A Robust and Lossless Information Embedding in Image Based on DCT and Scrambling Algorithms

Dr. Mohammad V. Malakooti
Faculty and Head of Department of Computer Engineering,
Islamic Azad University, UAE branch, Dubai, UAE
malakooti@iau.ae

Mehrzad Khederzadeh
Student of Department of Computer Engineering,
Islamic Azad University, UAE branch, Dubai, UAE
mz_khederzadeh@yahoo.com

Abstract – We have presented a new secure, lossless information embedding based on the DCT and scrambling algorithms to be use for the data embedding. Our algorithm can be used to embed the vital information such as, bank credit cards, social security numbers, and other secret information into a cover image and transfer it over internet or other network systems. Our algorithm applied three levels of securities to implement high securities procedures on the original data before they are transmitted on the internet or other network system. The three security levels are as follows: Level -1 to scramble original data, Level 2 to apply encryption based on the XOR operation of the scrambled data with the secret keys, and level 3 of security for hiding the scrambled, and encrypted data in to mid-frequencies of the cover image, DCT coefficients.

Keywords: Credit card, transmission, Data hiding,Information Embedding,Watermarking,Scramble, Desramble.

I. INTRODUCTION

There are several techniques that can be used to hide or embed the vital information inside the cover images, videos or even the voice signals. There are also some applications in which the embedded information may not require to be retrieved from the cover image without distortion. A good example for this case is to hide the owner signature or the enterprise logo inside the cover image. However, there are some applications in which the embedded information is required to be retrieved from the cover image without distortion, i.e. the identification number, secret keys, and authentication patterns.

We proposed robust and lossless embedding techniques based on the Discrete Cosine Transforms (DCT) and Malakooti-Khederzadeh Scrambling-Descrambling MKSCDS algorithm to hide the vital information inside one cover image. Our algorithm applied three levels of security to make sure the hidden information, i.e. credit card numbers, are not detected by intruder and hackers.

In the first level of security we received the credit card of each bank customer and applied the MKSCDS algorithm to scramble all credit card numbers. In the second level of security, we generated some random keys by using Malakooti-Khederzadeh Randomized Key Generator (MKRKG) and then applied XOR operation on the elements of the scrambled credit card numbers with the generated keys to encrypt the scrambled data and increase the level of security. To apply the third level of security we embed the scrambled and encrypted credit card numbers into mid-frequencies of the DCT coefficients of the cover image, bank logo.

Since the data are embedded into the mid-frequency area of DCT coefficients of the bank logo it is protected against any hacker attack, such as possible high pass filtering.

Our Algorithm can be used to embed any vital information inside the cover image and then transmit it over the internet or any other communication facilities and network systems. The original credit cards numbers can be obtained exactly with a high rate of security and without any distortion from the received bank cover image, in which the credit card numbers are embed in it with three levels of security.

In this paper we used the Logo of Mellat Bank as the cover image to embed the credit card numbers in it. The size of cover image chosen to be 512 × 512, power of two, to be able to divide it into 8 × 8 blocks. Once the RGB cover image is converted into three matrices of Red, Green, and Blue, the elements of each matrix is divided into 64 x 64 blocks of 8 x 8. Then we applied the following algorithm on each 8 x 8 block.

1- Apply DCT function on each 8 × 8 block of the cover image.
2- Apply the MKSCDS algorithm on each credit number to scramble its content for the first level of security.
3- Generate the random keys using the MKRKG algorithm and apply XOR operation (encryption) on the elements of the scrambled credit card numbers with the elements of random keys.
4- Scale the value of each scrambled, encrypted credit card numbers (i.e. divide by 1000) to reduce its distortion effects on the DCT of cover image.
5- Embed the scaled, scrambled, and encrypted data into the mid frequency area of each 8 x 8 DCT block.
6- Apply the Inverse DCT function on each embedded DCT block matrix
7- Copy the 8 × 8 all IDCT blocks into the original Red , Green or Blue blocks

We repeat these seven steps for all 8 × 8 blocks of Red, Green and Blue matrices.

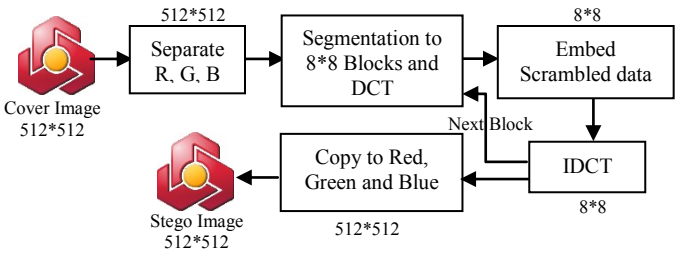

Fig 1: Data Embedding Processes

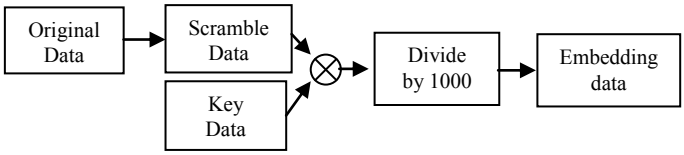

Fig. 2- Creating the Embedding Data by Using MKRKG Algorithm.

The rest of this paper is divided into the five sections as follows: Section II covers the related works and shows the most important research works conducted over the past few years. Section III explains the detail of our proposed method, about the scrambling, encryption, and data embedding as well as the extracting process at the received end. The performance analysis and experimental results is discussed in section IV. Conclusion and future works are discussed in section VI.

II. RELATED WORKS

The concept of data hiding was first introduced by David Lorge Parnas , a pioneer of software engineering, at Carnegie-Mellon University in 1972 [1]. He used the idea of modular programming presented by R. Gauthier and S. Pont in the book of Designing Systems Programs, published by Prentice-Hall [2]. He used the decomposition method to divide a large task in to m consecutive modules. We also used the concept of modularization and presented three consecutive modules for the data hiding. In our proposed method each module is responsible to apply one level of security to obtain secure and robust information embedding in the cover image.
The block diagram of data hiding JPEG encoder is shown on Figure 3. In this method the cover image is divided into smaller segment and DCT algorithm is applied on the each segment. Once the DCT blocks are obtained the main quantization table is applied on each DCT block, then the XOR operation is applied on each elements of secret data with the quantized DCT coefficients and coded by using a combination of the both run-length and Huffman coding.

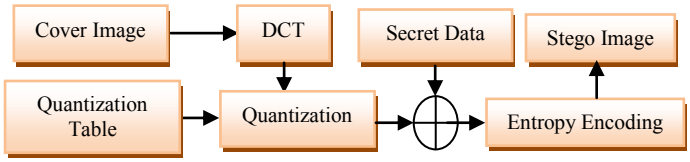

Fig 3: Data hiding JPEG encoder

The term "Digital Watermark" was first introduced in 1993, by A.Z.Tirkel, G.A.Rankin, R.M.VanSchyndel, W.J.Ho, N.R.A.Mee, C.F.Osborne [3]. They used two methods, fist one is based on the plane manipulation of the LSB, a fast decoding with streaming type of image processing and the second one utilizes the linear addition of watermark to the image data, difficult to decode but highly secure. Ridzon, Levisky and Kanocz [5] also have used an algorithm for embedding the information into the high frequency area. Chang, Chen and Chung [6] has embedded the secret data into the mid frequency area; meanwhile they changed the quantization component.

III. Proposed SDDE-MDCT Method

We have proposed a model in which the secret data or target data passed through our scrambling algorithm, then the result of scrambled data are embedded inside the mid-frequency region of the DCT coefficients. Our proposed model is called the Scramble/Descramble Data Embedding in Mid Frequency range of DCT coefficient (SDDE-MDCT).To hide or to embed the secret message or vital information inside the cover image we should divide our cover image into several 8 x 8 blocks.
Once the message data are scrambled, then the scrambled data are embedded into the mid frequency region of the DCT coefficients instead of high frequency to be protected from any high-pass filtering. To apply the higher level of security on the message data, the scrambled data are also encrypted, and then embedded inside DCT coefficients of the cover image. To demonstrate the performance of our algorithm, we have applied it on the credit cards of Mellat Banks's Customers. We used the bank Logo and divide it into several blocks of 8 x8 and hide the scrambled, encrypted, credit cards numbers into the mid-frequency region of the bank Logo DCT coefficients.

A) Original Message data (Credit Card Number)

4	0	0	0	0	8	0	7	0	6	2	0	0	0	0	2

B) Scrambled Message data Credit Card Number: Security Level- 1

8	6	0	0	0	2	0	0	7	0	2	0	0	0	4	0

C) Secret Keys Generated by M-K Randomized Key Generator

3	5	7	1	5	6	0	9	2	1	1	0	8	7	1	5

D) XOR the Scrambled Credit Card Number with Secret Keys: Security Level - 2

11	3	7	1	5	4	0	9	5	1	3	0	8	7	5	5

E) To apply the security Level-3 and the same time keep the shape of Bank's Logo unchanged, we have divided each scramble and encrypted digits by 1000 to reduce its effects. Then, the scrambled, encrypted, and scaled digits are inserted into the mid-frequency area of 8x8 blocks of message data (Credit Card Numbers)

							0.011
						0.003	0.005
					0.007	0.001	0.005
				0.001	0.003		
			0.005	0.000			
		0.004	0.008				
	0.000	0.007					
0.009	0.005						

Fig 4 : Locations which the Scrambled/Encrypted, and weighted Credit Card Numbers should be embedded

A. Scramble and Descramble Data Algorithm:

The main point of this paper is to use a secure, fast, and highly reliable Scramble/Descramble Algorithm that can be used as first level of security for hiding the message data(Credit Card

Number) inside the mid frequency range of DCT coefficient matrices as shown in Figure 4. In this approach the input data or messages are received and then scrambles by using our scrambled algorithm. Then the encryption algorithm is applied on the scrambled data based on our proposed SDDE-MDCT algorithm along with Malakooti Randomized Key generator. We can retrieve the original data by applying the inverse operation. It means we can Decrypt/Descramble the Weighted Credit Card Numbers to obtain the original message values.

First, we need to know the Size of original Data Array, i.e., the numbers of digits in credit cards are 16 (See Fig 4). The Original Data Array, *OData()*, has 16 elements and the values of each data elements lies between 0 to 9, see Figure 4. To enhance the security level we have started to process the elements of the *OData()*, from its S-th elements which is determined by the User or Administrator. The value of S parameter indicates the first elements of *OData()* that is used for the scramble algorithm.

We used Equations 1-4 to calculate the scrambled data array (S*cData( )*

$$Length = \sqrt{N} \quad (1)$$

$$K = \sum_{I=0}^{Length-1} \sum_{J=0}^{Length-1} J + I * 4 \quad (2)$$

$$L = \sum_{I=0}^{Length-1} \sum_{J=0}^{Length-1} (S + J * 4 + I) \bmod 16 \quad (3)$$

ScData (K) = OData (L) (4)

Descrambling of the data array is the reverse operation of the Scrambling Data Process as shown by Equations 5-8

$$Length = \sqrt{N} \quad (5)$$

$$L = \sum_{I=0}^{Length-1} \sum_{J=0}^{Length-1} J + 4 * I \quad (6)$$

$$K = \sum_{I=0}^{Length-1} \sum_{J=0}^{Length-1} (S + I + J * 4) \bmod 16 \quad (7)$$

DeSData (K) = ScData (L) (8)

In Equation 8 *DeSData* is our Descramble Data Array. Clearly after the processes are completed the values DeSData and OData arrays should be the same, DeSData = OData.

B. *M-K randomized Key Generator pseudo code*

In the second level of security we have used a robust M-K (Malakooti-khederzadeh) key generator algorithm to generate the randomized keys in which can be used for the XOR operations of these keys with the Scrambled Data. Our Algorithm can be shown as following:

Inputs = P, Q, M
Outputs = 16 keys in K array
Step1) suppose P,Q and M are three prime numbers :
P=11,Q=13 and M=3
Step2) A=P*Q, i=1
Step3) K_i=A mod 10
3.1) P=(K_i+1)*M
3.2) Q=int (P/Q)+i
3.3) A=P * Q, i=i+1
3.4) if i<=16 the go to step 3
Step4) end

C. *Embedding Algorithm*

Here are the main steps of the SDDE-MDCT algorithm to Scramble, Encrypt, and then embed the credit card numbers into the DCT blocks of the cover image:
Inputs: Cover Image and Credit Card Numbers
Outputs: Stego Image
Step-1) we need to have a bitmap image file because ***imread*** command in ***matlab*** program only works on bitmap images and it can divide the Bitmap Color Matrices into three blocks of 8*8 Red, Green and Blue matrices (Fig.1)

$$sub\_r = \sum_{rw=1}^{Dim/64} \sum_{cl=1}^{Dim/64} R(8rw-7:8rw, 8cl-7:8cl) \quad (9)$$

Equation 9 divided the Red matrices into the 8*8 Blocks of image data and similar equation is used for Green and Blue matrices.
Step- 2) Apply the DCT on each 8x8 block (Fig. 1)

$$DCT\_sub\_r = T \times sub\_r \times T' \quad (10)$$

Where the transformed matrix, T, is calculate by the following equations:

$$T_{ij} = \begin{cases} \frac{1}{\sqrt{N}} & if \quad i = 0 \\ \sqrt{\frac{2}{N}} \cos\left[\frac{(2_j + 1) i\pi}{2N}\right] & if \quad i > 0 \end{cases} \quad (11)$$

We assume that N=8, i and j are started from 1 to 8.
Step-3) ***OData(L)*** array keep the credit card numbers Based on the equations 1-4 we can calculate ***ScData (K).***
Step-4) By executing Malakooti Randomizedkey generator we have 16 random keys. Then create the Scrambled/ Encrypted data array using the following equation

$$E(i) = ScData(i) \otimes K(i) \quad (12)$$

Step-5) we have divided each Scrambled/Encrypted data elements into 1000 to reduce the energy of encrypted data prior to the insertion of the final result inside the mid-frequency region of the DCT coefficients.
We also have applied the IDCT on each blocks of 8 x 8 to retrieve the embedded information.
Step-6) Return to step-1 and repeat from step1 to 6 until all 8*8 sub-blocks have been processed.
Step-7) Once the embedding process is finished we have obtained the Stego Image that can be transferred over the network and carry the secret data with high security.

D. *Extracting Algorithm*

In order to have original numbers we should apply the inverse of embedding process in addition, to complete the extracting process the receiver have to know the keys for the extracting process. Here is the whole process of our extracting process
Inputs: Stego Image and Keys K(i)
Outputs: Original Image, Credit Card Numbers
Step-1) use the bitmap Image file and divide the Bitmap Color Matrices into three blocks of 8*8 for Red, Green and Blue matrices, as shown in Equation 9.
Step- 2) Apply the DCT Function on each divided 8x8 block, as shown in Equation 10

Step- 3) we know the location of 16 digits of each credit card (Fig.4), in which are embedded in the mid frequency area of the cover image, or bank logo. To extract and retrieve the original credit card numbers from the scrambled, encrypted, and scaled one we have to apply the inverse operation. First, we must multiply each value with the scale factor of 1000, to compensate the energy reduction process via scaling operation.

Step- 4) Second, Decrypt the data array using Equation 13

$$ScData(i) = E(i) \otimes K(i) \quad (13)$$

Step-5) Third, use the final result, scrambled data, and apply Descramble algorithm, ***DeSData,*** on the elements of the block. Then apply IDCT on the 8*8 blocks and copy these blocks into the red matrix image.

Step6) Return to step1 and repeat the same steps from 1 to 6 until all 8*8 blocks have been processed.

E. Capacity Estimation

We have presented an equation, Equation 14, to calculate the embedding capacity of the cover image based on the size of cover image as well as the levels of DCT applied on this image. The embedding capacity helps the user to calculate the appropriate number of DCT levels applied on the cover image as well as obtaining the optimum size of the each sub blocks that prevent the image distortion. Image distortion depends on several factors DCT levels and number of DCT coefficients that we used on each embedding process. Let *DL* be the DCT levels .To maintain the quality of transferable image we used just one Level of DCT for our proposed algorithm (*DL=1*) and checked the PSNR values for the embedding process of several Credit Card digits, i.e. *ND=16, 8, 4* digits. We assumed that cover image has a size of N x N and it has been divided into M x M blocks. Thus, the Embedding Capacity (*EC*), Total Digits that can be embedded into cover image, can be obtain from Equation 15 as following:

$$EC = 3 \times DL \times (\frac{N}{M})^2 \times ND \quad (14)$$

In this paper we used one DCT level, DL=1, size of image N=512 x 512 ,block size M =8 x 8 , and ND ={16,8,4}. Thus

ND=16 ➔ EC=196608 Digits ***or*** 12288 Credit Card Number

ND= 8 ➔ EC= 98304 Digits ***or*** 6144 Credit Card Number

ND= 4 ➔ EC= 49152 Digits ***or*** 3072 Credit Card Number

IV. HIDING COLOR INFORMATION'S IN DCT COEFFICIENTS

We can hide color information's (red, green and blue) into the DCT coefficients. On this method the color image is changed into Bitmap Image File and then divided into three matrices of Red, Green and Blue. Thus the Discrete Cosine Transform (DCT) is applied on the image block of size 8*8 obtained from segmentation process of the matrices. Then the scaled low frequency DCT coefficients of Green and Blue color matrices will be hidden inside the upper part of the mid frequency and the high frequency part of the Red matrix, respectively. The Inverse DCT (IDCT) of the resulting Red matrix will provide the compressed grayscale image. In the second stage the DCT of the resulting grayscale image is calculated and the DCT coefficients of the upper part of the mid frequency of the matrix as well its high frequency part will be rescaled and moved into upper left part of two matrices(low frequency parts) to create the Green and Blue Color matrices. The mid frequency and high frequency parts of these two matrixes will be initialized to zero. Moreover, the low frequency and lower mid frequency parts of the resulting grayscale image will be moved into a matrix called Red Color matrix. Then, the upper part of mid frequency and high frequency part of the Red matrix will be replaced with zero. The IDCT is applied on the resulting Red, Green, Blue matrices and the high quality Color Image is obtained from the composition of these matrices.

A. Color hiding process

Input: A 200x200 RGB image.

Output: A Gray Scale image.

Step1: Transfer the image into the Suitable Color Space and divide the main matrix into the Red, Green and Blue matrices.

Step 2: Apply the DCT Transform o each 8x8 Block of the Red, Green, and Blues matrices.

Step 3: Scale the DCT coefficients of low frequency part the of Green and Blue matrices and insert them in to the upper mid frequency and high frequency parts of the Red DCT matrix.

Step 4: Apply the IDCT on the resulting DCT coefficient matrix.

Step 5: Display the images.

B. Retrieved Original image

Step 1: Transfer the Gray Scale image into the suitable gray scale space.

Step 2: Apply DCT on each 8 x8 block of grayscale image and separate the resulting DCT matrix into the Red, Green and Blue matrices.

Step 3: Rescale the low frequency part of the DCT coefficients for Green and Blue matrices.

Step 4: Add zeroes to Red, Green and Blue DCT matrices to obtain the 8 x 8 blocks for each modified Red, Green and Blue DCT matrix.

Step 5: Calculate the IDCT of each 8 x 8 block.

Step6: Copy the resulting IDCT of each block into main Red, Green and blue matrixes and combine them to retrieve the original matrix.

Step7: Display the resulting images

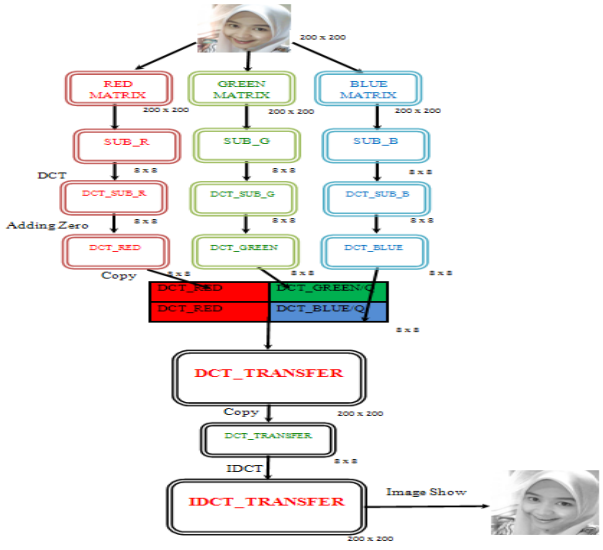

Fig 5 : color hiding process

Original Picture	Gray Level of Original Picture	Reconstructed Picture	Gray To Original Ratio	Resolution
 Size=117 Kb	 Size=41.7 Kb	 Size=114 Kb	35.6%	97.44
 Size=93 Kb	 Size=32.1 Kb	 Size=87 Kb	34.5%	93.55
 Size=128 Kb	 Size=44 Kb	 Size=119 Kb	34.3%	92.97
 Size=67 Kb	 Size=24Kb	 Size=59 Kb	35.8%	88.06

Fig6: compare compression and resolution of retrieved images

V. EXPERIMENTAL RESULTS

Following examples have been performed based on our algorithm to prove the resolution of our algorithm. The security level, quality of original image and credit card numbers are 3 factors that we should care about them in embedding and retrieving process. Several images has been tested and the retrieve images have been displayed algorithms were developed using ***Mat lab*** programming language.

		
(a)Mellat Bank	(b)Saderat Bank	(c)American home bank

Fig7: Six 512×512 RGB images served as test images

In our first experiment, we used three 512 x 512 RGB images, shown in Figure 7, as our cover images.

Figure 8 compare PSNR of our algorithm (SDEM-DCT) with LSB and KABAYASHI algorithms

		
SDEM (PSNR)=29.43 LSB (PSNR)=38.59 KABAYASHI(PSNR) =37.27	SDEM (PSNR)=28.93 LSB (PSNR)=39.3 KABAYASHI(PSNR) =36.72	SDEM (PSNR)=30.8 LSB (PSNR)=45.26 KABAYASHI(PSNR) = 41.11

Fig 8-compare PSNR of three different algorithms

Figure 9 shows the retrieved cover images obtained from the Cover images along with the values of the corresponding PSNR and Distortion Rate.

Cover Image	Retrieved Cover image	Image Distortion Rate
 Size=440 KB	 Size=431KB,PSNR=38	2%
 Size=326 KB	 Size=318KB,PSNR=34.6	2.5%
Size=480 KB	Size=468KB,PSNR=31	2.5%

Fig 9-compare the Size of cover image and retrieved image with correspond Distortion rate

VI. CONCLUSION AND FUTURE DIRECTIONS

In this paper, a lossless, high capacity data hiding method is proposed based on the DCT and scrambling algorithm to embed the vital information or Credit card numbers into the DCT coefficients of the cover image.

We have introduced a robust Scramble and Descramble algorithms along with M-K Randomize key Generator to increase the levels of security for the embedding process. First, the cover image is divided into blocks of 8 x 8 and the DCT is applied on each block. Then, the target information is scrambled and the XOR operation is applied on the scrambled data and secret keys. Finally the Scrambled/Encrypted information is embedded in to mid-frequency of area of the DCT coefficients for each block.

The embedding capacity calculated based on the number of DCT levels, the block size, and number of selected DCT coefficients in each block. Thus the embedding capacity is quite restricted. To improve the embedding capacity of the cover image, we have conducted several tests for the adaptive capacity estimation based on the levels of DCT, size of image block, and number of selected coefficient in mid-frequency.

The result of simulation indicates that we can change the indicated parameters and increase the embedding capacity but there is a tradeoff between the image quality and the embedding capacity. An adaptive smart algorithm is required to calculate the maximum capacity of the embedding algorithm while the image is not distorted and its quality satisfies some predefined threshold.

VII. REFERENCES

[1] D. L. Parnas, "On the Criteria to Be Used in Decomposing Systems Into Modules ", Communication of ACM, 1972, PP. 1053-1058.

[2] R. Gauthier, and S. Pont, Designing Systems Programs, (C), Prentice-Hall, Englewood Cliffs, N.J., 1970.

[3] A. Z.Tirkel, G.A. Rankin, R.M. Van Schyndel, W.J.Ho, N.R.A.Mee, C.F.Osborne. "Electronic Water Mark". DICTA 93, Macquarie University, Sydney, December 1993, PP. 666-673.

[4] K. B. Raja, Vikas, Venugopal K. R and L. M. Patnaik, "High Capacity Lossless Secure Image Steganography using Wavelets", International Conference on Advances Computing and Communications, PP. 230 – 235, 2006.

[5] R. Ridzon, D. Levisky and T. Kanocz, "Information Hiding within Still Images Based on the DCT Coefficients Flipping and Encryption", Fifty Second International Symposium, PP. 147 – 150, September 2010.

[6] C.C. Chang, T.S. Chen and L.Z. Chung, "A steganographic method based upon JPEG and quantization table modification", Information Sciences, 2002, 141, PP. 123–138.