3D Wavelet Transform with SPIHT algorithm for Image Compression

Mr. M. R. Zala¹, Mr. S. S. Parmar²

¹²³Computer Engineering Department, C. U. Shah College of Engineering & Technology, Wadhwan City, India.

ABSTRACT

Wavelet theory has great potential in color image compression. Our work is based on 3D discrete wavelet transform using SPIHT algorithm with PSO to compress sequence of color images simultaneously. 3D-SPIHT is the modern-day benchmark for three dimensional image compressions. The three-dimensional coding is based on the sequence of color images are contiguous and there is no motion between image slices. We propose a compression algorithm 3D SPIHT with PSO based on 3D discrete wavelet transform, 3D set partitioning into hierarchical tree, arithmetic coding and particle swarm optimization (PSO), which is developed using Matlab 7.8.0. Using 3D SPIHT with PSO algorithm we obtained improvement than standard SPIHT algorithm in terms of compression ratio (CR), mean-squared error (MSE), peak signal to noise ratio (PSNR), correlation coefficient and multiscale structural similarity index (MSSIM).

Keywords: 3D Discrete Wavelet Transform, 3D SPIHT, Particle Swarm Optimization, Compression Ratio, Peak Signal to Noise Ratio, Structural Similarity Index

1. INTRODUCTION

When we speak of image compression, basically there are two types: lossless and lossy[12], [13]. With lossless compression, the original image is recovered exactly after decompression. Much higher compression ratios can be obtained if some error, which is usually difficult to perceive, is allowed between the decompressed image and the original image. We attempt to discuss about image compression, compression of sequence of color Images using 3D wavelet transform [4], [5] based 3D SPIHT [6] with PSO algorithm. Particle Swarm Optimization (PSO) [8], [9] is computational method that optimize a problem by iteratively trying to improve performance with regard to given measure of quality. We have implemented a new image compression 3D SPIHT with PSO algorithm, in which arithmetic coding is mainly concern with to improve compression ratio of sequence of color image and particle swarm optimization (PSO) is mainly concern with to improve the fidelity criteria of images.

2. DISCRETE WAVELET TRANSFORM [1]

Figure 1 shows 1-D, 2-D, and 3-D decomposition of data. It assumes that each signal is broken down into 3 levels of resolution. The 1-D case shows that a 1-D signal of 32 values would be broken down into 16 first octave details, followed by 8 second octave details, followed by 4 details of the third octave, and finally 4 signal approximation values also generated in the third octave. The 2-D decomposition has 3 detail blocks in the first octave. These appear as the 3 largest blocks. The 3 blocks of the second octave are each one quarter the size of the remaining large block. The four smallest blocks represent the 3 details of the third octave, and the third octave approximation signal. The 3-D graphic shows how the first octave detail signals comprise 7/8 of the transformed data size. The second octave again breaks up the remaining data into 7 detail blocks and 1 approximation block. Finally, the third octave breaks the approximation block from the previous level into 7 details and 1 approximation. Three-dimensional data, such as a video sequence, can be compressed with a 3-D DWT, like the one shown in Figure 1. Most applications today perform a 2-D DWT on the individual images, and then encode the differences between images (called difference coding). Given the first image, followed by the differences between the first and second frame, the second image can be constructed.

The 3-D DWT [10] is like a 1-D DWT in three directions. Refer to Figure 2 first; the process transforms the data in the x-direction. Next, the low and high pass outputs both feed to other filter pairs, which transform the data in the y-direction. These four output streams go to four more filter pairs, performing the final transform in the z-direction. The process results in 8 data streams. The approximate signal, resulting from scaling operations only, goes to the next octave of the 3-D transform.
The 3D wavelet decomposition is computed by applying three separate 1D transforms along the coordinate axes of the sequence of image data. The 3D data is usually organized slice by slice. A single slice has rows and columns as in the 2D case, x and y direction often denoted as “spatial co-ordinates”, whereas for the sequence of image data, a third dimensions no. of slice is added (z-direction). The input data is a set of multiple slice each consisting of N rows and N columns. Hence the input data can be denoted as NxNxN, where N is an integer. The 3D DWT can be considered as a combination of three 1D DWT in the x, y and z directions as shown in Figure 2. A preliminary work in the DWT processor design is to build 1D DWT modules, which are composed of high-pass and low-pass filters that perform a convolution on filter coefficients and input pixels. After a one-level of 3D discrete wavelet transform, the volume of image data is decomposed into HHH, HHL, HLH, HLL, LHH, LHL, LLH and LLL signals as shown in Figure 2.

3. SET PARTITIONING INTO HIERARCHICAL TREE (SPIHT) ALGORITHM

The SPIHT algorithm [2] (Set Partitioning into Hierarchical Trees) was developed by Amir Said and William Pearlman in 1996. It is an image compression algorithm based on three concepts:

- Partial ordering of the transformed image elements by magnitude and transmission of this ordering information.
- Ordered bit plane transmission.
- Application of similarity between coefficients from different wavelet levels which describe the same origin.

3.1 Spatial Orientation Tree

A tree structure called spatial orientation tree [11] is therefore chosen to represent the transformed image, see Figure 3 each node of the tree corresponds to a coefficient and is identified by the coefficient’s coordinates. Its direct descendants represent the same spatial orientation in the next wavelet level, which has a more detailed resolution.
Therefore a node has either no direct descendants (the leaves) or four, which form a group of $2 \times 2$ adjacent coordinates. The only exceptions are the nodes at the highest hierarchy level where the coordinates in the low-low-band marked with a star do not have any descendants. Nodes at the same hierarchy level match the coordinates at the same wavelet level.

![Spatial Orientation Tree](image)

**Figure 3** Building the spatial orientation tree [11]: (a) Pyramidal form created by the wavelet transformation, (b) Relationship between offspring, (c) Transformed image represented as tree

### 3.2 SPIHT Algorithm

The following sets of coordinates are used in the algorithm [2], [3]:
- $H(i, j)$ is the set of coordinates of the tree roots, which are the nodes in the highest wavelet level
- $O(i, j) = \{(2i, 2j), (2i, 2j +1), (2i+1, 2j), (2i+1, 2j +1)\}$ is the set of coordinates of the children of node $(i, j)$
- $D(i, j)$ is the set of all descendants of node $(i, j)$
- $L(i, j) = D(i, j) \setminus O(i, j)$ is the set of descendants except the children of node $(i, j)$.

The function is used to indicate the significance of a set of coordinates $T$.

$$S_n(T) = \begin{cases} 1, & \text{if } \max_{(i,j) \in T} |c_{ij}| \geq 2^n \\ 0, & \text{else} \end{cases}$$  \hspace{1cm} (1)

To store the significance information three ordered sets are used. These sets are as given below:
- The coordinates of those coefficients, which are insignificant with respect to the current threshold, are contained in the list LIP of insignificant pixels.
- The coordinates of those coefficients, which are significant with respect to the current threshold, are contained in the list LSP of significant pixels.
- The coordinates of the roots of insignificant subtrees are contained in the list LIS of insignificant sets. The sets of coefficients in LIS are refined during compression and if the coefficients become significant they are moved from LIP to LIS.

### 3.3 3D SPIHT Algorithm [6]

1) Initialize to the number of bit planes

2) Set the LSP as an empty list, and add the coordinates $(i, j)$ to the LIP, and only those with descendants also to the LIS, as type A entries.

3) Sorting Pass:
   a. for each entry $(i, j, k)$ of the LIP
      i. If $S_n(i, j, k) = 1$,
         1. Move $(i, j, k)$ in LSP
         2. Output the sign of $c_{i,j,k}$
   b. for each entry $(i, j, k)$ of the LIS
      i. if the entry is type A then
         1. Output $S_n(D(i, j, k))$
         2. if $S_n(D(i, j, k)) = 1$ then
            a. for all $(i', j', k') \in O(i, j, k)$ do:
               i. if $S_n(i', j', k') = 1$ then
                  1. add $(i', j', k')$ to the LSP
                  2. Output the sign of $c_{i',j',k'}$
ii. else
1. add \((i', j', k')\) to the end of the LIP
b. if \(L(i, j, k) \neq \emptyset\).
i. move \((i, j, k)\) to the end of the LIS as a type B entry.
c. else,
i. remove \((i, j, k)\) from the LIS
3. if the entry is type B then
   a. output \(S_o(L(i, j, k))\)
   b. if \(S_o(L(i, j, k)) = 1\)
i. add all the \((i', j', k') \in O(i, j, k)\) to the end of the LIS as a type A entry
ii. remove \((i, j, k)\) from the LIS

4) Refinement Pass:
a. for all entries \((i, j, k)\) of the LSP, except those included in the last sorting pass:
i. output the \(n^{th}\) most significant bit of \(c_{ijk}\)

5) Quantization-Step Update: decrement \(n\) by 1 and go to Step 3.a).

4. PARTICLE SWARM OPTIMIZATION

Particle swarm optimization [8] is a heuristic global optimization method put forward originally by Doctor Kennedy and Eberhart in 1995. It is developed from swarm intelligence and is based on the research of bird and fish flock movement behavior. Basically particle swarm optimization algorithm, consists of “\(n\)” particles and position of each particle stands for the potential solution in \(D\) dimensional space. Each particle can be shown by its current velocity and position, the most optimist position of each individual and the most optimist position of the surrounding. In the partial PSO, the velocity and position of each particle change according the following equality [8], [9]:

\[
\begin{align*}
V_{i,d}^{k+1} &= \omega V_{i,d}^{k} + c_1 r_1 (p_{best,d}^{k} - X_{i,d}^{k}) + c_2 r_2 (g_{best,d}^{k} - X_{i,d}^{k}) \\
X_{i,d}^{k+1} &= X_{i,d}^{k} + V_{i,d}^{k+1}
\end{align*}
\]  

Let \(s\) denotes the swarm size. Each particle \(1 \leq i \leq s\) is characterized by three attributes [9]:
- The particle position vector \(P_i\);
- The particle position change (velocity) vector \(V_i\);
- The personal (local) best position achieved by the particle so far \(p_{best}\). Moreover, let \(g_{best}\) denote the best particle in the swarm.

4.1 PSO Algorithm

Step 1: Initialize \(P_i\) and \(V_i\), and set \(p_{best} = P_i\) for \(i = 1, 2, ..., s\).
Step 2: Evaluate each particle \(p_{best}\) for \(i = 1, 2, ..., s\).
Step 3: Let \(g_{best}\) to be the best particle in \(\{p_{best_1}, p_{best_2}, ..., p_{best_s}\}\)
Step 4: For \(i = 1, 2, ..., s\) do:
   a. Update \(V_i\) according to:
      \[ V_i = \omega V_i + c_1 r_1 (p_{best,d} - Y_i) + c_2 r_2 (g_{best,d} - Y_i) \]  
   b. Update \(P_i\) according to:
      \[ P_i = P_i + V_i \]
Step 5: Go to Step 3, and repeat until convergence. Where \(w\) inertia weight factor; \(c_1, c_2\) self-confidence factor and swarm-confidence factor, respectively; \(r_1, r_2\) two random numbers uniformly distributed between 0 and 1. If \(P_i\) is better than \(p_{best}\), then \(p_{best} = P_i\)
Step 6: Go to Step 4, and repeat until convergence.

5.3D SPIHT WITH PSO ALGORITHM

In 3D SPIHT with PSO algorithm, we have initialized no. of particle is 5, and maximum iteration is 10. We have also initialized the value of speeding figure \(c_1, c_2\) with 2.1, random fiction initialize with random number, which values in between 0 and 1. Usually, PSO has the biggest speed of convergence when \(\omega\) is between 0.8 and 1.2 in 3D SPIHT with PSO algorithm we obtained inertia weight \(\omega\) is 0.82 by setting omxax with 0.9 and omin with 0.1. Our proposed 3D SPIHT with PSO algorithm described below:
Step 1: Initialize swarm size $s$, particle position vector $p_i$ and velocity vector $v_i$. Set particle’s personal best $p_{besti} = p_i$ for $i = 1, 2, \ldots s$.

Step 2: The image is transformed by 3D discrete wavelet transform (3D DWT) and full decomposition tree is gotten and fed to the 3D SPIHT algorithm with arithmetic coding.

Step 3: Calculate each particle’s fitness value

$$\text{MSE} = \frac{1}{N} \sum (f(x,y,z) - g(x,y,z))^2$$

$$\text{Fitness} = \frac{10 \log_{10} \left( \frac{1}{\text{MSE}} \right)}{\text{MSE}}$$

Where, $f(x,y,z)$ is input sequence of color images, $g(x,y,z)$ is reconstructed sequence of color images, and $L$ is maximum intensity value.

Step 4: Let global best $g_{best}$ to be the best particle in $\{ p_{best1}, p_{best2} \ldots p_{bests} \}$

Step 5: If particle’s position vector $p_i$ is better than particle’s personal best $p_{besti}$, then $p_{besti} = p_i$. If particle’s position vector $p_i$ is better than particle’s global best $g_{best}$, then $g_{best} = p_i$.

Step 6: for $i = 1, 2, \ldots s$, do:

Update velocity vector $v_i$ according to:

$$v_i = \omega * v_i + c_1 * r_1 (p_{besti} - p_i) + c_2 * r_2 (g_{best} - p_i)$$

Update position vector $p_i$ according to:

$$p_i = p_i + v_i$$

Step 7: The exploration process continue until a pre-specified iteration is specified, otherwise return to step 3.

### 6. IMAGE QUALITY METRICS

- **Compression Ratio (CR)**

$$\text{CR} = \frac{\text{Original Image Size}}{\text{Compressed Image Size}}$$

- **Mean Squared Error (MSE)**

MSE measures differences of reference image and distorted image pixels.

$$\text{MSE} = \frac{1}{M*N} \sum_{i=1}^{M} \sum_{j=1}^{N} (X_{ij} - Y_{ij})^2$$

Where $X_{ij}$ and $Y_{ij}$ are image gray values of reference image $X$ and distorted image $Y$, respectively. $M$ and $N$ are the width and height of the image.

- **Peak Signal-to-Noise Ratio (PSNR)**

The PSNR is mainly used to measure the quality of image. The PSNR between two images having 8 bits per pixels or samples in term of decibels (db) is given by:

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\text{MSE}}$$

- **Structural Similarity index (SSIM)**

The similarity index[7] compares the brightness, contrast and structure between each pair of vectors, where the structural similarity index (SSIM) between two signals $x$ and $y$ is given by the following expression:

$$\text{SSIM}(x, y) = 1 \cdot \text{c}(x, y) \cdot \text{r}(x, y)$$

However, the comparison of luminance is determined by the following expression:

$$\text{l}(x, y) = \frac{\sum y_{ij}^{1 \cdot \alpha} \cdot c_1}{\sum y_{ij}^{1} \cdot c_1 + \sum y_{ij}^{\alpha} \cdot c_2}$$

Where the average intensity of signal $x$ is given by: $L_x = \frac{\sum y_{ij} \cdot c_1}{\sum y_{ij}^{1} \cdot c_1 + \sum y_{ij}^{\alpha} \cdot c_2}$, the constant $K_1 << 1$, and $L$ is the dynamic row of the pixel values.

The function of contrast comparison takes the following form:

$$\text{c}(x, y) = \frac{\sum y_{ij}^{1} \cdot c_1}{\sum y_{ij}^{1} \cdot c_1 + \sum y_{ij}^{\alpha} \cdot c_2}$$
Where \( \sigma_x = \sqrt{\mu_x(x^2) - \mu_x^2} \) is the standard deviation of the original signal \( x \), \( C_2 = (K_2L) \), and the constant \( K_2 \ll 1 \).

The function of structure comparison is defined as follows:

\[
\delta(x, y) = \frac{g(x,y)}{\sigma_x \cdot \sigma_y \cdot \sigma_z} = \frac{\text{cov}(x,y) \cdot \sigma_z}{\sigma_x \cdot \sigma_y \cdot \sigma_z}
\]

(16)

Where \( \text{cov}(x,y) = \mu_{xy} - \mu_x \mu_y \), and \( \sigma_z = \frac{\sigma_z}{\sigma_z} \).

Then the expression of the structural similarity index becomes:

\[
\text{SSIM}(x, y) = \frac{(2\mu_x \mu_y + \gamma_1)(2\sigma_{xy} + \gamma_1)}{(\mu_x^2 + \mu_y^2 + \gamma_1)(\sigma_x^2 + \sigma_y^2 + \gamma_2)}
\]

(17)

For application, we require a single overall measurement of the whole image quality that is given by the following formula [12]:

\[
\text{MSIM}(I, \hat{I}) = \frac{1}{d} \sum_{d} \text{SSIM}_d(I_d, \hat{I}_d)
\]

(18)

Where \( I \) and \( \hat{I} \) are respectively the reference and degraded images, \( I_d \) and \( \hat{I}_d \) are the contents of images at the \( d \)th local window.

7. SIMULATION AND RESULTS

Simulation study is done in Matlab 7.8.0. We have used compression ratio (CR) parameter for compression and for image quality assessment we have used mean squared error (MSE), peak signal to noise ratio (PSNR), structural similarity index (SSIM) and Correlation coefficient.

For result analysis we have used continuous sequence of images from dataset, which is downloaded from http://vision.middlebury.edu/mview/

Figure 4. Shows sequence of input images for dataset.

![Input Images](image1)
![Input Images](image2)
![Input Images](image3)
![Input Images](image4)

Figure 4. Sequence of input images

Figure 5. Shows PSNR values of reconstructed images using 3D SPIHT algorithm at 0.1 bpp.

![Recon Images](image1)
![Recon Images](image2)
![Recon Images](image3)
![Recon Images](image4)


![Recon Images](image5)
![Recon Images](image6)
![Recon Images](image7)
![Recon Images](image8)

PSNR: 28.57  PSNR: 28.37  PSNR: 27.50  PSNR: 27.00

Figure 5. Reconstructed images using 3D SPIHT algorithm
Figure 6. Shows significant improvement in PSNR values of reconstructed images using 3D SPIHT with PSO algorithm at 0.1 bpp.

Figure 7. Shows graph of BPP v/s MSE for 3D SPIHT and 3D SPIHT with PSO algorithm at 0.1 to 0.5 bpp.

Figure 8. Shows graph of BPP v/s PSNR for 3D SPIHT and 3D SPIHT with PSO algorithm at 0.1 to 0.5 bpp.

Figure 9. Shows graph of BPP v/s Correlation Coefficient for 3D SPIHT and 3D SPIHT with PSO algorithm at 0.1 to 0.5 bpp.
Figure 10. Shows graph of BPP v/s MSSIM for 3D SPIHT and 3D SPIHT with PSO algorithm at 0.1 to 0.5 bpp.

8. CONCLUSION

3D SPIHT with PSO algorithm can be used for any image size. When the size of the color image increases, the time required for compression and reconstruction of the image also increases. The algorithm was tested using two color image dataset. The results show that we obtained improvement using 3D SPIHT with PSO algorithm in terms of compression ratio, mean-squared error, and Peak signal to noise ratio, correlation coefficient and multi-scale structural similarity index.

References