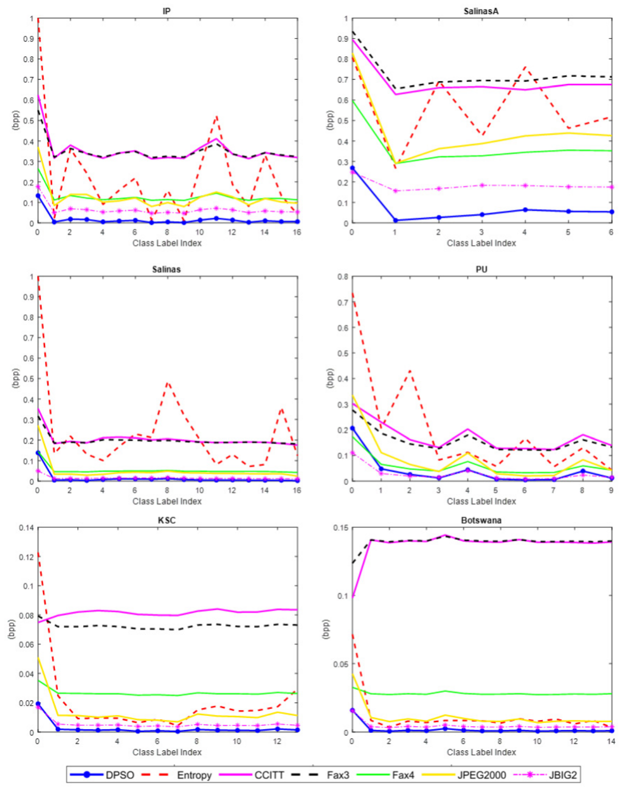

Fig. 8. Simulation results.

Since few of the ROI maps of Pavia University are not highly biased compared to other ROI maps, we expect that the proposed method will be a little less efficient and will be comparable to JBIG2. It achieves reasonable compression on average compared to other lossless schemes. More specifically, for ROI map 0, 1 and 8, which are globally scattered and regionally unbiased, the proposed algorithm achieves the second best compression efficiency, followed by Fax4. Similar improvements were observed for the other ROI maps as demonstrated in Table I.

Computational Complexity: Computational complexity of the proposed scheme can be divided into two subparts.

1) Preprocessing: Computation of optimal prediction coefficient (3) takes $O((N+1)^2\ mn)$ to multiply $P^T P$,

$O((N+1)mn)$ to multiply $P^T y$ and $O((N+1)^3$ to compute $P^T P^{-1}$. Now we know that $O((N+1)^2\, mn)$ dominates $O((N+1)mn)$ and since $mn > N+1$, $O((N+1)^2 mn)$ also dominates $O((N+1)^3)$. And computation of error image E has complexity of $O(mn)$. Therefore, the total time complexity of the preprocessing step is $O((N+1)^2 mn)$.

2) DPSO: The complexity of DPSO can be estimated using computation cost of initialization (T_{init}), main search algorithm (T_s) and fitness evaluation (T_{FE}) step. Since initialization of positions, velocity and computing personal best and global best is considered as one operation, T_{init} is $O(N_{\text{pop}})$. For the search algorithm, DPSO algorithm has an inner loop when going through population N_{pop} and an outer loop for iteration t. So the T_s at extreme case is $O({N_{\text{pop}}}^2\ t)$. Although the computation cost is relatively inexpensive, since T_s is linear in terms of t, the most computationally extensive part is the fitness evaluations (FEs). The complexity of FE is dependent on dimensionality of the search space D and size of the image (mn). Therefore, T_{FE} is $O(D \times mn)$, where D in our case $\in \{4, 7, 10, 13, 16\}$. As the size of the image increases, the overall time increases due to increase in T_{FE}, which is validated by recorded time in shown in Table I.

VI. Conclusion

We presented a new scheme for compression of ROI maps of hyperspectral images. It comprises of a preprocessing step that takes advantage of the correlations between the neighboring pixels and improves the sparsity of the ROI maps. In the next step, we divide the image into smaller blocks and use an optimization algorithm to choose an optimal scanning direction for each block, thereby resulting in a significant improvement in compression efficiency. The simulation results on six hyperspectral data sets show that the proposed algorithm has compression performance that is comparable or higher than other standard methods by significant percentages.

References

[1] H. Shen, W. D. Pan, Y. Dong, and Z. Jiang, “Golomb-rice coding parameter learning using deep belief network for hyperspectral image compression,” in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Fort Worth, TX, USA, Jul. 2017, pp. 2239–2242.

[2] Z. Jiang, W. D. Pan, and H. Shen, “Universal golomb–rice coding parameter estimation using deep belief networks for hyperspectral image compression,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 10, pp. 3830–3840, Oct. 2018.

[3] L. Zhou and S. Zahir, “A new efficient algorithm for lossless binary image compression,” in *Proc. Can. Conf. Electr. Comput. Eng.*, May 2006, pp. 1427–1431.

[4] H. Shen, W. D. Pan, and D. Wu, “Predictive lossless compression of regions of interest in hyperspectral images with no-data regions,” *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 1, pp. 173–182, Jan. 2017.

[5] A. Liaghati, W. D. Pan, and Z. Jiang, “Biased run-length coding of bilevel classification label maps of hyperspectral images,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 10, pp. 4580–4588, Oct. 2017.

[6] A. L. Liaghati, H. Shen, and W. David Pan, “An efficient method for lossless compression of bi-level ROI maps of hyperspectral images,” in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, Mar. 2016, pp. 1–6.

[7] Z. Jiang, W. D. Pan, and H. Shen, “Spatially and spectrally concatenated neural networks for efficient lossless compression of hyperspectral imagery,” *J. Imag.*, vol. 6, no. 6, p. 38, May 2020. [Online]. Available: https://www.mdpi.com/2313-433X/6/6/38

[8] Z. Jiang, W. D. Pan, and H. Shen, “LSTM based adaptive filtering for reduced prediction errors of hyperspectral images,” in *Proc. 6th IEEE Int. Conf. Wireless Space Extreme Environ. (WiSEE)*, Huntsville, AL, USA, Dec. 2018, pp. 158–162.

[9] H. Yu, Z. Zhang, B. Lei, and C. Wang, “Hyperspectral image compressing using wavelet-based method,” in *Proc. AOPC: Opt. Spectrosc. Imag.*, vol. 10461, J. Yu, Z. Wang, W. Hang, B. Zhao, X. Hou, M. Xie, and T. Shimura, Eds. Bellingham, WA, USA: SPIE, 2017, pp. 462–468, doi: 10.1117/12.2285781.

[10] S. Alzahir and A. Borici, “An innovative lossless compression method for discrete-color images,” *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 44–56, Jan. 2015.

[11] S. Zahir and M. Naqvi, “A new rectangular partitioning based lossless binary image compression scheme,” in *Proc. Can. Conf. Electr. Comput. Eng.*, May 2005, pp. 281–285.

[12] S. Zahir and M. Naqvi, “A near minimum sparse pattern coding based scheme for binary image compression,” in *Proc. IEEE Int. Conf. Image Process.*, vol. 2, Sep. 2005, pp. II–289.

[13] D. Novikov, N. Egorov, and M. Gilmutdinov, “Local-adaptive blocks-based predictor for lossless image compression,” in *Proc. XV Int. Symp. Problems Redundancy Inf. Control Syst. (REDUNDANCY)*, Sep. 2016, pp. 92–99.

[14] H. Radha, M. Vetterli, and R. Leonardi, “Image compression using binary space partitioning trees,” *IEEE Trans. Image Process.*, vol. 5, no. 12, pp. 1610–1624, Dec. 1996.

[15] S. Verma and A. K. Pandit, “Quad tree based image compression,” in *Proc. IEEE Region 10 Conf. (TENCON)*, Nov. 2008, pp. 1–4.

[16] S. Marusic and G. Deng, “Adaptive prediction for lossless image compression,” *Signal Process., Image Commun.*, vol. 17, no. 5, pp. 363–372, May 2002.

[17] H. Kobayashi and L. R. Bahl, “Image data compression by predictive coding I: Prediction algorithms,” *IBM J. Res. Develop.*, vol. 18, no. 2, pp. 164–171, Mar. 1974.

[18] X. Li and M. T. Orchard, “Edge-directed prediction for lossless compression of natural images,” *IEEE Trans. Image Process.*, vol. 10, no. 6, pp. 813–817, Jun. 2001.

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

[20] N. Memon, D. L. Neuhoff, and S. Shende, “An analysis of some common scanning techniques for lossless image coding,” *IEEE Trans. Image Process.*, vol. 9, no. 11, pp. 1837–1848, Nov. 2000.

[21] J. C. Bansal and K. Deep, “A modified binary particle swarm optimization for knapsack problems,” *Appl. Math. Comput.*, vol. 218, no. 22, pp. 11042–11061, Jul. 2012.

[22] A. Muruganandham and R. S. D. Wahida Banu, “Adaptive fractal image compression using PSO,” *Procedia Comput. Sci.*, vol. 2, pp. 338–344, 2010.

[23] J. Cai and W. David Pan, “On fast and accurate block-based motion estimation algorithms using particle swarm optimization,” *Inf. Sci.*, vol. 197, pp. 53–64, Aug. 2012.