Distributed Image Encryption Based On a Homomorphic Cryptographic Approach

Mamadou I. Wade
Dept. of Electrical Eng. and Comp. Sc.
Howard University
Washington, DC, USA
mamadou.wade@bison.howard.edu

Mohamed Chouikha
Dept. of Electrical Eng. and Comp. Sc.
Howard University
Washington, DC, USA

Tepper Gill
Dept. of Electrical Eng. and Comp. Sc.
Howard University
Washington, DC, USA
tgill@howard.edu

Wayne Patterson
Dept. of Electrical Eng. and Comp. Sc.
Howard University
Washington,DC, USA

Talitha M. Washington
dept. of Mathematics
Howard University
Washington, DC, USA

Jianchao Zeng
Senior Standards Advisor
Food and Drug Administration (FDA)
Silver Spring, MD, USA

Abstract— **The objective of this research is to develop a novel image encryption method that can be used to considerably increase the security of encrypted images. To solve this image security problem, we propose a distributed homomorphic image encryption scheme where the images of interest are those in the visible electromagnetic spectrum. In our encryption phase, a red green blue (RGB) image is first separated into its constituent channel images, and then the numerical intensity value of a pixel from each channel is written as a sum of smaller pixel intensity sub-values, leading to having several component images for each of the R, G, and B-channel images. A homomorphic encryption function is used to separately encrypted each of the pixel intensity sub-values in each component image using an encryption key, leading to a distributed image encryption approach. Each of the encrypted component images can be compressed before transmission and/or storage. In our decryption phase, each encrypted component image is decompressed if necessary, and then the homomorphic property of the encryption function is used to transform the product of individually encrypted pixel intensity sub-values in each encrypted component images, to the encryption of their sum, before applying the corresponding decryption function with a decryption key to recover the original pixel's intensity values for each channel image, and then recovering the original RGB image. Furthermore, a special case of an RGB image encryption and decryption where a pixel's intensity value from each channel is written as a sum of only two sub-values is implemented and simulated with a software. The resulting cipher-images are subject to a range of security tests and analyses. Results from these tests shown that our proposed homomorphic image encryption scheme is robust and can resist security attacks, as well as increases the security of the associated encrypted images. Our proposed homomorphic image encryption scheme has produced highly secure encrypted images.**

***Index Terms*— Distributed Image Encryption, Homomorphic Encryption, Image Encryption, Paillier Cryptographic System, RGB Image Encryption.**

I. Introduction

This research addresses information security problems also known as Cyber Security in general, and particularly image security. The security of encrypted images can always be improved through new encryption approaches and methods. Therefore, new encryption schemes that can efficiently protect information and counter any malicious Cyber behaviors are constantly researched. Our goal is to develop a novel homomorphic image encryption scheme that can be used to encrypt images before transmitting them through unsecured channels without compromising their contents, and then recover the encrypted images using a decryption process. The encryption scheme should also protect images when stored in computer servers or files. The application domains for our proposed homomorphic image encryption scheme include confidential images from satellites, military application images, industrial application images, some type of medical images, fingerprint images, and other images in the visible electromagnetic spectrum from any areas where there is a need to protect against security breach and to ensure their confidentiality and integrity. Image encryption schemes have been developed and presented in the literature for many years by various researchers from academia, industry, and other areas. Among these image encryption schemes, one can list chaos based encryptions, for some cases, use sequences or system of equations with chaotic behavior when encrypting a pixel intensity value from an image. The Chaos-based approach has been proposed by many researchers [4], [9], [15], [16], [23], [25]. For instance, Z. H. Guan, F. Huang, and W. Guan [25] proposed a Chaos-based image encryption algorithm where the position of the pixels in the special-domain is shuffled using the Arnold cat map, while each a pixel intensity value is changes using Chen's chaotic system. R. Tao, X. Meng, and Y. Wang in [17] have proposed an image encryption scheme based on Multiorders of Fractional Fourier Transforms (FRFT) from which they

978-1-7281-3885-5/19/$31.00 ©2019 IEEE

obtained an encrypted image by the summation of different orders inverse discrete FRFT of an image that is interpolated. In [12], L. D. Singh and K. M. Singh discussed an Elliptic Curve Cryptosystem in which an image encryption scheme is applied to a group of pixels to obtain a corresponding cipher-image. G. Ye and X. Huang proposed in [6] an image encryption scheme that uses an electrocardiography (ECG) signal to generate the initial encryption key used to encrypt a plain-image. In [11], J. Zhou, X. Liu, O. C. Au, and Y. Yan Tang proposed an efficient image encryption-then-compressed (ETC) system which operates in the prediction error domain and provides a high security level for its encrypted and compressed images.

A paradigm shift is proposed in this research where one plain-image is decomposed into multiple component images that are each encrypted to obtain multiple component cipher-images using a homomorphic function properties. The encrypted component cipher-images are then decrypted and combined to obtain the original image.

The organization of the paper is as follows: In Section II, the proposed image cryptographic scheme is presented. Section III provides the performance and security analysis results, while Section IV is the conclusion of the paper.

II. Proposed Image Cryptographic Scheme

Consider the following two propositions and associated encryption and decryption schemes [14].

Proposition 1

Let $g(i,j)$ be a 2-dimensional array representation of an image with M rows and N columns, where (i,j) is the spatial coordinate of each pixel, for $i=1,2,3,...,M$ and $J=1,2,3,...,N$. Let the image's data class be unsigned 8-bit integer, leading to having each pixel intensity value in the interval $[0,(L-1)] = [0,255]$, where $L=256$ is the number of pixel intensity levels. Let each pixel's intensity values y belongs to the finite Galois Field $Z_p = \{0,1,2,\dots(p-1)\}$, where p is a prime number chosen to be equal to $p=257$. For an 8-bit image, each pixel's intensity value y is in the interval $[0,255]$, but we chose $p=257$, the closet prime number to 255. This could lead to having a pixel intensity value of 256, which is out of the range $[0,255]$. So, if it happens that $y=256$, one can perform special processing to account for it, or map it $y=255$, which will have very little effect on the actual image because of redundancy or other factors. Note that the same concepts can be applied to images with data class unsigned 16-bit integer or others.

Proposition 2

Let E be a homomorphic Encryption function mapping from the finite field Z_p such that $E(y_1+y_2+y_3+\dots+y_k) = E(y_1)\times E(y_2)\times E(y_3)\dots E(y_k)$, for all y_k in Z_p, and k is a positive integer such that $1<k<L$, where L is the number of pixel's intensity levels. This means that the encryption of a sum of k pixel intensity sub-values y_1, y_2, $y_3,\dots,y_k$ is equal to the product of their individual encryption, and vice versa.

A. Homomorphic Image Encryption

Let E be a homomorphic encryption function, and let y be the intensity value of a pixel in image $g(i,j)$ where $i=1,2,3,\dots M$ and $j=1,2,3,\dots,N$, where M and N are the number of rows and columns of pixels in the digital image, respectively. One can write a pixel's intensity value y as a sum of k pixels' intensity sub-values as shown:

$$y = y_1+y_2+y_3+\dots+y_k = \sum_{n=1}^{k} y_n, \tag{1}$$

where the number of pixel components k, also called the number of pixel intensity sub-values, which also corresponds to the number of component images, is an integer such that $1<k<L$ where L is the number of pixel intensity levels.

When the number of pixel intensity sub-values is greater than the pixel intensity value, meaning that $k>y$, extra special processing is needed, where the difference $d=k-y$ can be used to find the original pixel value y. Now, to encrypt a pixel's intensity value y using the homomorphic Encryption function E, one can write:

$$\begin{aligned} E(y) &= E(y_1+y_2+y_3+\dots+y_k) && \text{(2a)}\\ &= E\left(\sum_{n=1}^{k} y_n\right) && \text{(2b)}\\ &= \Pi_{n=1}^{k}\left(E(y_n)\right) && \text{(2c)}\\ E(y) &= E(y_1)\times E(y_2)\times E(y_3)\dots E(y_k) && \text{(2d)} \end{aligned}$$

The final expression of $E(y)$ in (2)d has a profound implications. One can perform distributed and/or parallel or sequential encryption processing of each $E(y_k)$ simultaneously, or at different time using the same or different encryptions keys. Each $E(y_k)$ can also be computed by the same or different processors at the same or different locations. This can greatly increase the security of the encrypted image because an intruder may not have access to all $E(y_k)$ that can be stored at different locations, or transmitted at different time intervals. Also, if different encryption keys are used for each $E(y_k)$, opponents who have access to some of the decryption keys may not have access to other decryption keys, leading to not being able to decipher all corresponding encrypted component images without all the decryption keys. Also, each y_k can be randomly generated, the only requirement is that their sum should be equal to y. It is also important to note that the larger the value of k, the more secure the encrypted image is, but also the higher the computational cost.

In addition, each of the encrypted values $E(y_k)$ could be a very large integer, out of the range $[0,(L-1)]$, of the associated image's pixel intensity values. So, to make these $E(y_k)$ meaningful from an image point of view, one can apply $(\mod p)$ to each of the encrypted values $E(y_k)$, to map them back to Z_p, and obtain pixels' intensity values within the range $[0,(p-1)]$ that can be meaningful from an image point of view. For instance the pixel intensity values range is

$[0,255]$ for the case of an 8-bit image, and p can be chosen to be $p = 257$. So, we can write:

$$C_1 = E(y_1) \tag{3}$$

$$C_2 = E(y_2) \tag{4}$$

$$\vdots$$

$$C_k = E(y_k) \tag{5}$$

Applying $\mod p$ to the above equations, we have

$$C_{p1} = C_1 \mod p = E(y_1) \mod p \tag{6}$$

$$C_{p2} = C_2 \mod p = E(y_2) \mod p \tag{7}$$

$$\vdots$$

$$C_{pk} = C_k \mod p = E(y_k) \mod p \tag{8}$$

The quantities C_{p1}, C_{p2} ... ,C_{pk} represent the encrypted values for each of the pixel's intensity sub-values y_1, y_2, ... ,y_{pk}. They also represent the secure image pixel's intensity sub-values that will be transmitted or stored.

It is also important to note another quantity needed for the decryption. It is the greatest integer less than or equal to $(E(y_k)/p)$ also known as the floor of $(E(y_k)/p)$ or $\lfloor(E(y_k)/p\rfloor$. It also represents the quotient when $E(y_k)$ is divided by p. This quantity is not secrete but can also be encrypted by other means and transmitted at the transmitter side to increase security, or it can be computed at the receiver side. Without $\lfloor(E(y_k)/p\rfloor$ reconstruction of $E(y_k)$ for decryption purposes at the receiver side may be difficult.

So, we can write:

$$qt_1 = \lfloor(E(y_1)/p\rfloor \tag{9}$$

$$qt_2 = \lfloor(E(y_2)/p\rfloor \tag{10}$$

$$\vdots$$

$$qt_k = \lfloor(E(y_k)/p\rfloor \tag{11}$$

B. Homomorphic Image Decryption Phase

Let $C_{p1} = E(y_1) \mod p$, $C_{p2} = E(y_2) \mod p$, $C_{p3} = E(y_3) \mod p$, and in general $C_{pk} = E(y_k) \mod p$, be the individual encrypted pixel intensity sub-values mapped to Z_p, and available at the receiver side, where E is an homomorphic encryption function. Also, let $qt_1 = \lfloor(E(y_1)/p\rfloor$, $qt_2 = \lfloor(E(y_2)/p\rfloor$, $qt_3 = \lfloor(E(y_3)/p\rfloor$, ..., $qt_k = \lfloor(E(y_k)/p\rfloor$ be decryption parameters that are also available at the receiver side. To decrypt the encrypted pixel intensity value $E(y)$ and obtain the pixel intensity value y, one must first reconstruct or compute the individual encrypted pixel intensity sub-values $E(y_1)$, $E(y_2)$,$E(y_3)$, ..., and $E(y_k)$ as follows:

$$E(y_1) = qt_1 \times p + C_{p1} \tag{12}$$

$$E(y_2) = qt_2 \times p + C_{p2} \tag{13}$$

$$\vdots$$

$$E(y_k) = qt_k \times p + C_{pk} \tag{14}$$

where $qt_k \times p + C_{pk}$ is a different constant integer for each k value. Once the above quantities from (12) through (14) are computed, one can apply the corresponding decryption function D to the following product in (15) and recover y. First, compute the product.

$$E(y) = E(y_1) \times E(y_2) \times E(y_3) \times \ldots \times E(y_k) \tag{15}$$

Applying the decryption function gives:

$$D[E(y)] = D[E(y_1) \times E(y_2) \times \ldots \times E(y_k)] \tag{16}$$

$$D[E(y)] = D\left[\Pi_{n=1}^{k} E(y_n)\right] \tag{17}$$

$$D[E(y)] = D[E(\sum_{n=1}^{k} y_n)] \tag{18}$$

$$D[E(y)] = D[E(y_1 + y_2 + y_3 + \ldots + y_k)] \tag{19}$$

$$D[E(y)] = y_1 + y_2 + y_3 + \ldots + y_k \tag{20}$$

$$D[E(y)] = y \tag{21}$$

Note that the transition from (17) to (18) is achieved using the homomorphic property of the encryption function E. In addition, if the encryption/decryption keys for each individual pixel intensity sub-values y_k are different, one can first decrypt each $E(y_k)$, then add the sum $y_1 + y_2 + y_3 + \ldots + y_k$ to obtain the pixel intensity value y. For implementation efficiency, the image's pixel intensity values can be processed together as a matrix instead of single pixels.

C. Special Case Implementation for $k = 2$ Component Images

In order to implement the proposed image encryption scheme and to verify that the proposed theoretical approach will provide expected results, we chose to implement the special case for $k = 2$, where k is the number of pixel intensity sub-values $y_1, y_2, \ldots y_k$, which we also named the number of pixel components.

1) Special Case Encryption Phase Implementation for $k = 2$ Component Images : Let $y = y_1 + y_2$ and $E(y) = E(y_1) \times E(y_2)$. We need to have an encryption function, E, with the above homomorphic property, where the encryption of a sum of two pixel intensity sub-values y_1 and y_2 equal to the product of the individual encrypted sub-values $E(y_1)$ and $E(y_2)$.

Consider Paillier's Cryptographic System with its encryption and decryption functions [3], [18], [24], where a value y can be encrypted as follows:

$$E(y) = g^y x^N \mod N^2 \tag{22}$$

where $N = s \times q$, and s, q are two prime numbers, while x is a random number such that

$x \in Z_N^* = \{1, 2, \ldots, (N-1)\}$, and g is an integer whose order l is a multiple of N, that is $g^l \equiv 1(\mod N)$, a value of $g = 1+N$ satisfies this condition when prime numbers s and q have the same length.

One can also note that the Paillier encryption scheme is a public-key cryptographic system that is also probabilistic. For the encryption scheme to be secure, the public key N must be a very large integer with for example more than 300 digits. One can show that Paillier encryption function is homomorphic and satisfies (2d) for $k = 2$. Using (22), one can encrypt each of the two pixel intensity sub-values y_1 and y_2 with Paillier encryption function and the same public key N as follows:

$$C_1 = E(y_1) = g^{y_1} x_1^N \mod N^2 \tag{23}$$

and

$$C_2 = E(y_2) = g^{y_2} x_2^N \mod N^2 \tag{24}$$

Applying $\mod p$ as shown in (6) and (7) to the above equations (23) and (24) for $k = 2$ gives

$$C_{p1} = E(y_1) \mod p = [g^{y_1} x_1^N \mod N^2] \mod p \tag{25}$$

and

$$C_{p2} = E(y_2) \mod p = [g^{y_2} x_2^N \mod N^2] \mod p \tag{26}$$

The quantities C_{p1} and C_{p2} represent the cipher values corresponding to each of the pixel intensity sub-values y_1 and y_2. These encrypted values C_{p1} and C_{p2} represent the secure image pixel intensity sub-values that will be transmitted and / or stored.

2) Special Case Decryption Phase Implementation for $k = 2$ Component Images: For the special case where the number of pixel intensity sub-values $k = 2$, assume that the same key is used to encrypt y_1 and y_2, and C_{pk} and qt_k from (8) and (11) are available at the front end of the receiver. Before applying the decryption function D, one must first calculate the encrypted pixel intensity sub-values $E(y_1)$ and $E(y_2)$ using the expression from (14) for $k = 2$ as shown:

$$E(y_1) = qt_1 \times p + C_{p1} \tag{27}$$

and

$$E(y_2) = qt_2 \times p + C_{p2} \tag{28}$$

Using Eqs. (16) through (21) for $k = 2$, one can write

$$D[E(y_1) \times E(y_2)] = D[E(y_1 + y_2)] = D[E(y)] = y \tag{29}$$

When applying the Paillier Decryption function in the context of (2) d we can write:

$$C = E(y) = E(y_1) \times E(y_2) \tag{30}$$

and

$$y = \frac{L(C^\lambda \mod N^2)}{L(g^\lambda \mod N^2)} \mod N \tag{31}$$

$$y = \left[L(C^\lambda \mod N^2) \times \left(\left(L(g^\lambda \mod N^2)\right)^{-1} \mod N\right)\right] \mod N \tag{32}$$

where $N = s \times q$, and s, q are prime numbers, g can be set to be $g = 1+N$ when s and q have the same length as previously stated. The parameter λ is given by the least common multiple of $s-1$ and $q-1$, while the function $L(U)$ is defined as

$$L(U) = \frac{(U-1)}{N}. \tag{33}$$

3) Proposed Image Cryptographic Scheme Block Diagram: Figure 1 shows the block diagram for the special case implementation where the number pixels intensity sub-values, $k = 2$. It is possible to expand the component images homomorphic encryption sub-block in Fig. 1 to 3, 4, 5, 6, or more, based on the value of k, to produce more encrypted component images and increase the security.

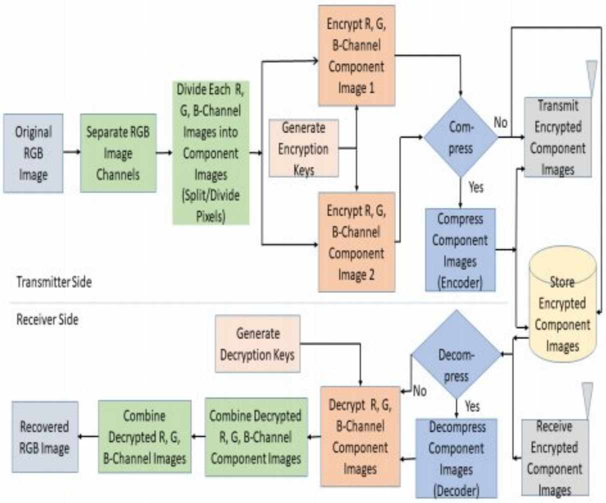

Original RGB Image
Separate RGB Image Channels
Divide Each R, G, B-Channel Images into Component Images (Split/Divide Pixels)
Encrypt R, G, B-Channel Component Image 1
Generate Encryption Keys
Encrypt R, G, B-Channel Component Image 2
Com-press
No
Yes
Compress Component Images (Encoder)
Transmit Encrypted Component Images
Store Encrypted Component Images
Transmitter Side
Receiver Side
Generate Decryption Keys
Decom-press
No
Yes
Decrypt R, G, B-Channel Component Images
Decompress Component Images (Decoder)
Receive Encrypted Component Images
Combine Decrypted R, G, B-Channel Component Images
Combine Decrypted R, G, B-Channel Images
Recovered RGB Image

Fig. 1: Proposed Special Case Image Cryptographic Scheme for $k = 2$ Component Images.

III. Performance and Security Analysis Results

The performance and security analyses are designed to verify that the proposed homomorphic image encryption scheme meets some required performance tests and can resist security attacks. This performance and security analyses include Correlation Analysis, Information Entropy, Cipher Cycle, Histogram Analysis, Chosen-Plaintext Attacks, and Brute force Attacks. Simulation results are obtained from a Mathematica software we have written and run using a Laptop computer with a processor having the following specifications: Intel (R) Core(TM) i3-4030U CPU @ 1.90 GHz 1.90 GHz.

A. *Input Output and Encrypted Component Images*

The test image (Baboon) used in this research is from the University of Southern California Signal and Image Processing Institute image set [22].
The recovered images and cipher-images are presented and discussed in this chapter. We randomly generate our private keys using our Mathematica code implementation, prime numbers s and q, with each 165 digits to be
$s =$ 13523180600716206144275068639190443169718519
7566367006618981784691676107437603053795350082927
6656130249253598608770933002374219101956801142939
47567777974339655680621
and
$q =$ 61599275563119689204225566492771724864296690
581662139357940729257768885406090709024547872520
964035633288261474750330107825840672796400600750
3687722380089774424939227.
The number of bits for s and q are 546 bits and 548 bits, respectively. Therefore, the value of the public key $N = s \times q$ used for encryption has 329 digits and 1093 bits, and is given by
$N =$ 833018128313352036165812028938761904191512668
90455136273236468160701838902880335783336786651206
9718197388149783138232536324446481100155132776532
0992255474004600188765660480640447542299262750458
8876720729766075784720708721480441898318354050969
8865985893995996890095100921352234625385021610968
21345036062518664770571190023246619967.
The original RGB image shown in Fig. 2 (a) is processed channel-wise by separating each of its R, G, and B-channel images which are shown in Figs. 2 (d), 2 (g) and 2 (j), respectively. Each of these channel images is processed separately by first decomposing it into two component images before encrypting each of them. For instance, the R-Channel original image R in Fig. 2 (d) is decomposed into two component images R1 and R2 which are not shown here, then encrypted to produce Encrypted Component Images R1 and R2 shown in Figs. 2 (e) and 2 (f), respectively. Similarly, the G and B-Channel original images have each two component images that are encrypted to produce Encrypted Component Images G1 and G2 shown in Figs. 2 (h) and 2 (i) for the G-Channel, while Figs. 2 (k) and 2 (l) correspond to the encrypted component images for the B-Channel original image. Fig. 2 (b) shows the histogram of the original Baboon image that is discussed next, while Fig. 2 (c) displays the recovered RGB image obtained by combining the recovered R, G, and B-Channel images.
The number of component images is not just limited to the special case of two component images for each channel. For instance, the R-Channel component images can also be extended to R3, R4, R5, ...Rk as explained in section II.
On the receiver side, the encrypted component images for each channel are used to recover the corresponding original channel images. For instant, the encrypted component images R1 and R2 in Figs. 2 (e) and 2 (f) are used to recover the R-Channel image shown in Figure 3 (a). Similar approach is taken for the G and B-Channel images. Also, the recovered R, G, and B-Channel images in Fig. 3 are combined to form the recovered RGB image in Fig. 2 (c).

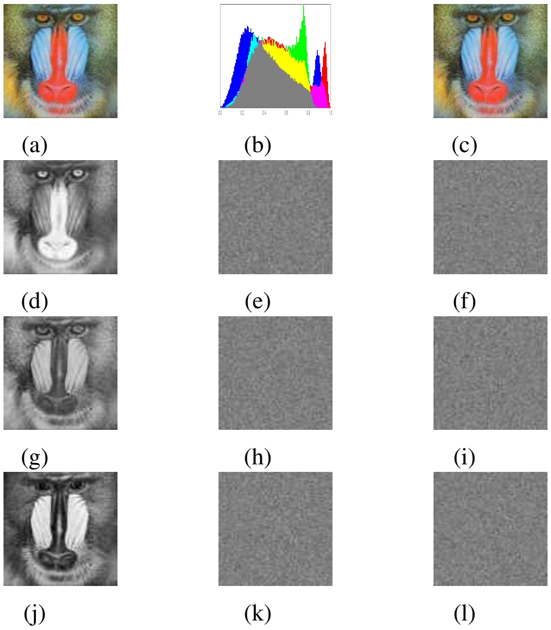

(a) (b) (c)
(d) (e) (f)
(g) (h) (i)
(j) (k) (l)

Fig. 2: (a) Original Baboon RGB Image;
(b) Histogram of Original Baboon RGB Image;
(c) Recovered Baboon RGB Image;
(d) Original R-Channel Baboon Image R;
(e) Encrypted Baboon Component Image R1;
(f) Encrypted Baboon Component Image R2;
(g) Original G-Channel Baboon Image G;
(h) Encrypted Baboon Component Image G1;
(i) Encrypted Baboon Component Image G2;
(j) Original B-Channel Baboon Image B;
(k) Encrypted Baboon Component Image B1;
(l) Encrypted Baboon Component Image B2

B. *Histogram Analysis*

The histogram of a digital image provides information about the distribution of its pixels intensity values. For an

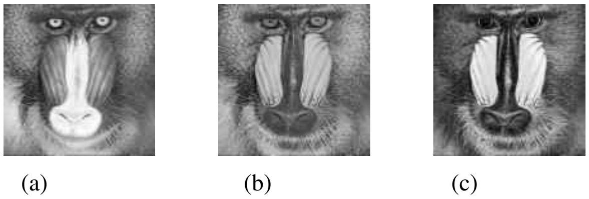


Fig. 3: (a) Recovered R-Channel Baboon Image
(b) Recovered G-Channel Baboon Image
(c) Recovered B-Channel Baboon Image

image with intensity levels in the discrete interval $[0, L-1]$, the histogram is given by the discrete function $h(l) = n_l$, where l is the l^{th} intensity value, and n_l represents the number of pixels in the image with intensity value l. [19]

A histogram analysis of the cipher-images produced by our proposed homomorphic image encryption scheme is performed for each channel image and its associated encrypted component images. So, the histogram analysis for the R-Channel original image R in Fig. 4 (a) is shown in Fig. 4 (d), and it is nonuniform, while the encrypted component images R1 and R2 in Figs. 4 (b) and 4 (c) have their histograms in Figs. 4 (e) and 4 (f), respectively; which also show that the encryption algorithm is able to produce a uniform-like distribution of the pixel intensity values for each of the encrypted component images; and therefore, can resist histogram analysis attacks.

Similar histograms for the G, and B-Channel original images, and associated encrypted component images G1, G2, B1, and B2 also provide similar results.

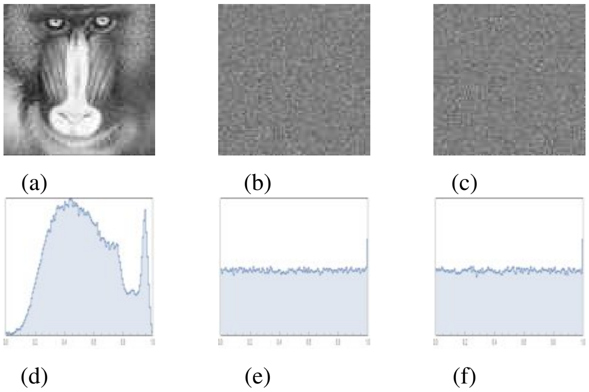


Fig. 4: (a) Original R-Channel Baboon Image R;
(b) Encrypted Baboon Component Image R1;
(c) Encrypted Baboon Component Image R2;
(d) Histogram of Original R-Channel Baboon Image R;
(e) Histogram of Encrypted Baboon Component Image R1;
(f) Histogram of Encrypted Baboon Component Image R2

C. Cipher Cycle

One of the requirements of an image encryption scheme is to produce an encrypted image that is very different from its original plain-image. To quantify this difference between the corresponding pairs of plain-image and cipher-image, one can use two criteria denoted by the Number of Pixel Change Rate (NPCR) and the Unified Average Change Intensity (UACI) [5], [9], [10], [20], [21]. The NPCR and UACI also may be used as a security test to protect against differential attacks, which consist of making a slight change on the cipher-image and observing the changes on the result. On one hand, the expression for the NPCR, which measures the average number of pixels in difference of a color component between two images C and C', is given by:

$$NPCR_{R,G,B} = \frac{\sum_{i,j} D_{R,G,B}(i,j)}{N} \times 100\% \tag{34}$$

where N is the image's total number of pixels, and the definition of $D_{R,G,B}$ is given by:

$$D_{R,G,B}(i,j) \triangleq \begin{cases} 0, & if \quad C_{R,G,B}(i,j) = C'_{R,G,B}(i,j) \\ 1, & if \quad C_{R,G,B}(i,j) \neq C'_{R,G,B}(i,j) \end{cases} \tag{35}$$

where $C_{R,G,B}(i,j)$ and $C'_{R,G,B}(i,j)$ represent the values of corresponding color component R, G, and B in images C and C', respectively.

Given two random images, an expression for the expected value of $NPCR$ can be found to be

$$\mathbb{E}(NPCR) = (1 - 2^{-L_{R,G,B}}) \times 100\% \tag{36}$$

where $L_{R,G,B}$ represents the number of bits used to encode each color components R, G, or B. For instance, given two random images each with size 512×512 and 24-bit true color with 8 bits for each R, G, and B channels $(L_R = L_G = L_B = 8)$, the expected value of the NPCR is given by:

$$NPCR_R = NPCR_G = NPCR_B = 99.609375\% \tag{37}$$

One the other hand, the expression for the $UACI$ is defined as

$$UACI_{R,G,B} = \frac{1}{N}\left[\sum_{i,j} \frac{\left|C_{R,G,B}(i,j) - C'_{R,G,B}(i,j)\right|}{2^{L_{R,G,B}}-1}\right] \times 100\% \tag{38}$$

where $L_{R,G,B}$ represents the number of bits used for each color component of Red (R), Green (G), or Blue (B), respectively. Given two random images, the expected value of $UACI_{R,G,B}$ is given by

$$\mathbb{E}(UACI_{R,G,B}) = \frac{\frac{1}{2^{2L_{R,G,B}-1}}\left(\sum_{i=1}^{2^{L_{R,G,B}}-1} i(i+1)\right)}{2^{L_{R,G,B}}-1} \times 100\% \tag{39}$$

For the case of an RGB image where each channel is encoded using 8 bits, we have the following expected value:

$$\mathbb{E}(UACI_R) = \mathbb{E}(UACI_G) = \mathbb{E}(UACI_B) = 33.46354\% \tag{40}$$

The NPCR and UACI analyses comparing the plain-images and cipher-images produced by our image encryption scheme is performed and results for the B-Channel are shown in Table I. Similar results are also obtained for R and G-Channel images but not shown here. As can be seen in Table I, the NPCR between the original B-Channel image B and its associated encrypted component images B1 and B2 is 99.6136 and 99.6273, respectively. These values are very close to the expected value of 99.60937, and therefore, the encryption algorithm performs very well on changing the pixel intensity values in the B-Channel component images B1 and B2. The values for the UACI between the original B-Channel image B and its corresponding encrypted component images B1 and B2 shown in Table I, are very close to the expected value of 33.4635; therefore, the encryption algorithm perform also well for this case.

TABLE I: NPCR and UACI for B-Channel Original and Encrypted Images

B-Channel			
Test Type	B and B1	B and B2	Expected Value
NPCR (%)	99.6136	99.6273	99.60937
UACI (%)	31.3253	31.3231	33.4635

D. Correlation Analysis Results and Discussions

Many images in the visible electromagnetic spectrum range have redundancies that sometimes lead to a high correlation between neighboring pixels at the horizontal, vertical, and diagonal directions. A good image encryption scheme should break this correlation between neighboring pixels and reduce its value to almost zero; and therefore, producing an image more resistant to statistical attacks. The expression for a correlation coefficient between adjacent pixel pairs in an image is discussed by several authors that include A. Soleymani, A. Daneshgar, A. Kano [2], [4], [5], [7]–[10], [12], [20], [21].

$$r_{XY} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{41}$$

where $cov(X,Y)$ and $\sigma_X \sigma_Y$ are the covariance and product of the standard deviation of X and Y, respectively. The covariance and standard deviation for pairs of adjacent pixels have the following forms

$$cov(X,Y) = \frac{1}{N}\sum_{i=1}^{N}\left[(x_i - \mu_X)(y_i - \mu_Y)\right] \tag{42}$$

where x_i and y_i are values of adjacent pixel pairs selected at random, N the total number of adjacent pixel pairs (x_i, y_i) from the image, μ_X and μ_Y are the mean or expected values of X and Y, respectively, and are given by

$$\mu_X = \frac{1}{N}\sum_{i=1}^{N} x_i \quad \text{and} \quad \mu_Y = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{43}$$

Using the variance, the expression for the standard deviation σ_X and σ_Y are:

$$\sigma_X = \left(\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_X)^2\right)^{\frac{1}{2}} \quad \text{and} \quad \sigma_Y = \left(\frac{1}{N}\sum_{i=1}^{N}(y_i - \mu_Y)^2\right)^{\frac{1}{2}} \tag{44}$$

A correlation analysis comparing each channel original images and their associated encrypted component cipher-images is conducted using a set $N = 2500$ random pairs of adjacent pixels in the horizontal and vertical directions. Results for this correlation analysis for the G-Channel images is shown in Table II. The correlation analyses for the R and B-Channel images are also conducted, and they provide similar results.

Table II shows that adjacent pairs of pixel intensity values in the original G-Channel image G are highly correlated in the horizontal and vertical directions, with correlation coefficient values close to 1. However the encrypted component images G1 and G2 have very uncorrelated adjacent pairs of pixels' intensity values because the associated correlation coefficients are close to 0. These results for the G-Channel image correlation analyses are also displayed in Fig. 5 Based on these results, one can conclude that the proposed image encryption scheme performed very well breaking up the correlation of adjacent pixel pairs in the horizontal and vertical directions; and therefore increases the security of the encrypted images by making them more resistant to statistical correlation attacks.

TABLE II: Correlation Coefficients of 2500 Adjacent Pixel Pairs for Baboon G-Channel Images

G-Channel			
Direction	Original G	Encrypted G1	Encrypted G2
Horizontal	0.86158	−0.000659307	0.00697017
Veritical	0.760878	0.0401025	−0.00679211

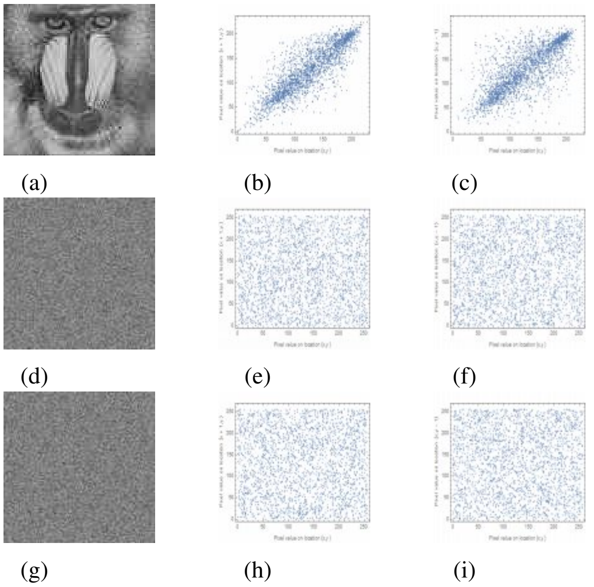


Fig. 5: (a) G-Channel Original Baboon Image G;
(b) Horizontal Correlation of Image G;
(c) Vertical Correlation of Image G;
(d) Encrypted Component Image G1;
(e) Horizontal Correlation of Image G1;
(f) Vertical Correlation of Image G1;
(g) Encrypted Component Image G2;
(h) Horizontal Correlation of Image G2;
(i) Vertical Correlation of Image G2
$N = 2500$ Adjacent Random Pixel Pairs.

E. Information Entropy

The degree of uncertainty in a random variable can be evaluated using its Information Entropy. Among all features of randomness, entropy is one of the most important. Given a source S with N symbols, where $N = 2^k$ and k the number of bits used to represent a symbol S_i, one can obtain the information entropy $h(S)$ as follows [4], [5], [10], [19], [20], [23]:

$$h(S) = -\sum_{i=0}^{N-1} p(S_i)\log_2[p(S_i)] \tag{45}$$

where $p(S_i)$ is the probability of occurrence for the symbol S_i, N the total number of symbols generated by the source, and the log base 2 is used in order to express the entropy in bits. When S is a truly random source, $p(S_i) = \frac{1}{2^k}$ for all i, the Entropy can be calculated to be

$$h(S) = k \tag{46}$$

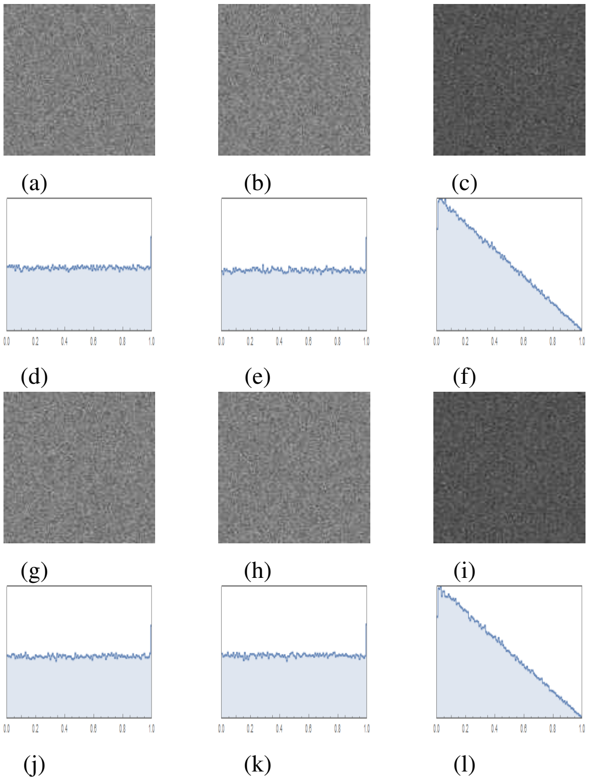
TABLE III: Entropy Analysis for Baboon R, G, and B-Channel Images

R-Channel		
Encrypted R1	Encrypted R2	Expected Value
7.94141	7.94136	8
G-Channel		
Encrypted G1	Encrypted G2	Expected Value
7.94062	7.94512	8
B-Channel		
Encrypted B1	Encrypted B2	Expected Value
7.94333	7.94229	8

Results of our entropy analysis are given in Table III.
Table III shows the entropy values for each of the encrypted channel component images R1, R2, G1, G2, B1, and B2, are very close to the expected value of 8. So, these encrypted component images are truly random; and therefore, our proposed homomorphic image encryption scheme is robust enough to protect against Entropy attacks.

F. Chosen-Plaintext Attacks

In the chosen plaintext attacks, the opponents have access to at least a pair of plaintext and ciphertext in addition to the encryption algorithm and try to find the structure of the encryption key. If the key is found, all past and future ciphers encrypted using this key can be decrypted using the found key [12], [13], [23]. Our proposed Homomorphic cryptographic approach can resist this type of attack because it always produces a difference cipher-image when the same original image is encrypted several times using the same encryption key as shown in Fig. 6 . Similar results are also obtained for the encrypted G, and B-Channel images but not shown here.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Fig. 6: (a) Encrypted Comp. Image R11 and its Hist. in (d); (b) Encrypted Comp. Image R12 and its Histogram in (e); (c) Pixel-wise difference $|R11 - R12|$ and its Histogram in (f); (g) Encrypted Component Image R21 and its Histogram in (j); (h) Encrypted Component Image R22 and its Histogram in (k); (i) Pixel-wise difference $|R21 - R22|$ and its Histogram in (l).

G. Timing and Recovered Image Quality Analyses Results and Discussions

This section presents and discusses the results for the timing and quality of the recovered images in Tables IV and V for the R-Channel. The timing simulation results are obtained using a Laptop computer with an Intel (R) processor with Core(TM) i3-4030U CPU @ 1.90 GHz 1.90 GHz, and they can change when using a processor with a different speed. Timing results can also change based on the efficiency of the algorithms used to implement the encryption and decryption functions. In Table IV, the columns denoted by component image R1 encryption time (R1ET) and component image R2 encryption time (R2ET) give the time in seconds (S) or minutes (min) it takes to encrypt component images R1 and R2, respectively, while the column decryption time (DT) provides the time needed to decrypt and recover the R-Channel original image. In Table V, The column NPCR (Number of Pixels Change Rate) gives the data that compares the R-Channel original and recovered images in order to show the percentage of corresponding pixel values that have changed as a result of the encryption and decryption processes, when the encrypted images are not compressed. It is desired to have the values on this NPCR colunm as small as possible, meaning that the original and recovered R-Channel images are almost the same and very little information is lost during the encryption and decryption operations. The column RGBNPCR gives the data that compares the original and recovered RGB images similar to the previous NPCR column. Finally, the last column named Quality includes terms such as VL = Very Low, H = High, VH = Very High are subjective qualitative description used to judge the quality of the recovered images based on the percentage results shown in the previous two columns, namely columns NPCR and RGBNPCR; which show that very high quality recovered images can be obtained when the number of bits of the public encryption key N is ≥ 20. Similar tables are obtained for the G and B-Channel.

TABLE IV: R-Channel Timing Analysis

R-Channel:

n (Digits)	n(Bits)	R1ET	R2ET	DT
4	11	31.42 S	30.89 S	16.27 S
8	27	55.59*S*	54.25 S	29.85 S
20	65	2.18 min	2.17 min	1.79 min
60	197	6.05 min	6.03 min	5.99 min
100	330	10.60 min	10.54 min	10.81 min
150	496	17.32 min	17.73 min	18.65 min
200	662	23.42 min	23.38 min	25.75 min
249	827	32.95 min	32.57 min	38.73 min
329	1093	53.18 min	53.69 min	64.43 min

TABLE V: R-Channel Recovered Image Quality Analysis

R-Channel:

n (Digits)	n(Bits)	NPCR	RGBNPCR	Quality
4	11	16.56%	41.89%	VL
8	27	0.068%	0.18%	H
20	65	0.025%	0.055%	VH
60	197	0.0256%	0.056%	VH
100	330	0.026%	0.056%	VH
150	496	0.0256%	0.056%	VH
200	662	0.0256%	0.056%	VH
249	827	0.0256%	0.056%	VH
329	1093	0.0256%	0.056%	VH

H. Image Compression Results and Discussions

This section provides the image compression results shown in Table VI, for the R-Channel. For each encrypted channel component image R1 and R2, the number of bytes before and after compression is found using our simulation program. The last column of this table shows a reduction in encrypted image data size by a factor approximately equal to 3.5 for the R, G, and B-Channel encrypted images. This means that less bandwidth and time are needed to transmit the encrypted data, and less memory is needed to store this encryption information for future decryption processing. Similar results are also obtained for the G and B-Channel encrypted images.

TABLE VI: Image Compression Results

R-Channel

Images	Before (Bytes)	After (Bytes)	Reduction
Encrypted R1	2097304	594648	1 : 3.53
Encrypted R2	2097304	594768	1 : 3.53

IV. Conclusion

In this research, a novel image encryption scheme that uses a homomorphic function property to encrypt an image and produce more than one cipher-image for each plain-image is proposed. During the encryption phase, an original RGB image is separated into its R, G, and B-channel images, then each pixel intensity value in each channel image is divided or decomposed into a sum of several pixel intensity sub-values to produce many component channel images that are separately encrypted using the same encryption key, compress if necessary, and then transmitted or stored. On the decryption side, the encrypted component channel images are decompressed if necessary, then decrypted using the same key and combined to produce each of the R, G, and B-channel recovered images that are also combined to obtain the recovered RGB image. Simulation results show that the associated component cipher-images can withstand a wide range of security and analysis attacks including Histogram Analysis, Entropy Analysis, Correlation Analysis, Chosen-Plaintext Attacks, Brute Force Attacks, and others. Also, high quality recovered channel images and recovered RGB image is obtained, meaning that very little information is lost as a results of applying our proposed encryption, decryption, and other image cryptographic and processing actions. Our proposed homomorphic image encryption scheme can be used in non real-time applications whereby highly secure encrypted images are needed, such as confidential satellite images, some confidential medical images, confidential fingerprint images, and any confidential images in the visible electromagnetic spectrum range. However real-time applications may be possible if faster encryption and decryption algorithms are implemented, as well as faster microprocessors or hardware are used instead.

Our main contribution is the formulation of a novel homomorphic image encryption scheme where each pixel intensity value in the original image is written as a sum of several sub-values, leading to producing many component images that are encrypted to produce many corresponding cipher-images; and therefore, increase the security of the associated images. The formulation includes encryption and decryption phases, as well as a block diagram for a special case implementation.

Future research could include the use of a homomorphic encryption and decryption functions that will require less computational time, for possible real-time applications. Our proposed encryption scheme could also potentially be extended to video encryption when using the appropriate homomorphic functions and special rapid processing approaches. Our proposed approach can be applied to any data that can be mapped to number greater than 1. For instance, alphabet letters from a to z mapped to numbers greater than 1 can be encrypted using our approach by writing each number as a sum of several numbers, then use a homomorphic encryption function to encrypt each and take the necessary steps as described in our approach to increase security.

References

[1] A. Daneshgar and B. Khadem, *“A self-synchronized chaotic image encryption scheme,"* Signal Processing: Image Communication 36 (2015) 106-114.
`www.elsevier.com/local/image`

[2] A. Soleymani, Md. J. Nordin, and Z. Md. Ali, *“A novel public key Image encryption based on elliptic curves over prime group field,"* Journal of Image and Graphics, Vol. 1, No. 1, March, 2013.

[3] A. K. A. Hassan, *“Reliable implementation of Paillier cryptosystem,"* Iraqi Journal of Applied Physics, IJAP, Vol. 10, No. 4, October-December 2014, pp. 27-29

[4] A. Daneshgar and B. Khadem, *“A self-synchronized chaotic image encryption scheme,"* Signal Processing: Image Communication 36 (2015) 106-114.
`www.elsevier.com/local/image`

[5] A. Kanso, M. Ghebleh, *“ A novel image encryption algorithm based on a 3D chaotic map,"* Commun Nonlinear Sci Numer Simulat 17 (2012) 2943–2959,
`www.elsevier.com/locate/cnsns`

[6] G. Ye and X. Huang, *“An image encryption algorithm based on autoblocking and electrocardiography,"* Published by the IEEE Computer Society. April-June 2016.

[7] G. Zhang, and Q. Liu, *“A novel image encryption method based on total shuffling scheme,"* Optics Communications 284 (2011) 2775–2780,
`www.elsevier.com/locate/optcom`

[8] G. Chen, Y. Mao, and C. K. Chui, *“A symmetric image encryption scheme based on 3D chaotic cat maps,"* Chaos, Solitons and Fractals 21 (2004) 749–761,
`www.elsevier.com/locate/chaos`

[9] H.S. Kwok, Wallace K.S. Tang, *“A fast image encryption system based on chaotic maps with finite precision representation, "* Chaos, Solitons and Fractals 32 (2007) 1518–1529,
`www.elsevier.com/locate/chaos`

[10] H. Liu, X. Wang, and A. kadir, *“Image encryption using DNA complementary rule and chaotic maps , "* Applied Soft Computing 12 (2012) 1457–1466,
`www.elsevier.com`

[11] J. Zhou, X. Liu, O. C. Au, and Y. Yan Tang, *“Designing an efficient image encryption-then-compression system via prediction error clustering and random permutation,"* IEEE Transactions on Information Forensics and Security, VOL. 9, NO. 1, January 2014.

[12] L. D. Singh and K. M. Singh, *“Image Encryption using elliptic curve cryptography, "* Procedia Science 54 (2015) 475-481,
`www.sciencedirect.com`

[13] M. Kumar, D. C. Mishra, and R. K. Sharma, *“A first approach on an RGB image encryption, "*Optics And Lasers in Engineering 52 (2014) 27–34,
`www.elsevier.com/locate/optlaseng`

[14] Mamadou I. Wade, *“ Distributed mage encryption based on a homomorphic cryptographic approach , "* Ph.D. Dissertation, Howard University, May 2017

[15] N. K. Pareek, V. Patidar, and K. K. Sud, *“Image encryption using chaotic logistic map, "* Image and Vision Computing 24 (2006) 926-934,
`www.elsevier.com/locate/optlaseng`

[16] P. P. Dang and P. M. Chau, *“Image encryption for secure internet multimedia applications, "* IEEE Trans. On Consumer Electronics, Vol. 46. No. 3, August 2000.

[17] R. Tao, X. Meng, and Y. Wang, *“Image encryption with multiorders of fractional fourier transforms, "* IEEE Trans. Inf. Forensics and Security, Vol. 5, No 4, Dec 2010.

[18] R. Rivest, Lecture Notes 15, Computer and Network Security: *“Voting, homomorphic encryption,"* October, 2002

[19] R. C. Gonzalez and R. E. Woods, *“Digital image processing.,"*3rd ed. Person Education Inc., 2008.

[20] R. Rhouma, S. Meherzi, and S. Belghith, *“OCML-based colour image encryption, "* Chaos, Solitons and Fractals 40 (2009) 309–318,
`www.elsevier.com/locate/chaos`

[21] S. Mazloom and A. M. E-Moghadam, *“Color image encryption based on coupled nonlinear chaotic map, "* Chaos, Solitons and Fractals 42 (2009) 1745–1754,
`www.elsevier.com/locate/chaos`

[22] *“ University of Southern California, signal and image processing institute"*
`http://sipi.usc.edu/database/`

[23] Y. Zhou, L. Bao, C. L. P. Chen *“A new 1D chaotic system for image encryption,"* Signal Processing 97 (2014) 172–182,
`www.elsevier.com/locate/sigpro`

[24] Yi Xun, P. Russell, B. Elisa, *“ Homomorphic encryption and applications "* 2014 XII, 126 p. 23 illus.,
`http://www.springer.com/978-3-319-12228-1`

[25] Z. H. Guan, f. Huang, and W. Guan , *“ Chaos-based image encryption algorithm , "* Physics Letters A 346 (2005) 153-157 ,
`www.sciencedirect.com ; www.elsevier.com/locate/pla`