

## FRACTAL IMAGE COMPRESSION OF GRAYSCALE AND RGB IMAGES USING DCT WITH QUADTREE DECOMPOSITION AND HUFFMAN CODING

Moheb R. Girgis and Mohammed M. Talaat

**Abstract:** *Fractal image compression (FIC) is a well-known technique for image compression, but it suffers from slow encoding process. To improve the efficiency of FIC, hybrid encoding methods that combine fractal coding with other coding methods are used. This paper presents proposed hybrid FIC algorithms, for both grayscale and RGB images, which combine the Discrete Cosine Transform (DCT) with the Quadtree decomposition and Huffman coding techniques. The Quadtree decomposition method is used for the reduction of the search space and Huffman coding is used for improving the compression quality. The DCT is combined with the Quadtree decomposition and Huffman coding to improve the compression ratio. The paper also presents the results of the experiments that have been conducted to evaluate the effectiveness of the proposed hybrid FIC algorithms for grayscale and RGB images.*

**Keywords:** *Fractal image compression, Quadtree decomposition, Huffman coding, Discrete Cosine Transform, Hybrid fractal coding methods.*

**ITHEA Keywords:** *H. Information Systems: H.3 Information Storage and Retrieval, H.3.2 Information Storage.*

---

### Introduction

---

As the Internet is growing at an exponential rate, the amount of digital data being transferred, such as images, from one site to another has increased dramatically. There is now, more than ever, a need for quick data transfer methods and more efficient use of memory space. Since images require significantly large amount of memory, they are not transmittable quickly. Compression techniques are used to enable storing an image with much less memory than it would normally require, hence allowing it to be transmitted more quickly.

Fractal Image Compression (FIC) is a well-known technique for image compression. It exploits the self-similarity property efficiently by means of block self-affine transformations to generate a fractal code [Barnsley, 1988], [Jacquin, 1992], [Jacquin, 1993], [Fisher, 1994]. Most FIC techniques can compress a natural image at high compression rates. One of these techniques is based on a mathematical theory called iterated function systems (IFS) [Barnsley and Demko, 1985].

The main problem with fractal compression encoding is that it takes too much time. Therefore combining fractal coding and other coding methods becomes an important direction of fractal compression methods, in order to reduce the encoding time, while achieving better PSNR value and higher compression ratio. Several hybrid FIC methods have been proposed in the literature, such as, FIC with quadtree method for grayscale images [Fisher, 1994], [Ali and Mahmood, 2006]; FIC using quadtree decomposition and parametric line fitting method for synthetic images [Khan and Ohno, 2007]; FIC using quadtree decomposition with Huffman coding method for grayscale images [Veenadevi and Ananth, 2012], and for color images [Pandey and Seth, 2014]; hybrid image compression scheme for color images using DCT and FIC [Rawat and Meher, 2013]; FIC using quadtree decomposition and Discrete Wavelet Transform (DWT) method for color images [Chetan and Sharma, 2015]; and FIC using Discrete Cosine Transform (DCT) and quadtree decomposition with Huffman encoding method for grayscale images [Padmavati and Mesharam, 2015].

In this paper, we present six proposed hybrid FIC algorithms, for grayscale and RGB images, which combine the DCT with the quadtree decomposition and Huffman coding techniques. The quadtree decomposition method is used for the reduction of the search space and Huffman coding is used for improving the compression quality [Pandey and Seth, 2014]. The DCT is combined with the quadtree decomposition and Huffman coding to improve the compression ratio, [Padmavati and Mesharam, 2015].

The rest of the paper is organized as follows: Sections 2 to 5 give brief descriptions of the FIC technique, quadtree decomposition technique, Huffman coding technique, and DCT, respectively. Section 6 presents the hybrid FIC algorithms for grayscale images (Quadtree, Quadtree-Huffman, proposed DCT-Quadtree, and proposed DCT-Quadtree-Huffman). Section 7 presents the proposed hybrid FIC algorithms for RGB images (Quadtree, Quadtree-Huffman, DCT-Quadtree and DCT-Quadtree-Huffman). Section 8 presents the results of the experiments that we have conducted to evaluate the presented hybrid FIC algorithms. Section 9 presents the conclusion of the work presented in this paper, and some points for future work.

---

## Fractal Image Compression

---

In FIC, the image is firstly divided into a number of square blocks called ranges, and then the image is divided into bigger square blocks, called domains, such that the domain is four times larger than the range [Fisher, 1995]. Next, the domains are mapped to ranges using contractive affine transformations, then the domains are searched for the best match for every range, and the parameters representing the

corresponding fractal affine transformation will form the fractal compression code. For every range, the number of the appropriate domain and the fractal compression code are stored. Hence the compression is achieved by storing only the parameters instead of the range. The decoder performs a number of iterative operations in order to reconstruct the original image [Veenadevi and Ananth, 2012].

The advantages of FIC include: good mathematical encoding frame, resolution-free decoding, high compression ratio, and fast decompression. On the other hand, it suffers from slow encoding process [Ali and Mahmood, 2006]. So, the main aim of the research in this area is to achieve better PSNR value, higher compression ratio, and reduce long encoding time without hindering the image quality. To improve the efficiency of FIC, hybrid encoding methods that combine fractal coding with other coding methods are used [Veenadevi and Ananth, 2012].

---

### **Quadtree Decomposition Technique**

---

Quadtree decomposition is a representation of an image as a tree in which each node corresponding to a square portion of the image contains four sub-nodes corresponding to the four quadrants of the square, and the root of the tree is the initial image. The basic quadtree decomposition technique works as follows. After some initial number of quadtree partitions are made (corresponding to a specified minimum tree depth), the squares (ranges) at the nodes are compared with domains in the domain pool  $D$ , which are twice the range size. The pixels in the domain are averaged in groups of four so that the domain is reduced to the size of the range and the affine transformation of the pixel values is found that minimizes the root mean square (RMS) difference between the transformed domain pixel values and the range pixel values. All the potential domains are compared with a range. If the resulting RMS value is above a specified threshold and if the depth of the quadtree is less than a specified maximum depth, then the range is subdivided into four quadrants, which means adding four sub-nodes to the node corresponding to the range, and the process is repeated. If the RMS value is below the threshold, the optimal domain and the affine transformation on the pixel values are stored. The collection of all such transformations forms the encoding of the given image. [Fisher, 1995]

---

### **Huffman Coding Technique**

---

Huffman coding is one of the most popular techniques for removing coding redundancy, which has been used in various compression applications, including image and video compression. It utilizes the statistical property of alphabets in the source stream, and then produces respective binary codes for

these alphabets, called code words. These code words are strings of bits of variable length, i.e., their lengths are not fixed like ASCII. The code words for alphabets having higher probability of occurrence are shorter than those code words for alphabets having lower probability. Thus, Huffman coding is based on the frequency of occurrence of a data item (pixel or small blocks of pixels in images) [Aarti et al., 2013]. Code words are stored in a Code Book (or Dictionary), which is constructed for each image. The code book plus encoded data must be transmitted to enable decoding.

**Huffman Encoding:** The Huffman encoding algorithm starts by creating a list of all symbols, sorted by their frequencies (or probabilities of frequencies of occurrence) in descending order. Next, the two smallest frequency values are combined and deleted from the list, and then the new value is sorted into the list. This process is repeated until all the frequencies have been added up. The final number at the head of the tree should be the sum of all the frequencies. The code word of each symbol is traced from the top of the tree, noting the path taken at each fork. At each fork, the higher frequency edge is designated by a “1” and the lower frequency edge by a “0”, [Lee, 2007]. Thus, an extremely large string of symbols can be shown as a small list of binary encoded strings. It will be no any ambiguity in decoding, as no code word is a prefix of any other code word. In Huffman encoding of images, a symbol represents an image block.

**Huffman Decoding:** Given the encoded (compressed) data and the code book generated by the Huffman encoder, the Huffman decoder can decompress the data. Any Huffman encoded string can be decoded by examining the individual bits of the string in a left to right manner. The algorithm for decoding is simple. Start at the root of the Huffman tree and read the first bit off the input (compressed) file. If it is zero, follow the left edge of the tree; if it is one, follow the right edge. Read the next bit and move another edge toward the leaves of the tree. When the decoder arrives at a leaf, it finds there the original, uncompressed symbol, and that code is emitted by the decoder. The process starts again at the root with the next bit. [Veenadevi and Ananth, 2012]

---

## Discrete Cosine Transform

---

The Discrete Cosine Transform (DCT) is the key to the JPEG baseline compression process. The DCT is a mathematical function that takes a signal and transforms it from one type of representation to another. For example, an image is a two-dimensional signal that is perceived by the human visual system. The DCT converts the signal (spatial information) into numeric data ("frequency" or "spectral" information) so that the image's information exists in a quantitative form that can be manipulated for compression. [Lossy Data Compression JPEG, 2017].

DCT separates images into parts of different frequencies where less important frequencies are discarded through quantization and important frequencies are used to retrieve the image during decompression. DCT is applied to every non-overlapping block of the image. The DCT is performed on an  $N \times N$  square matrix of pixel values,  $f(x,y)$ , and it yields an  $N \times N$  square matrix of frequency coefficients,  $DCT(i,j)$ . The formula for the two-dimensional DCT is as follows:

$$DCT(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\left[\frac{(2x+1)i\pi}{2N}\right] \cos\left[\frac{(2y+1)j\pi}{2N}\right] \quad (1)$$

where  $i, j = 0, 1, 2, \dots, N-1$ , and  $C(k) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } k = 0 \\ 1 & \text{if } k > 0 \end{cases}$

In the implementation of this function, simple table lookups can replace several terms of the equation to simplify the DCT algorithm and improve its computation efficiency. For example, the two cosine terms only need to be calculated once and stored for later use. Likewise, the  $C(x)$  terms can also be replaced with table lookups. [Lossy Data Compression JPEG, 2017]

DCT compression algorithm [Cabeen and Gent, 2010]:

1. Divide the original image into blocks of size  $8 \times 8$ ;
2. Pixel values of a gray-scale image range from 0-255 but DCT is designed to work on pixel values ranging from -128 to 127. Therefore each block is modified to work in that range;
3. Calculate the DCT matrix using Equation (1);
4. Apply DCT to each block;
5. Compress each block through quantization;
6. Entropy encode the quantized matrix.

---

### The Proposed Grayscale Images Compression Algorithms

---

In this section, we firstly present the known FIC algorithms for grayscale images using quadtree decomposition, and quadtree decomposition with Huffman coding technique. Then, we present two proposed hybrid FIC algorithms for grayscale images, using DCT with quadtree decomposition, and DCT with quadtree decomposition and Huffman coding technique.

***FIC of Grayscale Images Using Quadtree Decomposition Algorithm:*** The steps of this algorithm are as follows:

**Algorithm 1 (Grayscale Quadtree)**

Begin

1. Read a grayscale image;
2. Resize the image to 128x128;
3. Apply quadtree encoding technique;
  - 3.1 Divide the resized image using quadtree decomposition with threshold value between 0 and 1, and minimum and maximum dimensions equal 2 and 64, respectively;
  - 3.2 Record x and y coordinates and size of the blocks resulted from the quadtree decomposition step;
  - 3.3 Find mean value of each block values;
4. Calculate the Compression Ratio (CR);
5. Decode the data using quadtree decoding;
6. Reconstruct the image;
7. Calculate PSNR between the original image and the reconstructed image.

End.

***FIC of Grayscale Images Using Quadtree decomposition with Huffman Coding Algorithm:*** The steps of this algorithm are as follows:

**Algorithm 2 (Grayscale Quadtree-Huffman)**

Begin

1. Read a grayscale image;
2. Resize the image to 128x128;
3. Apply Quadtree encoding technique (Step 3 of Algorithm 1) on the resized image;

4. Apply Huffman encoding technique:
  - 4.1 Calculate probabilities of frequencies of occurrence of the image blocks;
  - 4.2 Create Huffman dictionary corresponding to the blocks probabilities;
  - 4.3 Encode the data using Huffman encoding algorithm;
5. Calculate the CR;
6. Decode the data using Huffman decoding algorithm;
7. Reconstruct the image.
8. Calculate PSNR between the original image and the reconstructed image.

End.

***Proposed FIC of Grayscale Images using DCT and Quadtree decomposition Algorithm:*** The steps of this algorithm are as follows:

**Algorithm 3 (Grayscale DCT-Quadtree)**

Begin

1. Read a grayscale image;
2. Resize the image to 128x128;
3. Apply the DCT compression algorithm on the resized image;
  - 3.1 Partition the image into non-overlapping blocks, then apply the DCT to each block;
  - 3.2 Quantize the DCT coefficients of each block of the image;
4. Apply quadtree encoding technique (Step 3 of Algorithm 1) on the resultant image;
5. Calculate the CR;
6. Decode the data using Quadtree decoding;
7. Reconstruct the image;
8. Calculate PSNR between the original image and the reconstructed image.

End.

**Proposed FIC of Grayscale Images using DCT and Quadtree decomposition with Huffman Coding Algorithm:** The steps of this algorithm are as follows:

#### **Algorithm 4 (Grayscale DCT-Quadtree-Huffman)**

Begin

1. Read a grayscale image;
2. Resize the image to 128x128;
3. Apply the DCT compression algorithm (Step 3 of Algorithm 3) on the resized image;
4. Apply Quadtree encoding technique (Step 3 of Algorithm 1) on the resultant image;
5. Apply Huffman encoding and decoding technique (Step 4-8 of Algorithm 2).

End.

---

#### **The Proposed RGB Images Compression Algorithms**

---

Although the algorithms presented above are for grayscale images, RGB images are compressed following the same algorithms. Since the RGB image is composed of three components, red, green, and blue, it is treated as though it were three separate images, each of a different color component, and each image would be compressed separately.

In this section, we present four proposed hybrid FIC compression methods for RGB images, using (1) quadtree decomposition, (2) quadtree decomposition with Huffman coding technique, (3) DCT with quadtree decomposition, and (4) DCT with quadtree decomposition and Huffman coding technique.

**Proposed FIC of RGB Images Using Quadtree Decomposition Algorithm:** The steps of this algorithm are as follows:

#### **Algorithm 5 (RGB Quadtree)**

Begin

1. Read a RGB image;
2. Resize the image to 128x128;
3. Set the threshold values (between 0 and 1) for red, green and blue components of the image;



4. Apply quadtree encoding technique:
  - 4.1 Divide resized image using quadtree decomposition;
  - 4.2 Record x and y coordinates and size of the R, G and B blocks resulted from the quadtree decomposition step;
  - 4.3 Find mean value of the values in each R, G and B block;
5. Calculate the CR;
6. Decode the data using quadtree decoding;
7. Reconstruct the RGB image:
  - 7.1 Reconstruct the R component of the image;
  - 7.2 Reconstruct the B component of the image;
  - 7.3 Reconstruct the G component of the image;
  - 7.4 Reconstruct the image from its R, G, and B components;
8. Calculate the PSNR between the original image and the reconstructed image.

End.

***Proposed FIC of RGB Images Using Quadtree decomposition with Huffman Coding Algorithm:***

The steps of this algorithm are as follows:

**Algorithm 7 (RGB Quadtree-Huffman)**

Begin

1. Read a RGB image;
2. Resize the image to 128x128;
3. Apply quadtree encoding technique (Steps 3 and 4 of Algorithm 5) on the resized RGB image;
4. Apply Huffman encoding technique (Step 4 of Algorithm 2);
5. Calculate the CR;
6. Decode the data using Huffman decoding algorithm;
7. Reconstruct the RGB image (Step 7 of Algorithm 5);
8. Calculate the PSNR between the original image and the reconstructed image.

End.

**Proposed FIC of RGB Images using DCT and Quadtree decomposition Algorithm:** The steps of this algorithm are as follows:

**Algorithm 6 (RGB DCT-Quadtree)**

Begin

1. Read a RGB image;
2. Resize the image to 128x128;
3. Apply the DCT compression algorithm on the resized RGB image:
  - 3.1 Partition the image into non-overlapping R, G and B blocks, then apply the DCT to each block;
  - 3.2 Quantize the DCT coefficients of each block of the image;
4. Apply quadtree encoding technique (Steps 3 and 4 of Algorithm 5) on the resultant RGB image;
5. Apply Steps 5-8 of Algorithm 5 to reconstruct the RGB image.

End.

**Proposed FIC of RGB Images using DCT with Quadtree decomposition and Huffman Coding Algorithm:** The steps of this algorithm are as follows:

**Algorithm 8 (RGB DCT-Quadtree-Huffman)**

Begin

1. Read a RGB image;
2. Resize the image to 128x128;
3. Apply the DCT compression algorithm on the resized RGB image (Step 3 of Algorithm 7);
4. Apply quadtree encoding technique (Steps 3 and 4 of Algorithm 5) on the resultant RGB image;
5. Apply Huffman encoding and decoding technique (Step 4-8 of Algorithm 6).

End.

---

## Experimental Results

---

This section presents the results of two sets of experiments that we have conducted to evaluate the effectiveness of the presented hybrid compression algorithms, for grayscale and RGB images, which combine the DCT with the quadtree decomposition method and Huffman coding technique.

The algorithms are implemented using MATLAB 7, on MacBook Pro, Intel Core i5, 2.3GHz. The performance of the presented image compression techniques is evaluated by measuring the quadtree decomposition time, the decompression time, the PSNR, and the compression ratio.

**Peak Signal-to-Noise Ratio (PSNR)** is defined as the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation [Wikipedia, 2016]. PSNR is most commonly used to measure the reconstruction quality of lossy image compression. The signal in this case is the original data, and the noise is the error introduced by compression. When comparing compression codecs, PSNR is an approximation to human perception of reconstruction quality. Higher the value of PSNR, better the quality of the reconstructed image. The PSNR (in dB) is defined, via the mean squared error (MSE), as:

$$PSNR = 10 \times \log_{10} \left( \frac{MAX_I^2}{MSE} \right) \quad (2)$$

where  $MAX_I$  is the maximum possible pixel value of the image, and  $MSE$ , given a noise-free  $m \times n$  image  $I$  and its noisy approximation  $K$ , is defined as:

$$MSE = \frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \quad (3)$$

**Image Compression Ratio (CR)** determines the efficiency of the compression algorithm. It is defined as the ratio between the size of the original image and the size of compressed image:

$$CR = \text{Size of original image} / \text{Size of compressed image}$$

### **Grayscale Images Experiments**

In the first set of experiments, we have applied the presented four hybrid FIC algorithms for grayscale images (Quadtree, Quadtree-Huffman, proposed DCT-Quadtree, and proposed DCT-Quadtree-Huffman) to Lena and Camera Man images. The images are resized to 128x128. Figure 1 shows the original images and the decompressed images after applying each of the four algorithms, with two different threshold values: 0.5 and 0.1. The quadtree decomposition and decompression times (in seconds), the PSNR values, and the compression ratios obtained for the two grayscale images, by using the four algorithms, with the two threshold values, are shown in Table 1, and graphically represented in Figures 2 to 5, respectively. It can be seen from the table and the figures that:

- The effect of using different threshold values: for both images, when using different threshold values with each method, the compression ratios are not affected; when using Quadtree and Quadtree-Huffman with the smaller threshold value (0.1), the PSNR values are improved; but when using the proposed algorithms: DCT-Quadtree and DCT-Quadtree-Huffman, the PSNR values are same for both threshold values. The Quadtree decomposition time and the decompression time were better when using the four algorithms with the larger threshold value (0.5).
- The quadtree decomposition time: for both images, with threshold value 0.5, all algorithms have taken same quadtree decomposition time; but with threshold value 0.1, for Lana image, Quadtree, Quadtree-Huffman, and proposed DCT-Quadtree algorithms have taken less decomposition time than proposed DCT-Quadtree-Huffman algorithm, and for Camera Man image, Quadtree-Huffman and proposed DCT-Quadtree-Huffman algorithms have taken less decomposition time than the other two algorithms.
- The decompression time: for both images, with both threshold values, the proposed DCT-Quadtree, the proposed DCT-Quadtree-Huffman, and Quadtree-Huffman algorithms, respectively, have taken less decompression time than the Quadtree. These results indicate that combining Quadtree with any or all of the other 2 methods reduced the decompression time.
- The PSNR values: for both images, with the two threshold values, the proposed algorithms, DCT-Quadtree and DCT-Quadtree-Huffman, have higher PSNR values than the Quadtree-Huffman and Quadtree algorithms. These results indicate that the combination of DCT with Quadtree and with Quadtree-Huffman has improved the quality of the reconstructed images.
- The compression ratio: for both images, with the two threshold values, the proposed algorithms, DCT-Quadtree and DCT-Quadtree-Huffman, have higher CR than the Quadtree and Quadtree-Huffman algorithms. These results indicate that the combination of DCT with Quadtree and with Quadtree-Huffman has enhanced the CR.

















Original Image	Lena		Camera Man	
	0.5	0.1	0.5	0.1
Threshold				
Quadtree				
DCT-Quadtree				
Quadtree-Huffman				
DCT-Quadtree-Huffman				

Figure 1. The results of applying the four hybrid FIC algorithms to two grayscale images

Table 1. The results of applying the four hybrid FIC algorithms for grayscale images with two thresholds

		Lena		Camera Man	
		0.5	0.1	0.5	0.1
<b>Threshold</b>					
<b>Quadtree Decomposition Time (Seconds)</b>	<b>Quadtree</b>	0.007	0.009	0.008	0.08
	<b>DCT-Quadtree</b>	0.007	0.009	0.008	0.08
	<b>Quadtree-Huffman</b>	0.007	0.009	0.008	0.008
	<b>DCT-Quadtree-Huffman</b>	0.007	0.015	0.008	0.016
<b>Decompression Time (Seconds)</b>	<b>Quadtree</b>	0.016	0.085	0.023	0.098
	<b>DCT-Quadtree</b>	0.005	0.076	0.006	0.082
	<b>Quadtree-Huffman</b>	0.013	0.063	0.018	0.078
	<b>DCT-Quadtree-Huffman</b>	0.01	0.047	0.015	0.048
<b>PSNR</b>	<b>Quadtree</b>	21.33	22.35	19.60	22.16
	<b>DCT-Quadtree</b>	22.63	22.63	22.3	22.3
	<b>Quadtree-Huffman</b>	20.333	22.55	19.65	22.26
	<b>DCT-Quadtree-Huffman</b>	22.63	22.63	22.3	22.3
<b>CR</b>	<b>Quadtree</b>	16.0	16.0	4.0	4.0
	<b>DCT-Quadtree</b>	17.0	17.0	5.0	5.0
	<b>Quadtree-Huffman</b>	16.0	16.0	4.0	4.0
	<b>DCT-Quadtree-Huffman</b>	17.0	17.0	5.0	5.0

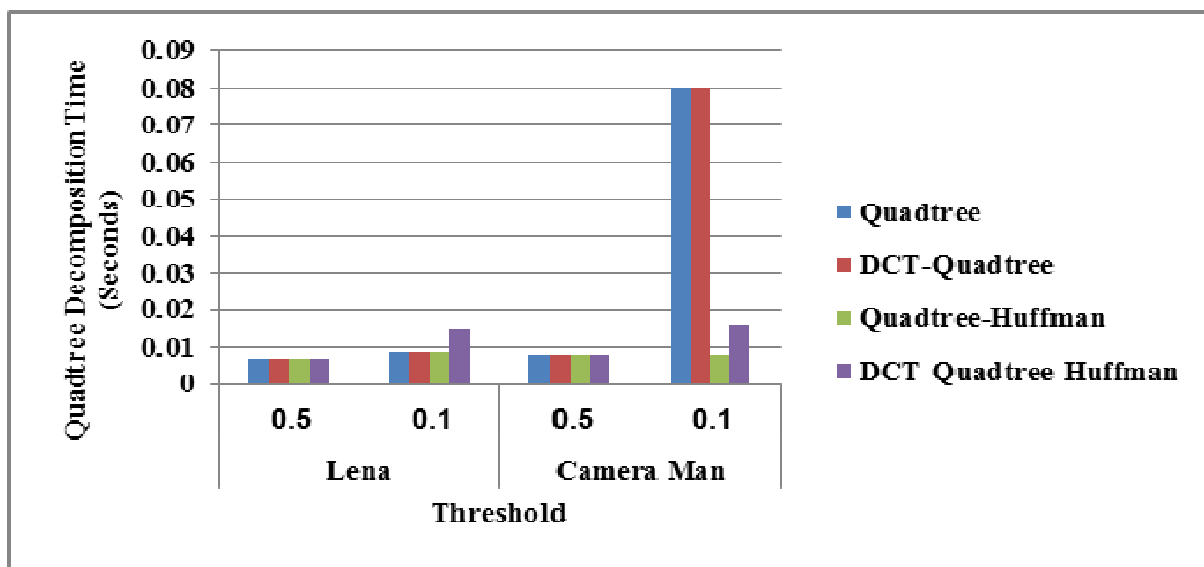


Figure 2. A comparison between the quadtree decomposition time for the four hybrid FIC algorithms for grayscale images with two different thresholds

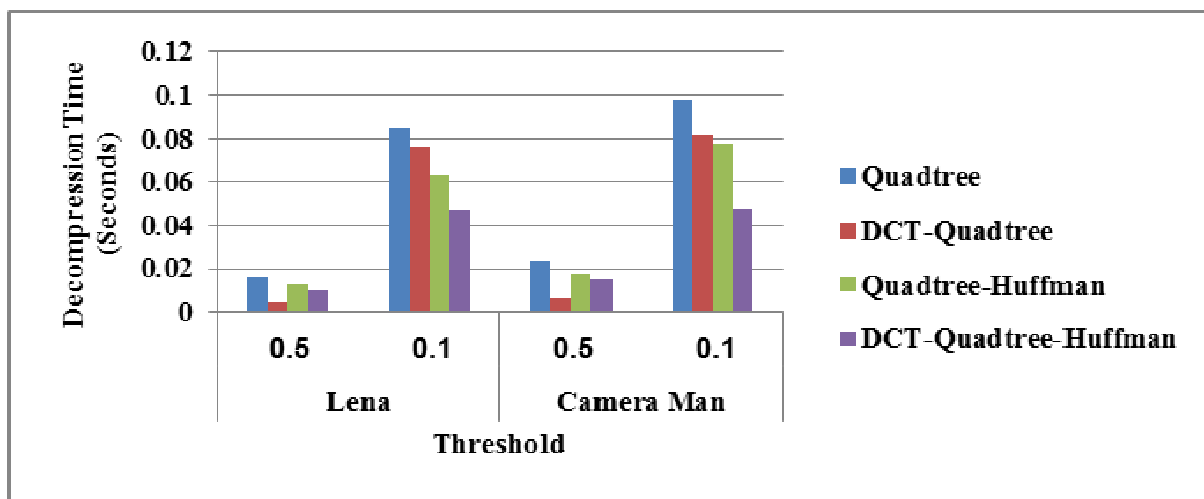


Figure 3. A comparison between the decompression time for the four hybrid FIC algorithms for grayscale images with two different thresholds

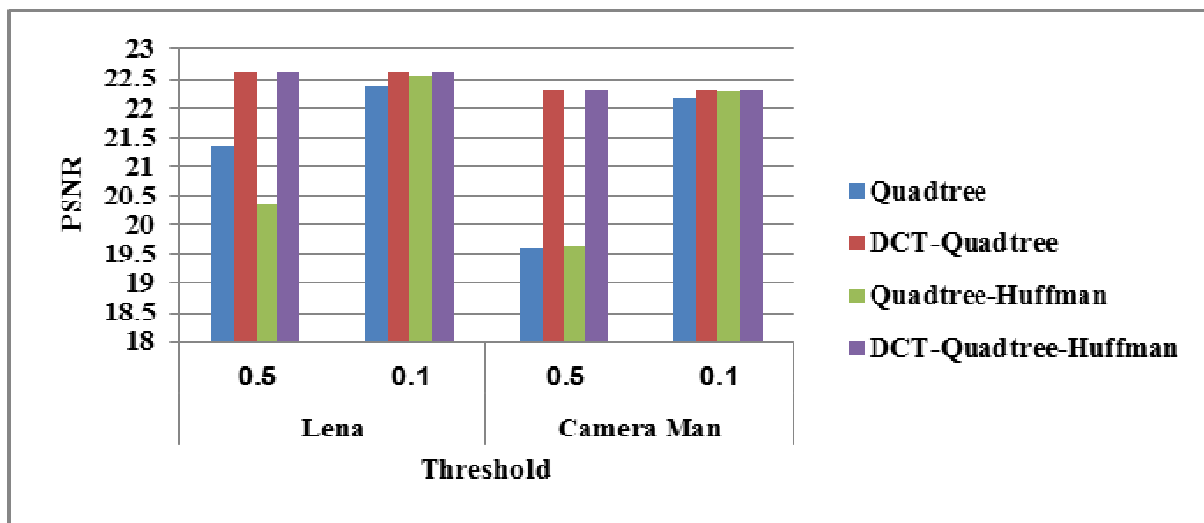


Figure 4. A comparison between the PSNR values for the four hybrid FIC algorithms for grayscale images with two different thresholds

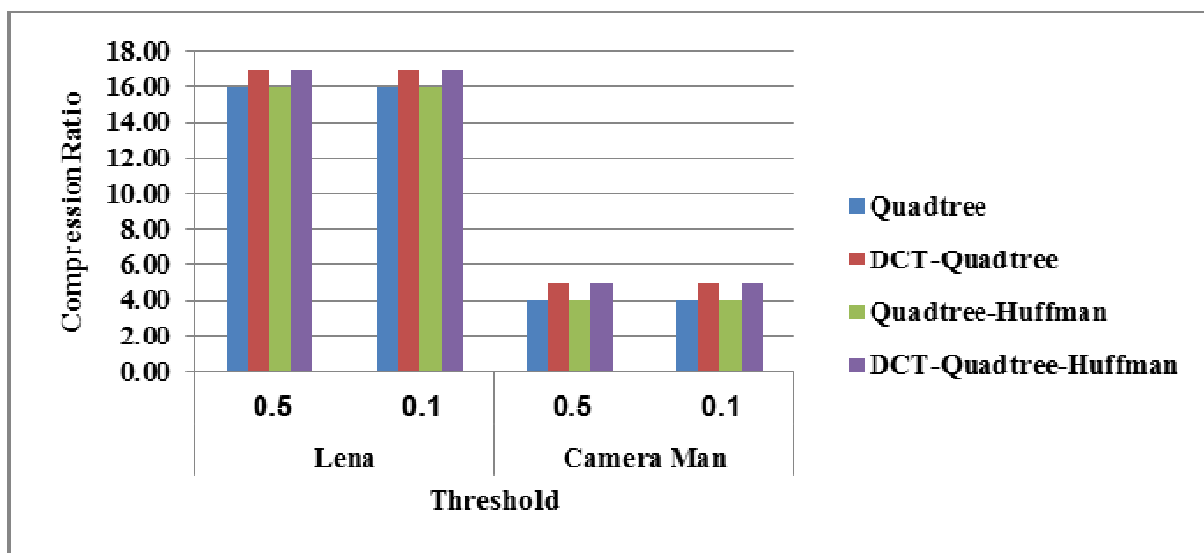


Figure 5. A comparison between the compression ratio for the four hybrid FIC algorithms for grayscale images with two different thresholds



### **RGB Images Experiments**

In the second set of experiments, we have applied the four proposed hybrid FIC algorithms for RGB images: (Quadtree, DCT-Quadtree, Quadtree-Huffman, and hybrid DCT-Quadtree-Huffman) to Lena and Ship images. The images are resized to 128x128. Figure 6 shows the original images and the decompressed images after applying each of the four algorithms, with two different sets of threshold values: [0.5, 0.5, 0.5] and [0.1, 0.1, 0.1] for red, green and blue components of the RGB image.

The quadtree decomposition and decompression times (in seconds), the PSNR values, and the compression ratios obtained for the two RGB images, by using the four algorithms, with the two sets of threshold values, are shown in Table 2, and graphically represented in Figures 7 to 10, respectively. It can be seen from the table and the figures that:

- The effect of using different threshold values: for both images, when using different threshold values with each method, the compression ratios are not affected; when using the smaller set of threshold values, [0.1, 0.1, 0.1], with the four algorithms, the PSNR values have improved; but the quadtree decomposition time and the decompression time were better when using the larger set of threshold values [0.5, 0.5, 0.5] with the four algorithms.
- The quadtree decomposition time: for the two images, with the two sets of threshold values, on average, the DCT-Quadtree and DCT-Quadtree-Huffman have taken less decomposition time than the other two algorithms. These results indicate that combining Quadtree and Quadtree-Huffman with DCT reduced the decompression time.
- The decompression time: for both images, with the two sets of threshold values, the DCT-Quadtree-Huffman, Quadtree-Huffman, and DCT-Quadtree have taken less decompression time than the Quadtree algorithm. It can be seen also that DCT-Quadtree has taken less decompression time than Quadtree, and DCT-Quadtree-Huffman has taken less time than Quadtree-Huffman, which means DCT caused reduction in the decompression time.
- The PSNR values: for both images, with the two sets of threshold values, the DCT-Quadtree and DCT-Quadtree-Huffman have higher PSNR values than the Quadtree and hybrid Quadtree-Huffman algorithms. These results indicate that the combination of DCT with Quadtree and with Quadtree-Huffman has improved the quality of the reconstructed images.
- The compression ratio: for both images, with the two sets of threshold values, the DCT-Quadtree and DCT-Quadtree-Huffman have higher CR than the Quadtree and hybrid Quadtree-Huffman algorithms. These results indicate that the combination of DCT with Quadtree and with Quadtree-Huffman has enhanced the CR.












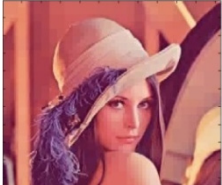


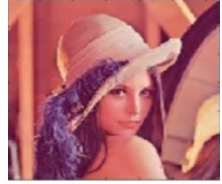
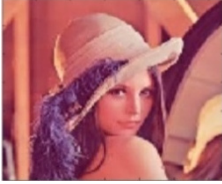


	Lena		Ship	
<b>Original Image</b>				
<b>Threshold</b>	<b>[0.5,0.5,0.5]</b>	<b>[0.1,0.1,0.1]</b>	<b>[0.5,0.5,0.5]</b>	<b>[0.1,0.1,0.1]</b>
<b>Quadtree</b>				
<b>DCT-Quadtree</b>				
<b>Quadtree-Huffman</b>				
<b>DCT-Quadtree-Huffman</b>				

Figure 6. The results of applying the four hybrid FIC algorithms to two RGB images

Table 2. The results of applying the four FIC algorithms to two RGB images, with two sets of thresholds

		Lena		Ship	
Threshold		0.5,0.5,0.5	0.1,0.1,0.1	0.5,0.5,0.5	0.1,0.1,0.1
Quadtree Decomposition Time (Seconds)	Quadtree	2.136	11.635	3.192	12.883
	DCT-Quadtree	2.102	2.131	2.134	2.171
	Quadtree-Huffman	2.136	11.371	3.089	12.592
	DCT-Quadtree-Huffman	2.172	2.295	2.109	2.388
Decompression Time (Seconds)	Quadtree	0.020	0.066	0.044	0.085
	DCT-Quadtree	0.014	0.061	0.034	0.063
	Quadtree-Huffman	0.016	0.059	0.026	0.065
	DCT-Quadtree-Huffman	0.013	0.052	0.023	0.065
PSNR	Quadtree	51.42	82.73	55.25	83.73
	DCT-Quadtree	87.52	88.35	82.63	83.64
	Quadtree-Huffman	51.42	82.73	55.25	83.73
	DCT-Quadtree-Huffman	87.52	88.35	82.63	83.64
CR	Quadtree	2.61	2.61	1.53	1.53
	DCT-Quadtree	3.09	3.09	3.072	3.072
	Quadtree-Huffman	2.94	2.94	1.88	1.88
	DCT-Quadtree-Huffman	3.09	3.09	3.072	3.072

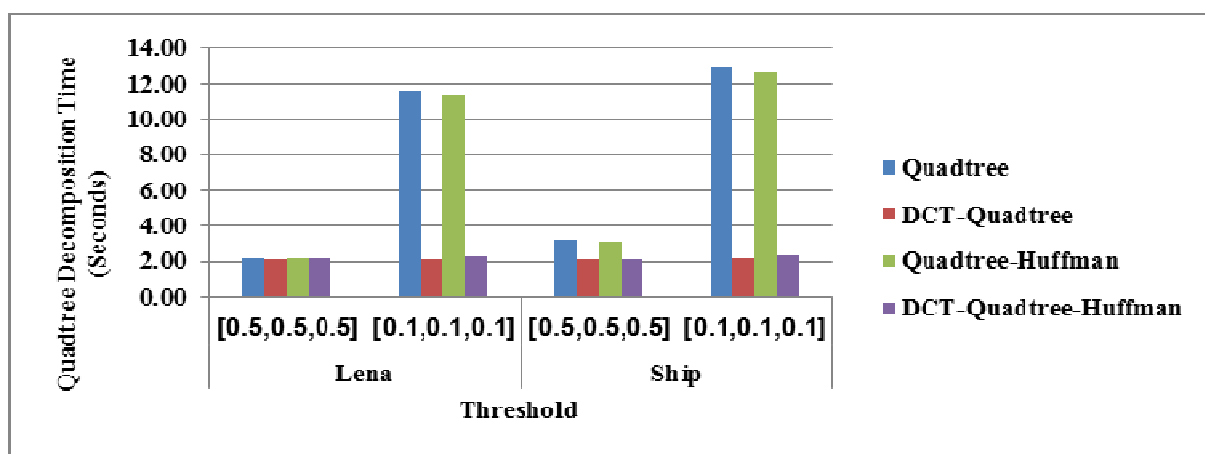


Figure 7. A comparison between the quadtree decomposition time for the four hybrid FIC algorithms for RGB images with two different sets of thresholds

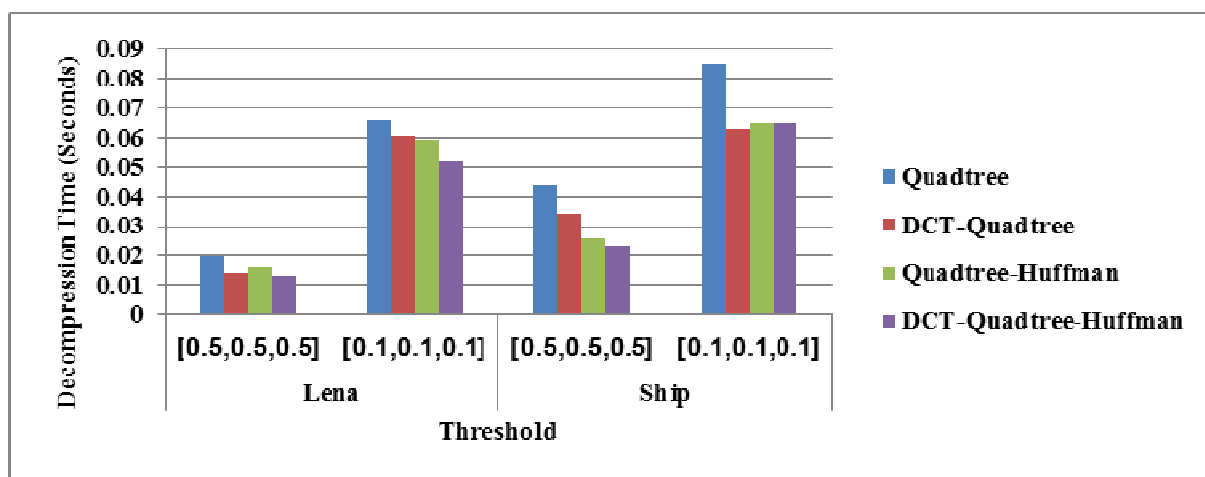


Figure 8. A comparison between the decompression time for the four hybrid FIC algorithms for RGB images with two different sets of thresholds

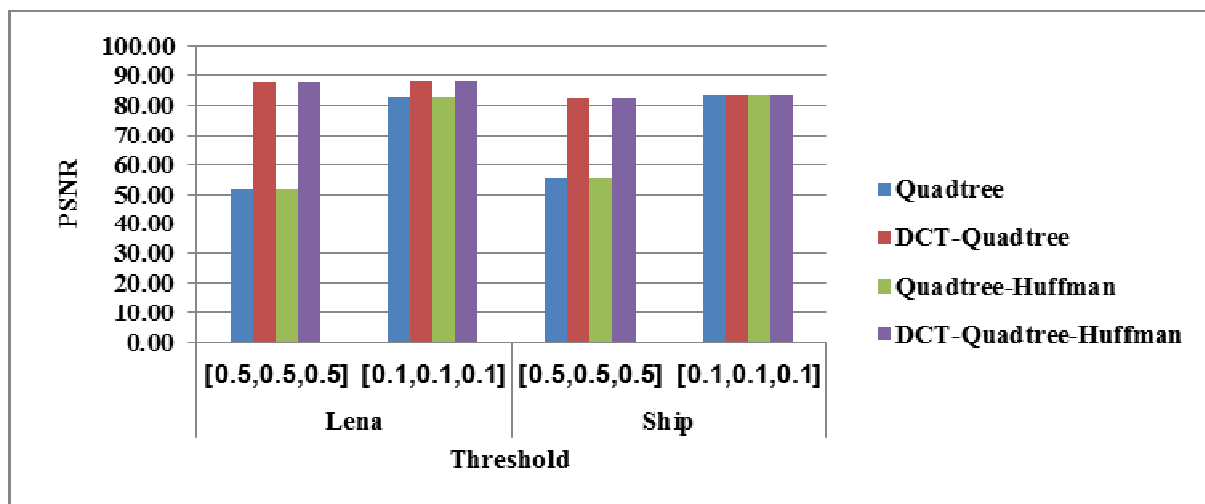


Figure 9. A comparison between the PSNR values for the four hybrid FIC algorithms for RGB images with two different sets of thresholds

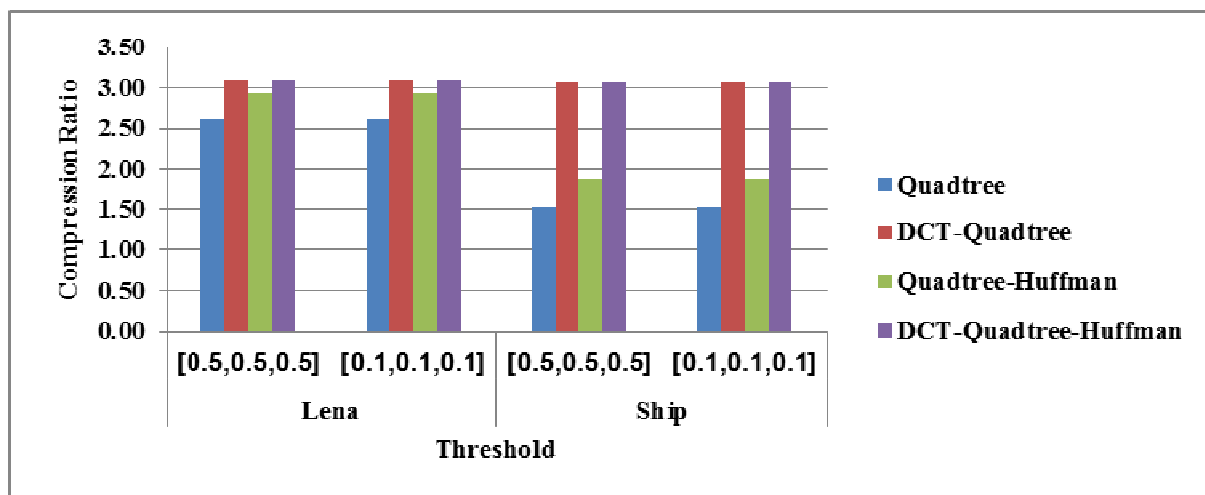


Figure 10. A comparison between the compression ratio for the four hybrid FIC algorithms for RGB images with two different sets of thresholds

## Conclusion

This paper presented a set of hybrid FIC methods, for grayscale and RGB images, that combine the DCT and the Quadtree decomposition method with Huffman coding technique. The Quadtree decomposition method is used for the reduction of the search space and Huffman coding is used for improving the compression quality. The DCT is combined with the Quadtree decomposition and Huffman coding to improve the compression ratio. The presented hybrid FIC algorithms are: Quadtree, DCT-Quadtree, Quadtree-Huffman, and DCT-Quadtree-Huffman, for grayscale and RGB images.

Experiments have been conducted to evaluate the effectiveness of the presented hybrid compression algorithms. The results of these experiments indicated that the combination of DCT with Quadtree and with Quadtree-Huffman has improved the quality of the reconstructed images and enhanced the compression ratio, for both grayscale and RGB images. Also, the results indicated that, for grayscale images, combining Quadtree with any or all the other 2 methods reduced the decompression time; and for RGB images, combining Quadtree and Quadtree-Huffman with DCT reduced the decomposition and decompression times.

The presented FIC algorithms are applied to still images. As a future work, we intend to apply them to images in video sequences. In addition, we intend to apply the presented algorithms to images of color spaces other than RGB.

## Bibliography

---

- [Aarti et al., 2013] Aarti et al., Performance Analysis of Huffman Coding Algorithm. International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, Issue 5, May 2013. pp. 615-619.
- [Ali and Mahmood, 2006] Ali, F. H., Mahmood, A. E., Quad-tree Fractal Image Compression. Al-Rafidain Engineering, Vol. 14, Issue 4, 2006. pp. 82-98.
- [Barnsley, 1988] M. Barnsley. Fractals everywhere. Academic Press, New York, 1988.
- [Barnsley and Demko, 1985] Barnsley, M. F., Demko, S., Iterated function systems and the global construction of fractals. In: Proceedings of Royal Society A, Volume 399, Issue 1817, 8 June 1985, pp. 243-275.
- [Cabeen and Gent, 2010] Cabeen, K., Gent, P., Image Compression and the Discrete Cosine Transform. Mathematica Journal, Vol. 4, Issue 1, 2010. pp. 81-88.
- [Chetan and Sharma, 2015] Chetan, Sharma D., Fractal image compression using quadtree decomposition and DWT. International Journal of Scientific Engineering and Research, Vol. 3, Issue 7, July 2015. pp. 112-116.
- [Fisher, 1995] Y. Fisher. Fractal image compression: theory and application. Springer-Verlag, New York, 1995.
- [Jacquin, 1992] Jacquin, A. E., Image coding based on a fractal theory of iterated contractive image transformation. IEEE Transactions on Image Processing, Vol. 1, Issue 1, Jan 1992, pp. 18-30.
- [Jacquin, 1993] Jacquin, A. E., Fractal image coding: a review. Proceedings of IEEE, Vol. 81, Issue 10, pp. 1451–1465, Oct 1993.
- [Khan and Ohno, 2007] Khan, M. and Ohno, Y., A Hybrid Image Compression Technique using Quadtree Decomposition and Parametric Line Fitting for Synthetic Images. Advances in Computer Science and Engineering Conference, Vol. 1, Issue 3. Publishing house, Allahabad, India, Nov. 2007. pp. 263-283
- [Lee, 2007] Lee, J., Huffman Data Compression. MIT Undergraduate Journal of Mathematics, May 23, 2007.
- [Lossy Data Compression JPEG, 2017] Lossy Data Compression JPEG, <https://cs.stanford.edu/people/eroberts/courses/soco/projects/data-compression/lossy/jpeg/index.htm>. Last accessed November 2017.

- [Padmavati and Mesharam, 2015] Padmavati, S., Mesharam V., DCT Combined With Fractal Quadtree Decomposition and Huffman Coding for Image Compression. In: CATCON 2015, Proceedings of the International Conference on Condition Assessment Techniques in Electrical Systems. Bangalore, India, 10-12 Dec. 2015. pp. 28- 33.
- [Pandey and Seth, 2014] Pandey, S., Seth, M., Hybrid Fractal Image Compression Using Quadtree Decomposition with Huffman Coding. International Journal of Science and Research (IJSR), Vol. 3, Issue 6, June 2014. pp. 943-948.
- [Rawat and Meher, 2013] Rawat, C. S., Meher, S., A Hybrid Image Compression Scheme using DCT and Fractal Image Compression. The International Arab Journal of Information Technology, Vol. 10, Issue 6, November 2013. pp. 553-562.
- [Veenadevi and Ananth, 2012] Veenadevi, S. V., Ananth, A. G., Fractal Image Compression Using Quadtree Decomposition and Huffman Coding. Signal & Image Processing: An International Journal (SIPIJ), Vol. 3, Issue 2, April 2012.
- [Wikipedia, 2016] [https://en.wikipedia.org/wiki/Peak\\_signal-to-noise\\_ratio](https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio). Last accessed May 2016.

---

#### Authors' Information

---



**Moheb R. Girgis** – Department of Computer Science, Faculty of Science, Minia University; Professor of Computer Science; El-Minia, Egypt; e-mail: [moheb.girgis@mu.edu.eg](mailto:moheb.girgis@mu.edu.eg)

*Major Fields of Scientific Research: Software Testing, Evolutionary Algorithms, Image Processing, Computer Networks.*



**Mohammed M. Talaat** – Department of Computer Science, Faculty of Science, Minia University; M.Sc. Student; El-Minia, Egypt; e-mail: [talaat\\_eng@yahoo.com](mailto:talaat_eng@yahoo.com)

*Major Fields of Scientific Research: Image Processing, Software Development.*