# Analysis of Image Cryptosystem with public key using Discrete Cosine Transformation with RGB Content

Anjali Sheoran[1], Taruna Sikka[2]
[1]M.Tech Scholar, SPGOI Rohtak
[2]Assistant Professor, ECE Deptt. SPGOI Rohtak

***Abstract*** - The discrete cosine transform (DCT) is a technique for converting a signal into elementary frequency components. It is widely used in image compression. Here we develop some simple functions to compute the DCT and to compress images. These functions illustrate the power of *Mathematical* in the prototyping of image processing Algorithms.

In this cryptosystem, we have considered RGB images for two-dimensional (2D) data security. Security of RGB images during transmission is a major concern, discussed globally. This paper proposes a novel technique for color image security by random cipher associated with 2D discrete cosine transform. Existing techniques have discussed the security of image data on the basis of the keys only (which provide only one layer of security for image data), but in the proposed cryptosystem, the keys and the arrangement of cipher parameters are imperative for correct decryption of color image data. Additionally, key multiplication side (pre or post) with the RGB image.

***Keywords*** - *DCT, RGB Image, cryptosystem.*

## I. INTRODUCTION

Many methods of lossy compression have been developed; however, a family of techniques called transform compression has proven the most valuable [1][2]. The best example of transform compression is embodied in the popular JPEG standard of image encoding. JPEG is named after its origin, the Joint Photographers Experts Group. Discrete Fourier Transform (DFT) and Discrete Cosine Transform (DCT) are commonly used algorithms to represent an arbitrary signal in terms of orhonormal bases [1]. In DFT case, these basis functions are cosine and sinusoids (in complex form) where DCT depends on cosine signals to represent the signal. Even though DCT is more used than DFT, the approach that they use to represent signal are similar but they only differ by basic functions [3].

Architectures are available for DCT and DFT. As compared to DFT, application of DCT results in less blocking artifacts due to the even symmetric extension properties of DCT. Also, DCT uses real computations, unlike the complex computations used in DFT. This makes DCT hardware simpler, as compared to that of DFT. These advantages have made DCT-based image compression a standard in still-image and multimedia coding standards. In this work, we are going to discuss DCT-based image compression in detail. We shall first present the basic theory of DCT and show how to transform a block of image into its corresponding transformed array of DCT coefficients. The transformed array is encoded into a bit-stream using two approaches: zonal coding and threshold coding. The two approaches will be presented in this paper. We shall show how the threshold-coded coefficients are scanned in a zigzag order and Huffman coded [4]. At the end of this lesson, we point out some of the performance limitations of DCT in low bit-rate situations.

Experimentally and also theoretically (with a better boundary condensations), DCT transform yields smaller coefficients for natural images and audios. Therefore, for compression purposes, DCT is de facto method for a number of applications including JPEG and lossy compression for audios [3]. DCT is an orthogonal transformation that is very widely used in image compression and is widely accepted in the multimedia standards. DCT belongs to a family of 16 trigonometric transformations. The type-2 DCT transforms a block of image of size $N$ x $N$ having pixel intensities $s$ ($n1$, $n2$) into a transform array of coefficients $S(k1,k2)$, described by the following equation:

$$S(k_1,k_2)=\sqrt{\frac{4}{N^2}}C(k_1)C(k_2)\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1}s(n_1,n_2)\cos\left(\frac{\pi(2n_1+1)k_1}{2N}\right)\cos\left(\frac{\pi(2n_2+1)k_2}{2N}\right)$$

where $k_1, k_2, n_1, n_2 = 0,1,\cdots\cdots N-1$, and

$$C(k)=\begin{cases}1/\sqrt{2} & \text{for } k=0\\ 1 & \text{otherwise}\end{cases}$$

The transformed array $S$ $(k1, K2)$ obtained through equation is also of the size $N$ x $N$, same as that of the original image block [5]. It should be noted here that the transform-domain indices $k1$ and $K2$ indicate the spatial frequencies in the directions of $n1$ and $n2$ respectively. $k1 = k2 = 0$corresponds to the average or the DC component and all the remaining ones are the AC components which correspond to higher spatial frequencies as $k1$ and $k2$ increase [6].

## II. EXPERIMENTAL APPROACH

The compression ratio of lossless methods (e.g., Huffman, Arithmetic, and LZW) is not high enough for image and video compression. JPEG uses transform coding; it is largely based on the following observations:

**Observation 1:** A large majority of useful image contents change relatively slowly across images, i.e., it is unusual for intensity values to alter up and down several times in a small area, for example, within an 8 x 8 image block. A translation of this fact into the spatial frequency domain implies, generally, lower spatial frequency components contain more information than the high frequency components which often correspond to less useful details and noises.

**Observation 2:** Experiments suggest that humans are more immune to loss of higher spatial frequency components than loss of lower frequency components.

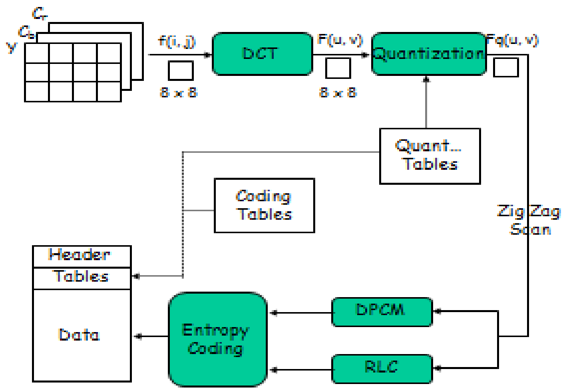



Fig.1: JPEG Coding

*A. Steps Involved*

1. Discrete Cosine Transform of each 8x8 pixel array f(x,y) $\rightarrow_T$ F(u,v)
2. Quantization using a table or using a constant
3. Zig-Zag scan to exploit redundancy
4. Differential Pulse Code Modulation(DPCM) on the DC component and Run length Coding of the AC components
5. Entropy coding (Huffman) of the final output

As shown in Figure 2, JPEG compression starts by breaking the image into 8×8 pixel groups. The full JPEG algorithm can accept a wide range of bits per pixel, including the use of color information. In this example, each pixel is a single byte, a grayscale value between 0 and 255[7]. These 8×8 pixel groups are treated independently during compression. That is, each group is initially represented by 64 bytes [1]. After transforming and removing data, each group is represented by, say, 2 to 20 bytes. During uncompression, the inverse transform is taken of the 2 to 20 bytes to create an approximation of the original 8×8 group.

In any event, the 8×8 size works well, and it may or may not be changed in the future. Many different transforms have been investigated for data compression, some of them invented specifically for this purpose. For instance, the Karhunen-Loeve transform provides the best possible compression ratio, but is difficult to implement. The Fourier transform is easy to use, but does not provide adequate compression. After much competition, the winner is a relative of the Fourier transform, the Discrete Cosine Transform (DCT).

Just as the Fourier transform uses sine and cosine waves to represent a signal, the DCT only uses cosine waves. There are several versions of the DCT, with slight differences in their mathematics. When the DCT is taken of an 8×8 group, it results in an 8×8 spectrum. In other words, 64 numbers are changed into 64 other numbers. All these values are real; there is no complex mathematics here. Just as in Fourier analysis, each value in the spectrum is the amplitude of a basis function. Figure 27-10 shows 6 of the 64 basis functions used in an 8×8 DCT, according to where the amplitude sits in the spectrum. The 8×8 DCT basis functions are given by:

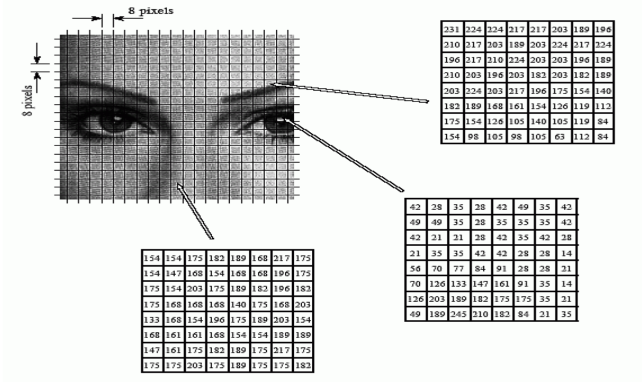
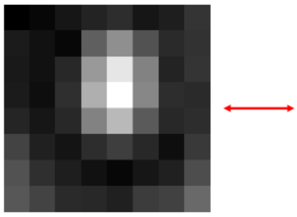



Fig.2: JPEG Image Division.

JPEG transform compression starts by breaking the image into 8x8 groups, each containing 64 pixels. Three of these 8x8 groups are enlarged showing the values of the individual pixels, a single byte value between 0 and 255

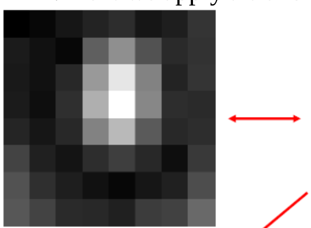
*B. Compression Steps*

1. We start with an image block (e.g. 8x8 pixels)



$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

2. next we apply a transform (e.g. 8x8 DCT)

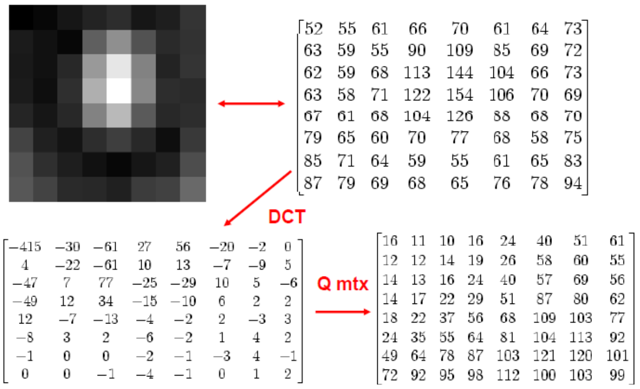


$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

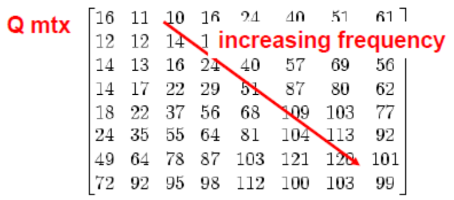
DCT

$$\begin{bmatrix} -415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\ 4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\ -47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\ -49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\ 12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\ -8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\ -1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\ 0 & 0 & -1 & -4 & -1 & 0 & 1 & 2 \end{bmatrix}$$

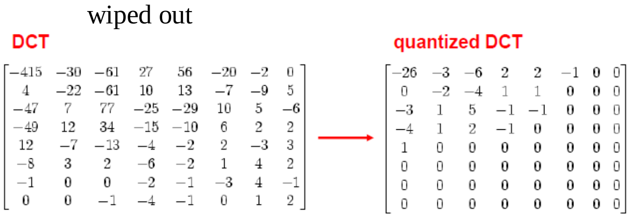
3. quantize with a varying Q



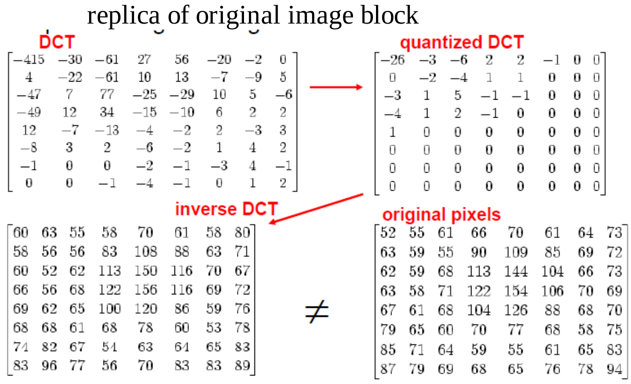

4. note that higher frequencies are quantized more heavily




5. in result, many high frequency coefficients are simply wiped out




6. this saves a lot of bits, but we no longer have an exact replica of original image block




## III. DCT COMPRESSION APPROACH

From computational considerations, it may be noted that direct application of the above equation to compute the transformed array requires $O(N^4)$ computations. Using *Fast Fourier Transform (FFT)*-like algorithm to compute the DCT, computations can be reduced to $O(2N^2 \log N)$. Such fast computational approaches and use of real arithmetic has made DCT popular for image compression applications. Since all natural images exhibits spatial redundancy, not all coefficients in the transformed array have significant values. This can be demonstrated by an example. We take an 8x8 block from Lena image, whose pixel intensities are shown in chapter 2.

### *A. 1D DCT Compression Approach*

From this equation we can immediately see that the DCT coefficients are real. To understand the better energy compaction it is interesting to compare the DCT to the DFT and it turns out that there is a simple relationship. We consider a sequence x[n] which is zero outside of {0… N-1} to relate need 21 DCT to DFT we three steps

Step 1): create a sequence

$$y[n] = x[n] + x[2N-n-1]$$
$$= \begin{cases} x[n], & 0 \le n < N \\ x[2N-n-1], & N \le n < 2N \end{cases}$$

Step 2): compute the 2N-point DFT of y[n]

$$Y[k] = \sum_{n=0}^{2N-1} y[n] e^{-j\frac{2\pi}{2N}kn}, \quad 0 \le k < 2N$$

Step 3): rewrite as a function of N terms only

$$Y[k] = \sum_{n=0}^{N-1} y[n] e^{-j\frac{2\pi}{2N}kn} + \sum_{n=N}^{2N-1} y[n] e^{-j\frac{2\pi}{2N}kn}$$

$$Y[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{2N}kn} + \sum_{n=N}^{2N-1} x[2N-n-1] e^{-j\frac{2\pi}{2N}kn}$$
$$= (m = 2N-1-n, \quad n = 2N-1-m) =$$
$$= \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{2N}kn} + \sum_{m=0}^{N-1} x[m] e^{-j\frac{2\pi}{2N}k(2N-1-m)}$$
$$= \sum_{n=0}^{N-1} x[n] \left\{ e^{-j\frac{2\pi}{2N}kn} + e^{j\frac{2\pi}{2N}kn} \underbrace{e^{-j\frac{2\pi}{2N}k2N}}_{1} e^{j\frac{2\pi}{2N}k} \right\}$$
$$= \sum_{n=0}^{N-1} x[n] \left\{ e^{-j\frac{2\pi}{2N}kn} + e^{j\frac{2\pi}{2N}kn} e^{j\frac{2\pi}{2N}k} \right\}$$

$$Y[k] = \sum_{n=0}^{N-1} x[n] \left\{ e^{-j\frac{2\pi}{2N}kn} + e^{j\frac{2\pi}{2N}kn} e^{j\frac{2\pi}{2N}k} \right\}$$
$$= \sum_{n=0}^{N-1} x[n] e^{j\frac{\pi}{2N}k} \left\{ e^{-j\frac{2\pi}{2N}kn} e^{-j\frac{\pi}{2N}k} + e^{j\frac{2\pi}{2N}kn} e^{j\frac{\pi}{2N}k} \right\}$$
$$= \sum_{n=0}^{N-1} 2x[n] e^{j\frac{\pi}{2N}k} \cos\left(\frac{\pi}{2N}k(2n+1)\right)$$
$$= e^{j\frac{\pi}{2N}k} \sum_{n=0}^{N-1} 2x[n] \cos\left(\frac{\pi}{2N}k(2n+1)\right)$$

$$Y[k] = e^{j\frac{\pi}{2N}k} C_x[k], \quad 0 \le k < 2N$$

$$C_x[k] = \begin{cases} e^{-j\frac{\pi}{2N}k} Y[k], & 0 \le k < N \\ 0, & otherwise \end{cases}$$

Summary

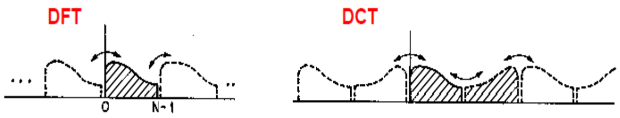
$$\underbrace{x[n]}_{N-pt} \leftrightarrow \underbrace{y[n]}_{2N-pt} \overset{DFT}{\leftrightarrow} \underbrace{Y[k]}_{2N-pt} \leftrightarrow \underbrace{C_x[k]}_{N-pt}$$

To understand the energy compaction property following point kept on the mind

- we start by considering the sequence y[n] = x[n]+x[2N-1-n]
- this just consists of adding a mirrored version of x[n] to itself


- next remember that the DFT is we identical to the DFS of the periodic extension of the sequence
- transform DFT: we work with extension of x[n]
- when transform is DCT: we work with extension of y[n]

$$\underbrace{x[n]}_{N-pt} \leftrightarrow \underbrace{y[n]}_{2N-pt} \overset{DFT}{\leftrightarrow} \underbrace{Y[k]}_{2N-pt} \leftrightarrow \underbrace{C_x[k]}_{N-pt}$$




Also gives us a fast algorithm for its computation and it consists exactly of the three steps

1) y[n] = x[n]+x[2N-1-n]

2) Y[k] = DFT{y[n]}

3)

$$C_x[k] = \begin{cases} e^{-j\frac{\pi}{2N}k} Y[k], & 0 \le k < N \\ 0, & otherwise \end{cases}$$

This can be computed with a 2N-pt FFT and the complexity of the N-pt DCT is that of the 2N-pt DFT

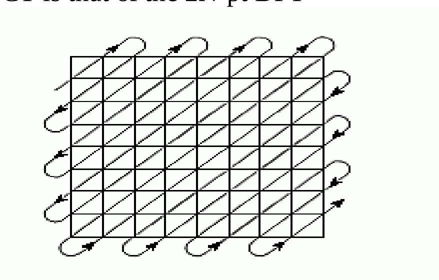


Figure 6: JPEG Serial conversion

A serpentine pattern used to convert the 8x8 DCT spectrums into a linear sequence of 64 values. This place all of the high frequency components together, where the large number of zero can be efficiently compressed with run-Lengh encoding.

*B. 2D-DCT Compression*

$$\underbrace{x[n_1,n_2]}_{N_1\times N_2-pt} \leftrightarrow \underbrace{y[n_1,n_2]}_{2N_1\times 2N_2-pt} \overset{2D\ DFT}{\leftrightarrow} \underbrace{Y[k_1,k_2]}_{2N_1\times 2N_2-pt} \leftrightarrow \underbrace{C_x[k_1,k_2]}_{N_1\times N_2-pt}$$

$$y[n_1,n_2] = x[n_1,n_2] + x[2N_1-1-n_1,n_2] + x[n_1,2N_2-1-n_2] + x[2N_1-1-n_1,2N_2-1-n_2]$$

$$C_x[k_1,k_2] = \begin{cases} e^{-j\frac{\pi}{2N_1}k_1} e^{-j\frac{\pi}{2N_2}k_2} Y[k_1,k_2], & \begin{matrix} 0 \le k_1 < N_1 \\ 0 \le k_2 < N_2 \end{matrix} \\ 0, & otherwise \end{cases}$$

## IV. RGB MODEL BASED ENCRYPTION AND DECRYPTION

The proposed work aimed at retrieving the query image from the images in the database by extracting the feature of shape and similarity is determined by using RGB projection. The step by step analysis of the proposed work is given below:

**Step 1:** Consider an image in the database as the query image

**Step 2:** Feature databases are considered from three different processing of the query image.

**Step 3:** The first feature database is considered from the border images extracted using SRM (Statistical Region Merging).

**Step 4:** Second feature database is generated by applying DCT on the query image.

**Step 5:** Third feature database is generated by obtaining the edge images from DCT by using Sobel in (BW) Black and white images.

**Step 6: RGB** projection is used to determine the similarity between the query image and the database images

*A. Proposed Worked*

This work also maintains a dictionary frequency which automatically classifies the test images from non target images. The classification is based on the similarities between the target and the non target images. The detailed steps are given below:

**Step 1:** test image is projected onto each one of the target and non target images in the dictionary.

**Step 2:** The target image is now compared with the sum of the images belonging to non target images to determine the similar image.

**Step 3:** Image comparer identifies the duplicate images and identifies the similar images automatically

**Step 4:** It scans the entire collection of database, analyzes its content and locates the similar images.

**Step 5:** returns the image pairs along with their similarity percentage.

*A. The Proposed Algorithm with Quantization*

The RGB value of an input image is directly obtained from Matlab. Rest of the operation and computation of our algorithm is carried out in simple matlab-source code. We have chosen matlab-program as we have to show practical real time image enhancement in DSP Image in RGB format space is converted into Y-Cb-Cr color space to find out luminance and chromatic component individually. Then Y, Cb, Cr component is split into (8×8) sub blocks respectively. Then for each sub block DCT-II is computed separately to obtain Y(u,v), Cb(u,v) and Cr(u,v) respectively, where Y(u,v), Cb(u,v) and Cr(u,v) represents the block transformed DCT coefficients and the first element of each DCT transformed coefficient Y(0,0), Cb(0,0) and Cr(0,0) represents DC component and rest are AC component. Each sub block after computing its DCT coefficient is normalized by a factor of 8. The proposed algorithm is implemented in four steps. In first step adjustment of local brightness is achieved. Local brightness is adjusted by mapping the DC coefficients of each sub block.

Quantization is achieved by compressing a range of values to a single quantum value. When the number of discrete symbols in a given stream is reduced, the stream becomes more compressible. A quantization matrix is used in combination with a DCT coefficient matrix to carry out transformation. Quantization is the step where most of the compression takes place.DCT really does not compress the image because it is almost lossless. Quantization makes use of the fact that higher frequency components are less important than low frequency components. It allows varying levels of image compression and quality through selection of specific quantization matrices. Thus quality levels ranging from 1 to 100 can be selected, where 1 gives the poorest image quality and highest compression, while 100 gives the best quality and lowest compression. As a result quality to compression ratio can be selected to meet different needs.JPEG committee suggests matrix with quality level 50 as standard matrix. For obtaining quantization matrices with other quality levels, scalar multiplications of standard quantization matrix are used. Quantization is achieved by dividing transformed image matrix by the quantization matrix used. Values of the resultant matrix are then rounded off. In the resultant matrix coefficients situated near the upper left corner have lower frequencies .Human eye is more sensitive to lower frequencies .Higher frequencies are discarded. Lower frequencies are used to reconstruct the image

## V.RESULT & CONCLUSION

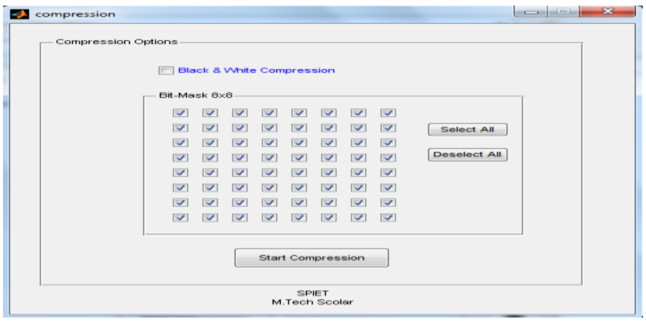



Fig.8: Image Compression Window using DCT with 8*8 Block


Fig.9: Image Compression image using DCT with 8*8 Block


Fig.10: The TxOutput Image


Fig.11: The Final Decrypted B/W Image

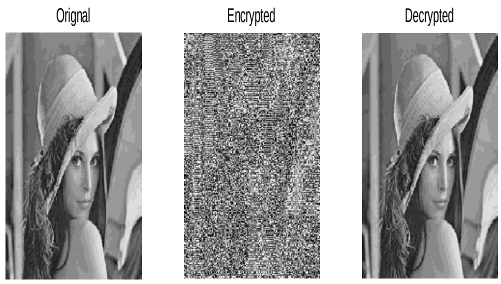



Fig.12: Comparison with public Key

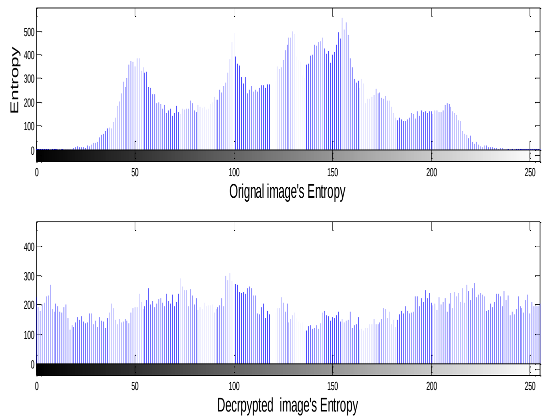



Fig.13: Comparison original image and decrypted Image


Fig.14: The Final Decrypted Image

## VI. RESULT AND CONCLUSION

In this work we proposed a new image encryption and compression method based on Embedding and Discrete Cosine Transform (DCT). Using DCT images can be compressed. For encryption, DCT blocks of transmitted images are rotated and mixed with a random image to hide them.

In the decryption stage, the covered images can be extracted from the mixtures by applying extraction algorithm. Finally using rotation keys and inverse discrete cosine transform, the original images can be reconstructed. Therefore we can achieve a fast and secure image transmission. In this paper color images used as original images, but grey color images can be applied in the same way.

Our future works include a more secure encryption method with an alternative rotation method and a reconstruction key. More complex rotation manner makes it harder for unauthorized people to reconstruct images without keys

## VII. REFERENCES

[1] *Andrew B. Watson* "Image Compression Using the Discrete Cosine Transform" Mathematical Journal, 4(1), 1994, p. 81-88

[2] Nageswara Rao Thota1, Srinivasa Kumar Devireddy2, "Image Compression Using Discrete Cosine Transform , "Georgian Electronic Scientific Journal: Computer Science and Telecommunications 2008|No.3(17)

[3] Prabhakar.Telagarapu, V.Jagan Naveen, A.Lakshmi..Prasanthi, G.Vijaya Santhi " Image Compression Using DCT and Wavelet Transformations", International Journal of Signal Processing, Image Processing and Pattern Recognition Vol. 4, No. 3, September, 2011

[4] N. AHMED, T. NATARAJAN, AND K. R. RAO "Discrete Cosine Transfonn" IEEE TRANSACTIONS ON COMPUTERS, JANUARY 1974

[5] Priyanka dixit1, Mayanka dixit,"Study of JPEG I"mage Compression Technique Using Discrete Cosine Transformation," International Journal of Interdisciplinary Research and Innovations (IJIRI)Vol. 1, Issue 1, pp: (32-35), Month: October-December 2013

[6] Maneesha Gupta*, Dr.Amit Kumar Garg** , "Analysis Of Image Compression Algorithm Using DCT", International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 ,www.ijera.com Vol. 2, Issue 1, Jan-Feb 2012,pp.515-521

[7] Robert M. Gray, IEEE, and David L. Neuhoff, IEEE" Quantization", IEEE Trans. on Information Theory, Vol. 44, NO. 6,pp. 2325-2383, OCTOBER 1998.(invited paper).

[8] RGB Image Compression Using Two Dimensional Discrete Cosine Transform" the international Arab journal of information technology Vol 9 , No 5, Sept. 2012

[9] Ronald A. DeVore, Bjorn Jawerth, and Bradley J. Lucier, Member,"Image Compression Through Wavelet Transform Coding" IEEE Trans. on Information Theory, Vol. 38. NO. 2, pp. 719-746, MARCH 1992.

[10] Shao, Dan; Kropatsch, Walter G. (February 3–5, 2010). "Irregular Laplacian Graph Pyramid" (PDF). Computer Vision Winter Workshop 2010 (Nove Hrady, Czech Republic: Czech Pattern Recognition Society).

[11] http://en.wikipedia.org/Image_compression.

[12] Greg Ames," Image Compression", Dec 07, 2002.

[13] Manish Kumar, D.C.Mishra, R.K.Sharma "A first approach on an RGB image encryption" Optics andLasersinEngineering52(2014)27–34

[14] D. C. Mishra∗ and R. K. Sharma† "Application of algebra and discrete wavelet transform in two-dimensional data (RGB-images) security" International Journal of Wavelets, Multiresolution and Information Processing Vol. 12, No. 6 (2014) 1450040 (25 pages)

**Anjali Sheoran** pursuing **MTECH** from SPGOI Rohtak, 2013-2015) and obtained **B.Tech in Electronics and Communication Engineering from** kurukshetra University in 2012. His main research interests includes: signal Processing, image processing and signal and system.