Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2012, Article ID 173931, 13 pages
doi:10.1155/2012/173931

Research Article

A Secure Image Encryption Algorithm Based on Rubik's Cube Principle

Khaled Loukhaoukha, Jean-Yves Chouinard, and Abdellah Berdai

Department of Electrical and Computer Engineering, Laval University, QC, Canada G1K 7P4

Correspondence should be addressed to Khaled Loukhaoukha, khaled.loukhaoukha.1@ulaval.ca

Received 22 August 2011; Accepted 15 November 2011

Academic Editor: Fouad Khelifi

In the past few years, several encryption algorithms based on chaotic systems have been proposed as means to protect digital images against cryptographic attacks. These encryption algorithms typically use relatively small key spaces and thus offer limited security, especially if they are one-dimensional. In this paper, we proposed a novel image encryption algorithm based on Rubik's cube principle. The original image is scrambled using the principle of Rubik's cube. Then, XOR operator is applied to rows and columns of the scrambled image using two secret keys. Finally, the experimental results and security analysis show that the proposed image encryption scheme not only can achieve good encryption and perfect hiding ability but also can resist exhaustive attack, statistical attack, and differential attack.

1. Introduction

The end of the 20th century was marked by an extraordinary technical revolution from analog to numerical as documents and equipments became increasingly used in various domains. However, the advantages of the digital revolution were not achieved without drawbacks such as illegal copying and distribution of digital multimedia documents. To meet this challenge, researchers were motivated more than ever to protect multimedia documents with new and efficient document protection techniques. In this context, different techniques have been introduced such as encryption and digital watermarking. The first one consists in transforming multimedia documents using an algorithm to make it unreadable to anyone except for the legitimate users. The second one consists of embedding digital watermarks into multimedia documents to guarantee the ownership and the integrity of the digital multimedia contents.

The protection of images is of particular interest in this paper. Traditional image encryption algorithms such as private key encryption standards (DES and AES), public key standards such as Rivest Shamir Adleman (RSA), and the family of elliptic-curve-based encryption (ECC), as well as the international data encryption algorithm (IDEA), may not be the most desirable candidates for image encryption, especially for fast and real-time communication applications. In recent years, several encryption schemes have been proposed [1–12]. These encryption schemes can be classified into different categories such as value transformation [1–4], pixels position permutation [5–8], and chaotic systems [9–12].

In the first category, Liu et al. [1] presented an image encryption scheme based on iterative random phase encoding in gyrator transform domains. A two-dimensional chaotic mapping is employed to generate many random data for iterative random phase encoding. In [2], a color image encryption method using discrete fractional random transform (DFRNT) and the Arnold transform (AT) in the intensity-hue-saturation (IHS) color space has been proposed. Each color space component is then encrypted independently with different approaches. In [3], an image encryption algorithm based on Arnold transform and gyrator transform has been proposed. The amplitude and phase of the gyrator transform are separated into several subimages, which are scrambled using the Arnold transform. The parameters of gyrator transforms and separating scheme serve as the key of the encryption method. Tao et al. [4] proposed an image encryption algorithm based on fractional

Fourier transform (FRFT) and which can be applied to the double or more image encryptions. The encrypted image is obtained by the summation of different orders of inverse discrete fractional Fourier transform (IDFRFT) of the interpolated subimages. The whole transform orders of the utilized FRFT are used as the secret keys for the decryption of each subimage.

In the second category, Zunino [5] use Peano-Hilbert curves as pixels position permutation to destroy the spatial autocorrelation of an image. Zhang and Liu [6] proposed image encryption scheme based on permutation-diffusion architecture and skew tent map system. In the proposed scheme, the P-box is chosen as the same size of original image, which shuffles the positions of pixels totally. To enhance the security, the keystream in the diffusion step depends on both the key and original image. Zhao and Chen [7] proposed to used ergodic matrices for scrambling and encryption of digital images. The authors analyzed the isomorphism relationship between ergodic matrices and permutation. Zhu et al. [8] proposed an innovative permutation method to confuse and diffuse the gray-scale image at the bitlevel, which changes the position of the pixel and modifies its value. This algorithm uses also the Arnold cat map to permute the bits and the logistic map to further encrypt the permuted image.

In the third category, Huang and Nien [9] proposed a novel pixel shuffling method for color image encryption which used chaotic sequences generated by chaotic systems as encryption codes. In [10], the two-dimensional chaotic cat map has been generalized to three-dimensional one and then was used to design a fast and secure symmetric image encryption scheme. This scheme employs the 3D cat map to shuffle the positions and the values of image pixels. Wang et al. [11] presented an image encryption algorithm based on simple Perceptron and using a high-dimensional chaotic system in order to produce three sets of pseudorandom sequence. Then to generate weight of each neuron of perceptron as well as a set of input signal, a nonlinear strategy is adopted. Recently, a new image encryption algorithm combining permutation and diffusion was proposed by Wang et al. [12]. The original image is partitioned into blocks and a spatiotemporal chaotic system is then employed to generate the pseudorandom sequence used for diffusing and shuffling these blocks.

The security of image encryption has been extensively studied. Almost some encryption schemes based on permutation had already been found insecure against the ciphertext-only and known/chosen-plaintext attacks, due to the high information redundancy, and it is quite understandable since the secret permutations can be recovered by comparing the plaintexts and the permuted ciphertexts. Generally, chaos-based image encryption algorithms are used more often than others but require high computational cost. Moreover, a chaos system is defined on real numbers while the cryptosystems are defined on finite sets of integers. One-dimensional chaotic cryptosystems are limited by their small key spaces and weak security in [1, 13].

In this paper, we present a novel image encryption algorithm based on the principle of Rubik's cube. First-, in order to scramble the pixels of gray-scale original image the principle of Rubik's cube is deployed which only changes the position of the pixels. Using two random secret keys, the bitwise XOR is applied into the odd rows and columns. Then, the bitwise XOR is also applied to even rows and columns using the flipped secret keys. These steps can be repeated while the number of iteration is not reached. Numerical simulation has been performed to test the validity and the security of the proposed encryption algorithm.

The remaining of this paper is organized as follows. Section 2 describes the proposed image encryption algorithm based on Rubik's cube principle. Experimental results and security analysis are presented in Section 3. Finally, we conclude in Section 4.

2. Rubik's Cube Image Encryption

In this section, the proposed encryption algorithm based on Rubik's cube principle is described along with the decryption algorithm.

2.1. Rubik's Cube Based Encryption Algorithm. Let I_o represent an α-bit gray scale image of the size $M \times N$. Here, I_o represent the pixels values matrix of image I_o. The steps of encryption algorithm are as follows:

(1) Generate randomly two vectors K_R and K_C of length M and N, respectively. Element $K_R(i)$ and $K_C(j)$ Each take a random value of the set $\mathcal{A} = \{0, 1, 2, \ldots, 2^{\alpha} - 1\}$. Note that both K_R and K_C must not have constant values.

(2) Determine the number of iterations, $\text{ITER}_{\max}$, and initialize the counter ITER at 0.

(3) Increment the counter by one: ITER = ITER + 1.

(4) For each row i of image I_o,

(a) compute the sum of all elements in the row i, this sum is denoted by $\alpha(i)$

$$\alpha(i) = \sum_{j=1}^{N} \mathsf{I}_o(i, j), \quad i = 1, 2, \ldots, M, \tag{1}$$

(b) compute modulo 2 of $\alpha(i)$, denoted by $M_{\alpha(i)}$,

(c) row i is left, or right, circular-shifted by $K_R(i)$ positions (image pixels are moved $K_R(i)$ positions to the left or right direction, and the first pixel moves in last pixel.), according to the following:

$$\begin{aligned} &\text{if } M_{\alpha(i)} = 0 \longrightarrow \text{right circular shift} \\ &\qquad \text{else} \longrightarrow \text{left circular shift.} \end{aligned} \tag{2}$$

(5) For each column j of image I_o,

(a) compute the sum of all elements in the column j, this sum is denoted by $\beta(j)$,

$$\beta(j) = \sum_{i=1}^{M} \mathrm{I}_o(i, j), \quad j = 1, 2, \ldots, N, \tag{3}$$

(b) compute modulo 2 of $\beta(j)$, denoted by $M_{\beta(j)}$.

(c) column j is down, or up, circular-shifted by $K_C(i)$ positions, according to the following:

$$\begin{aligned} &\text{if } M_{\beta(j)} = 0 \longrightarrow \text{up circular shift} \\ &\quad \text{else} \longrightarrow \text{down circular shift.} \end{aligned} \tag{4}$$

Steps 4 and 5 above will create a scrambled image, denoted by I_{SCR}.

(6) Using vector K_C, the bitwise XOR operator is applied to each row of scrambled image I_{SCR} using the following expressions:

$$\begin{aligned} \mathrm{I}_1(2i-1, j) &= \mathrm{I}_{\mathrm{SCR}}(2i-1, j) \oplus K_C(j), \\ \mathrm{I}_1(2i, j) &= \mathrm{I}_{\mathrm{SCR}}(2i, j) \oplus \mathrm{rot}\,180(K_C(j)), \end{aligned} \tag{5}$$

where $\oplus$ and $\mathrm{rot}\,180(K_C)$ represent the bitwise XOR operator and the flipping of vector K_C from left to right, respectively.

(7) Using vector K_R, the bitwise XOR operator is applied to each column of image I_1 using the following formulas:

$$\begin{aligned} \mathrm{I}_{\mathrm{ENC}}(i, 2j-1) &= \mathrm{I}_1(i, 2j-1) \oplus K_R(j), \\ \mathrm{I}_{\mathrm{ENC}}(i, 2j) &= \mathrm{I}_1(i, 2j) \oplus \mathrm{rot}\,180(K_R(j)). \end{aligned} \tag{6}$$

with $\mathrm{rot}\,180(K_R)$ indicating the left to right flip of vector K_R.

(8) If $\mathrm{ITER} = \mathrm{ITER}_{\max}$, then encrypted image I_{ENC} is created and encryption process is done; otherwise, the algorithm branches to step 3.

Vectors K_R, K_C and the max iteration number $\mathrm{ITER}_{\max}$ are considered as secret keys in the proposed encryption algorithm. However, to obtain a fast encryption algorithm it is preferable to set $\mathrm{ITER}_{\max} = 1$ (single iteration). Conversely, if $\mathrm{ITER}_{\mathrm{MAX}} > 1$, then the algorithm is more secure because the key space is larger than for $\mathrm{ITER}_{\mathrm{MAX}} = 1$. Nevertheless, in the simulations presented in Section 3, the number of iterations $\mathrm{ITER}_{\max}$ was set to one.

2.2. Rubik's Cube Decryption Algorithm. The decrypted image, I_o, is recovered from the encrypted image, I_{ENC}, and the secret keys, K_R, K_C, and $\mathrm{ITER}_{\max}$ as follows in the following.

(1) Initialize ITER = 0.

(2) Increment the counter by one: ITER = ITER + 1.

(3) The bitwise XOR operation is applied on vector K_R and each column of the encrypted image I_{ENC} as follows:

$$\begin{aligned} \mathrm{I}_1(i, 2j-1) &= \mathrm{I}_{\mathrm{ENC}}(i, 2j-1) \oplus K_R(j), \\ \mathrm{I}_1(i, 2j) &= \mathrm{I}_{\mathrm{ENC}}(i, 2j) \oplus \mathrm{rot}\,180(K_R(j)), \end{aligned} \tag{7}$$

(4) Then, using the K_C vector, the bitwise XOR operator is applied to each row of image I_1:

$$\begin{aligned} \mathrm{I}_{\mathrm{SCR}}(2i-1, j) &= \mathrm{I}_1(2i-1, j) \oplus K_C(j), \\ \mathrm{I}_{\mathrm{SCR}}(2i, j) &= \mathrm{I}_1(2i, j) \oplus \mathrm{rot}\,180(K_C(j)). \end{aligned} \tag{8}$$

(5) For each column j of the scrambled image I_{SCR},

(a) compute the sum of all elements in that column j, denoted as $\beta_{\mathrm{SCR}}(j)$:

$$\beta_{\mathrm{SCR}}(j) = \sum_{i=1}^{M} \mathrm{I}_{\mathrm{SCR}}(i, j), \quad j = 1, 2, \ldots, N, \tag{9}$$

(b) compute modulo 2 of $\beta_{\mathrm{SCR}}(j)$, denoted by $M_{\beta_{\mathrm{SCR}}(j)}$,

(c) column j is down, or up, circular-shifted by $K_C(i)$ positions according to the following:

$$\begin{aligned} &\text{if } M_{\beta_{\mathrm{SCR}}(j)} = 0 \longrightarrow \text{up circular shift} \\ &\quad \text{else} \longrightarrow \text{down circular shift.} \end{aligned} \tag{10}$$

(6) For each row i of scrambled image I_{SCR},

(a) compute the sum of all elements in row i, this sum is denoted by $\alpha_{\mathrm{SCR}}(i)$:

$$\alpha_{\mathrm{SCR}}(i) = \sum_{j=1}^{N} \mathrm{I}_{\mathrm{SCR}}(i, j), \quad i = 1, 2, \ldots, M, \tag{11}$$

(b) compute modulo 2 of $\alpha_{\mathrm{SCR}}(j)$, denoted by $M_{\alpha_{\mathrm{SCR}}(j)}$,

(c) row i is then left, or right, circular-shifted by $K_R(i)$ according to the following:

$$\begin{aligned} &\text{if } M_{\alpha_{\mathrm{SCR}}(j)} = 0 \longrightarrow \text{right circular shift} \\ &\quad \text{else} \longrightarrow \text{left circular shift.} \end{aligned} \tag{12}$$

(7) If $\mathrm{ITER} = \mathrm{ITER}_{\max}$, then image I_{ENC} is decrypted and the decryption process is done; otherwise, the algorithm branches back to step 2.

Table 1: Difference measures between original and encrypted images.

Image	NPCR (in %)	UACI (in %)
Lena	99.5850	28.6210
Black	99.6078	50.1931
Baboon	99.6094	27.4092
Checkerboard	99.6201	50.0233

Table 2: Number of pixel change rate (NPCR) between images encrypted with keys K_1 and K_2 (with 1 bit difference).

Image	Mean (%)	Standard deviation (%)
Lena	99.6111	0.0251
Black	99.6064	0.0238
Baboon	99.6085	0.0253
Checkerboard	99.6113	0.0274

3. Experimental Results

In this section, we present the tests that were conducted to assess the efficiency and security of the proposed image encryption algorithm. These tests involve visual testing and security analysis.

3.1. Visual Testing. For visual testing, four gray-scale images of size 256 × 256 pixels were used. Figure 1 depicts these test images—`lena`, `black`, `baboon` and `checkerboard`—as well as the images encrypted using the proposed Rubik's cube algorithm. From this figure, one can see that there is no perceptual similarity between original images and their encrypted counterparts.

The encrypted image should greatly differ from its original form. In general, two difference measures are used to quantify this requirement. The first measure is the number of pixels change rate (NPCR), which indicate the percentage of different pixels between two images. The second one is the unified average changing intensity (UACI), which measures the average intensity of differences in pixels between two images [10]. Let $\mathsf{I}_o(i,j)$ and $\mathsf{I}_{\mathrm{ENC}}(i,j)$ be the pixels values of original and encrypted images, I_o and I_{ENC}, at the ith pixel row and jth pixel column, respectively. Equations (13) and (15) give the mathematical expressions of the NPCR and UACI measures:

$$\mathrm{NPCR} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\mathsf{D}(i,j)}{M \times N} \times 100\%, \tag{13}$$

$$\text{with} : \mathsf{D}(i,j) = \begin{cases} 0 & \text{if } \mathsf{I}_o(i,j) = \mathsf{I}_{\mathrm{ENC}}(i,j), \\ 1 & \text{otherwise.} \end{cases} \tag{14}$$

$$\mathrm{UACI} = \left[\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{\left|\mathsf{I}_o(i,j) - \mathsf{I}_{\mathrm{ENC}}(i,j)\right|}{255}\right] \times \frac{100\%}{M \times N}. \tag{15}$$

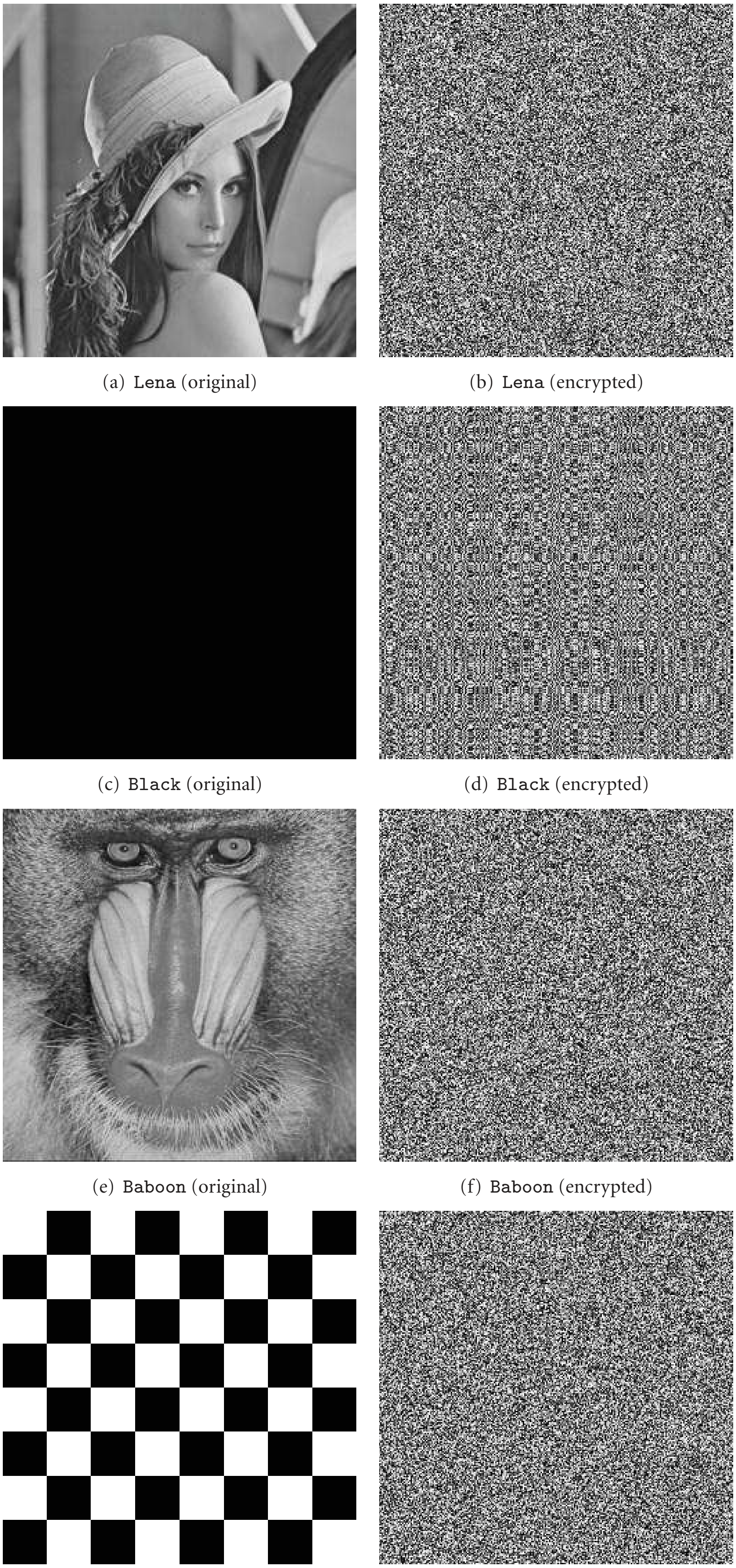
To approach the performances of an ideal image encryption algorithm, NPCR values must be as large as possible and UACI values must be around 33%. Table 1 gives the NPCR and UACI values for the original images and their encrypted versions. The values are very close to unity for the NPCR measure. The UACI values are also appropriate. The high percentage values of the NPCR measure indicate that the pixels positions have been randomly changed.

Furthermore, the UACI values show that almost all pixel gray-scale values of encrypted image have been changed from their values in the original images, making the original and encrypted image pixels more difficult to discriminate. It iss also observed that the UACI values for the case of the `black` and the `checkerboard` images are larger than those of the other images. This is due to the fact that all the pixels values of the `black` and `checkerboard` images are at the extremity of pixels values range, that is, 0 for the black pixels and 255 for the white pixels, leading to a large absolute difference between the original and encrypted images.

3.2. Security Analysis. Security is a major issue in cryptology. A good image encryption scheme should resist various attacks such as known plain text attack, cipher-text-only attack, statistical analysis attack, and brute-force attacks. In this section, a security analysis on the proposed image encryption algorithm is done. The security assessment has been done on key space analysis and statistical analysis.

3.2.1. Key Space Analysis. A secure image encryption scheme must have a large key space in order to make brute-force attack practically (computationally) infeasible. In theory, the proposed algorithm can accommodate an infinite key space. However, the encryption key used in our scheme is composed of the $(K_R, K_C, \mathrm{ITER}_{\max})$ triplet. For an α-bit gray-scale image I_o of size $M \times N$ pixels, the vectors K_R and K_C can take $2^{M\cdot\alpha}$ and $2^{N\cdot\alpha}$ possible values, respectively. If we consider that both vectors must not have constant values, and the key space size is $2^{\alpha\cdot(M+N)} \times \mathrm{ITER}_{\max} - 2^{2\alpha}$ keys, one can see that the size of the key space can be expanded when the number of iteration $\mathrm{ITER}_{\max}$ is increased. For instance, for an 8-bits, scale gray image of size 256 × 256 pixels and $\mathrm{ITER}_{\max} = 1$. The key space size is equal to $2^{4096} - 2^{16} \approx 10^{1233}$; this key space is large enough to resist exhaustive attack and it is larger than the key space size of the image encryption algorithms proposed in [1, 9, 10, 14–16].

3.2.2. Key Sensibility. Encryption algorithms should also have high sensibility to encryption key: this means that any small change in the key should lead to a significant change in the encrypted, or decrypted, image. We performed two tests to illustrate the key sensibility of our scheme. The first one shows the impact of a key change in the image encryption process. Here, the original image, I_o, is encrypted using the key $K_1 = (K_R, K_C, \mathrm{ITER}_{\max})$, where K_R, K_C, and $\mathrm{ITER}_{\mathrm{MAX}}$ are randomly generated. Then, the same image, that is, I_o, is encrypted using another key K_2 which differs only from the first key, K_1, in the least significant bit, that is, $K_2 = (\mathrm{K}_R, K_C, \mathrm{ITER}_{\max} + 1)$. This experiment is repeated 100 times using different key pairs K_1 and K_2 (still only differing by the least significant bit). Table 2 represents the mean and the standard deviation of the NPCR values between the



(a) Lena (original)

(b) Lena (encrypted)

(c) Black (original)

(d) Black (encrypted)

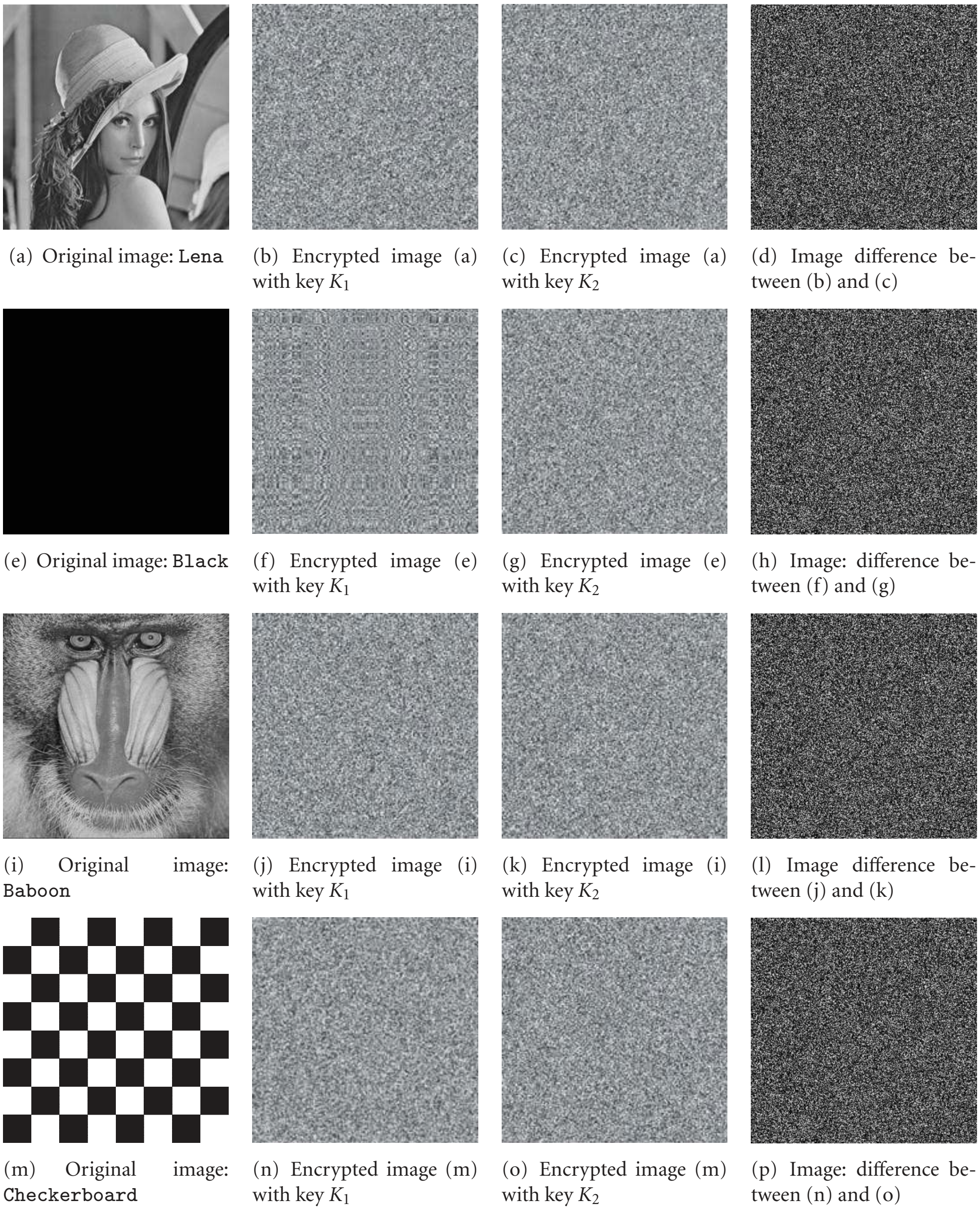
(e) Baboon (original)

(f) Baboon (encrypted)

(g) Checkerboard (original)

(h) Checkerboard (encrypted)

Figure 1: Original and encrypted images.


(a) Original image: `Lena`

(b) Encrypted image (a) with key K_1

(c) Encrypted image (a) with key K_2

(d) Image difference between (b) and (c)

(e) Original image: `Black`

(f) Encrypted image (e) with key K_1

(g) Encrypted image (e) with key K_2

(h) Image: difference between (f) and (g)

(i) Original image: `Baboon`

(j) Encrypted image (i) with key K_1

(k) Encrypted image (i) with key K_2

(l) Image difference between (j) and (k)

(m) Original image: `Checkerboard`

(n) Encrypted image (m) with key K_1

(o) Encrypted image (m) with key K_2

(p) Image: difference between (n) and (o)

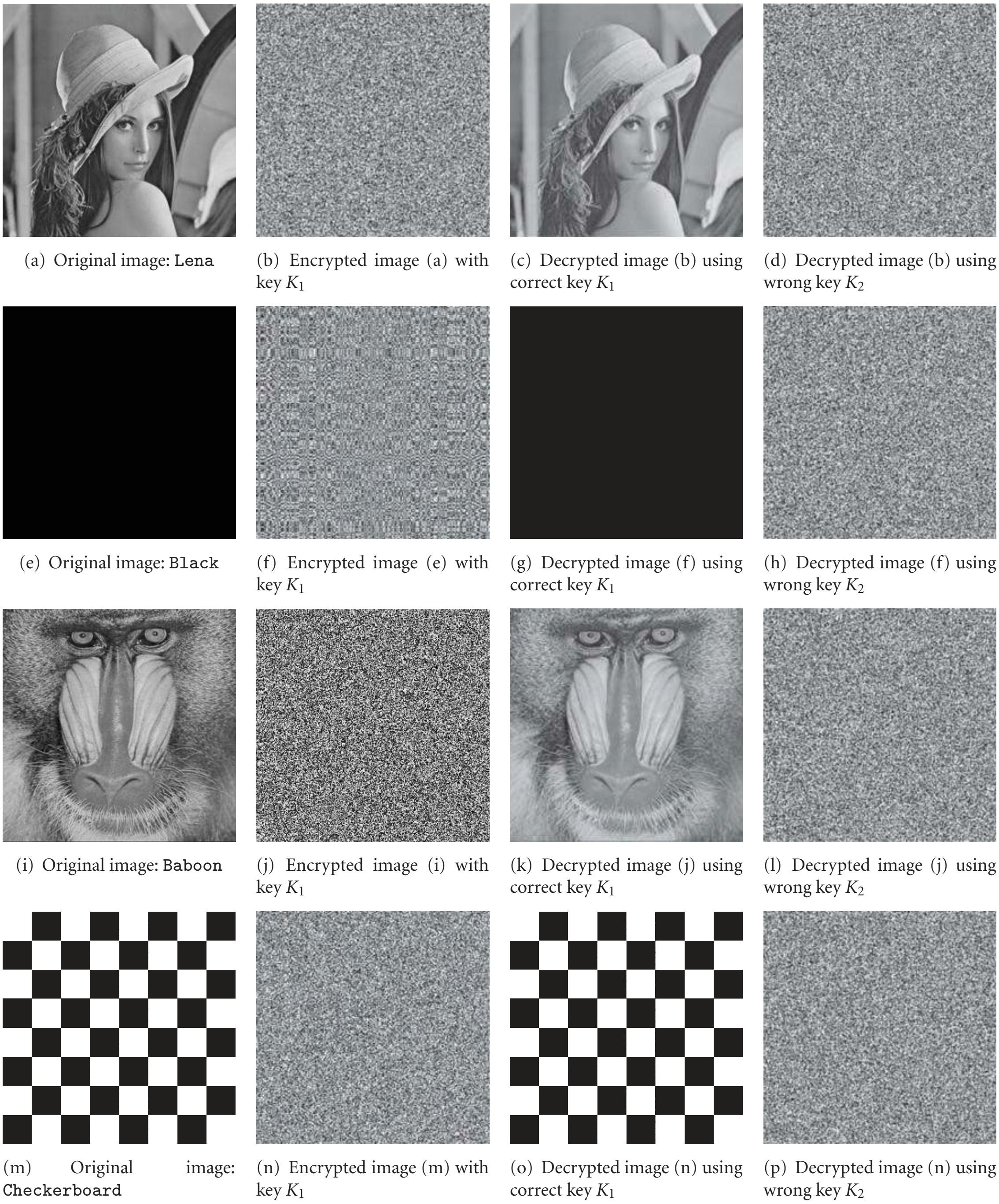
Figure 2: Key sensibility for image encryption using the proposed algorithm.

encrypted image with key K_1 and the encrypted image using the key K_2 using 100 different key pairs. One can see from Table 2 that the mean values of NPCR are close to 100%, which means that image encrypted by the key K_1 differs significantly from the one encrypted by key K_2. Moreover, standard deviation values are very small: this indicates that the NPCR values are clustered closely around the mean.

Figure 2 represents the original images, their encrypted images using two different keys K_1 and K_2 and the image difference between the encrypted images, respectively. As mentioned earlier, keys K_1 and K_2 differ only by one bit.

The second test consists of measuring the key sensibility in the image decryption process. Let original image I_o be encrypted using the key $K_1 = (K_R, K_C, \mathrm{ITER}_{\mathrm{MAX}})$, where K_R, K_C, and $\mathrm{ITER}_{\mathrm{MAX}}$ are again randomly generated, to give the encrypted image I_{ENC}. This image is decrypted separately using the keys K_1 and K_2; these keys always differ by only one bit in the least significant bit location. Figure 3 illustrates the original image, the encrypted image I_{ENC} with key K_1, the decrypted image of I_{ENC} using correct key K_1, and the decrypted image of I_{ENC} using the wrong key K_2. It is clear from this figure that decryption using a wrong key does not succeed.

3.3. Statistical Analysis. In a paper published in 1949 [17], Shannon stated that "*It is possible to solve many kinds of ciphers by statistical analysis.*" Consequently, he suggested two methods based on confusion and diffusion in order to counteract powerful attacks based on statistical analysis. In



(a) Original image: `Lena`

(b) Encrypted image (a) with key K_1

(c) Decrypted image (b) using correct key K_1

(d) Decrypted image (b) using wrong key K_2

(e) Original image: `Black`

(f) Encrypted image (e) with key K_1

(g) Decrypted image (f) using correct key K_1

(h) Decrypted image (f) using wrong key K_2

(i) Original image: `Baboon`

(j) Encrypted image (i) with key K_1

(k) Decrypted image (j) using correct key K_1

(l) Decrypted image (j) using wrong key K_2

(m) Original image: `Checkerboard`

(n) Encrypted image (m) with key K_1

(o) Decrypted image (n) using correct key K_1

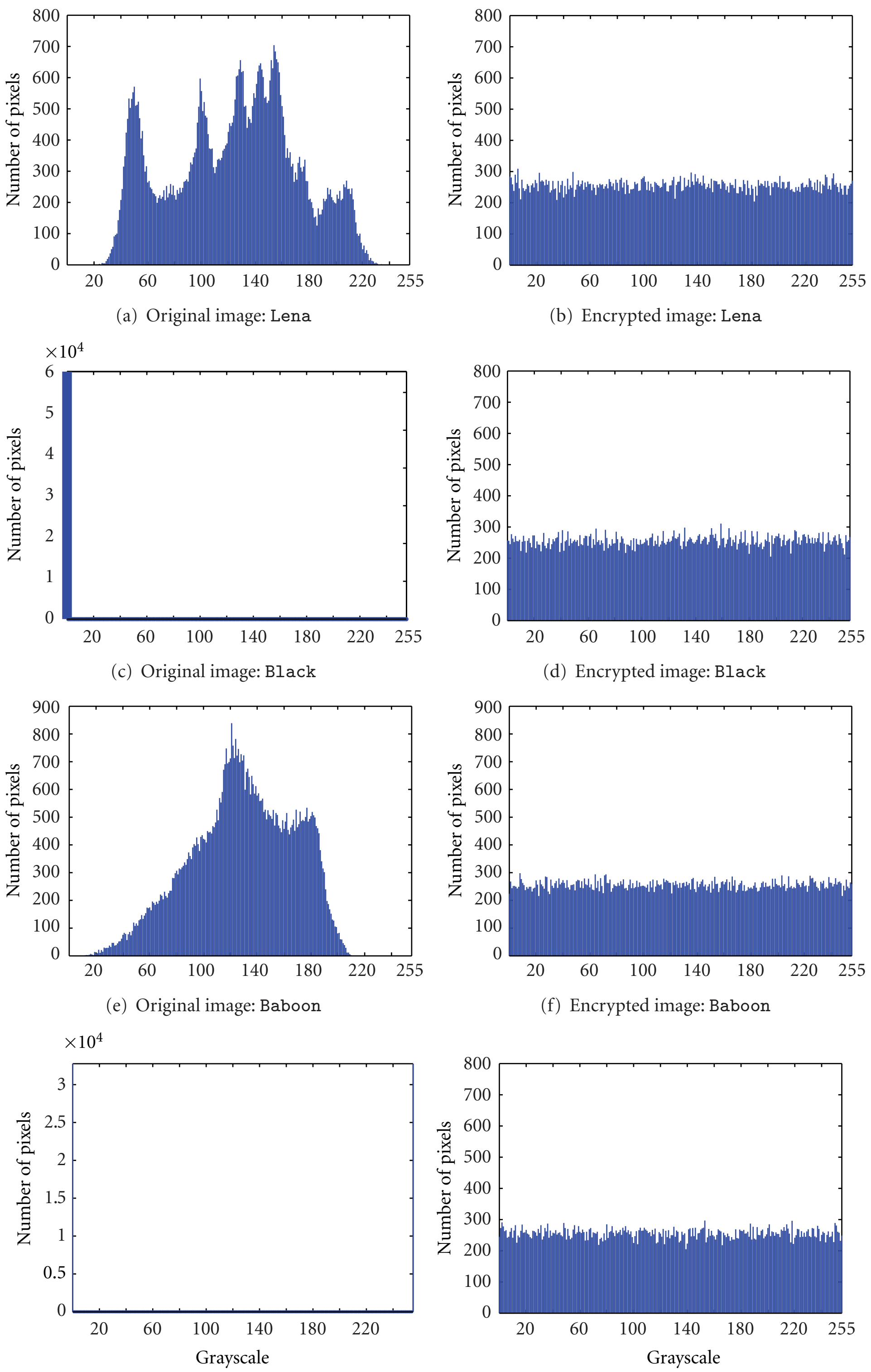
(p) Decrypted image (n) using wrong key K_2

Figure 3: Key sensibility for image decryption using the proposed algorithm.

the present paper, statistical analysis has been performed to demonstrate the superior confusion and diffusion properties of the proposed encryption algorithm against statistical attacks. This is done by performing two series of tests: histograms analysis of the encrypted images and the correlations computation of the adjacent pixels in encrypted images.

Figure 4 represents the histograms of the original and the encrypted images illustrated previously in Figure 1. One can see those the histograms of the encrypted images are almost uniform and are significantly different from that of the four original images. For instance, the histogram of original image `Checkerboard` shows as expected only two values: 0 and 255; however, the histogram of the encrypted `Checkerboard` image is fairly uniform. Therefore, the proposed image encryption algorithm responds well to the diffusion properties: it does not provide information that can be exploited for attacks based on statistical analysis of the encrypted image.

The other statistical test consists of computing the correlation between adjacent pixels [10]. It is obvious



(a) Original image: Lena

(b) Encrypted image: Lena

(c) Original image: Black

(d) Encrypted image: Black

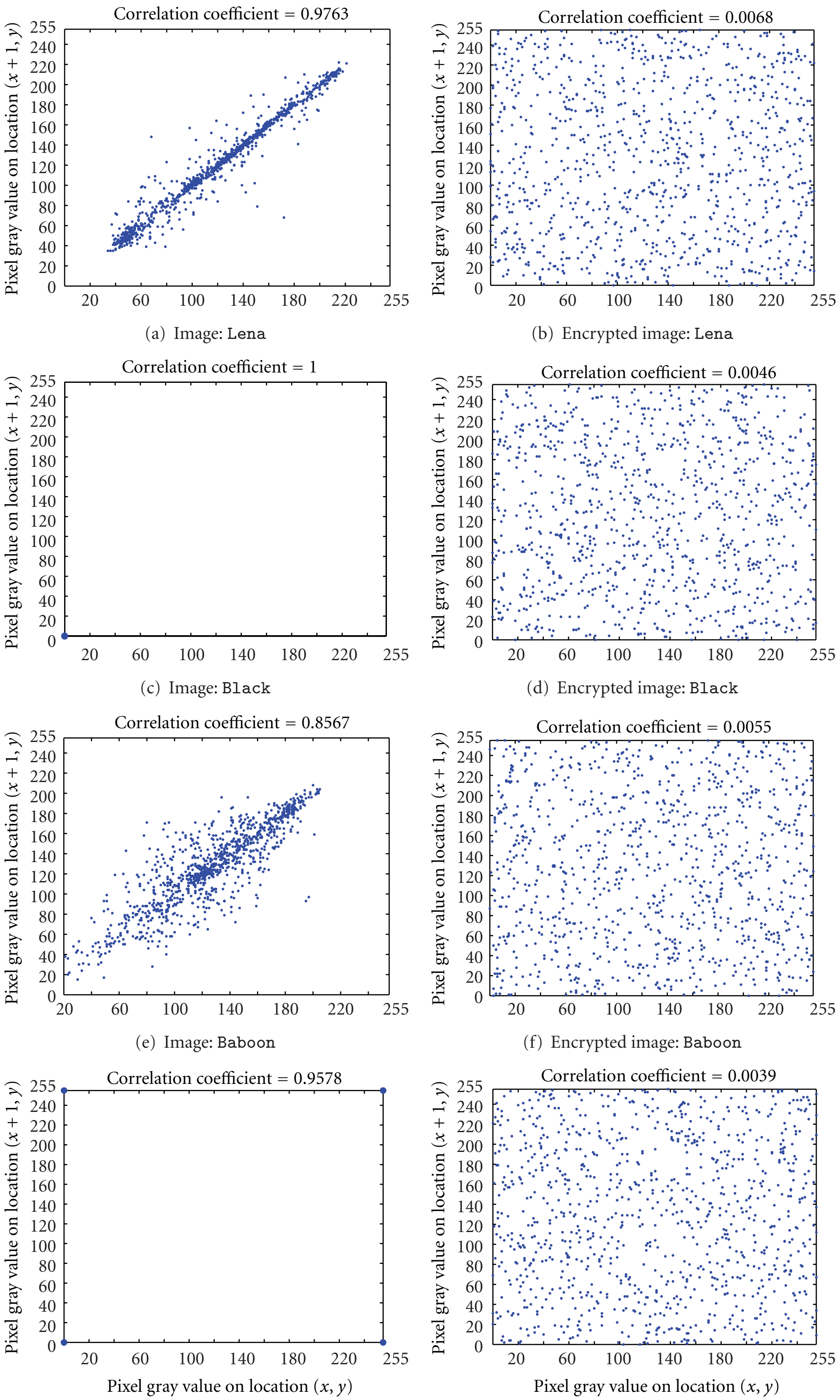
(e) Original image: Baboon

(f) Encrypted image: Baboon

(g) Original image: Checkerboard

(h) Encrypted image: Checkerboard

Figure 4: Histograms of original and encrypted images.



(a) Image: `Lena`

(b) Encrypted image: `Lena`

(c) Image: `Black`

(d) Encrypted image: `Black`

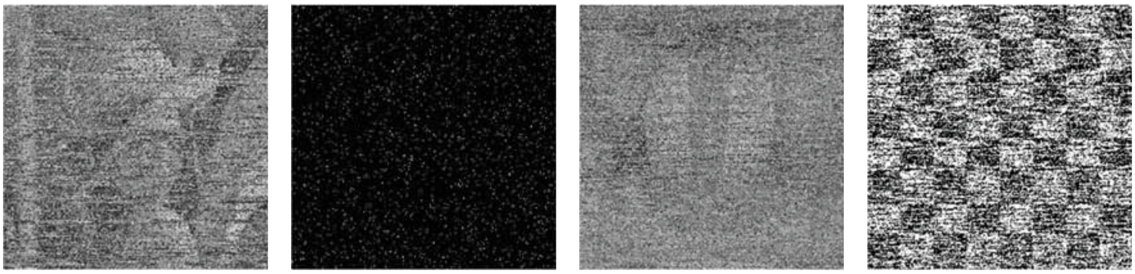
(e) Image: `Baboon`

(f) Encrypted image: `Baboon`

(g) Image: `Checkerboard`

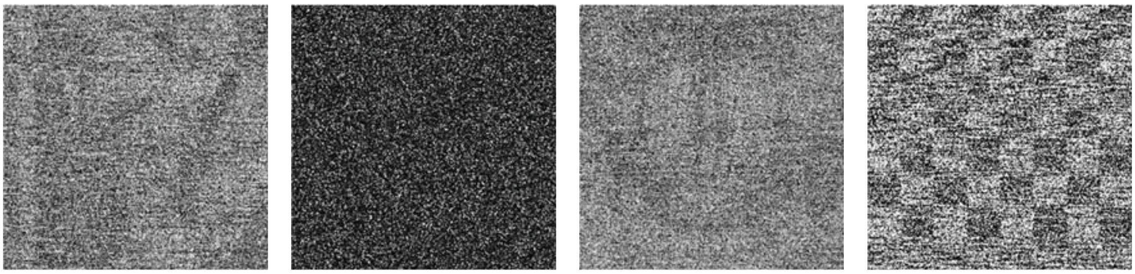
(h) Encrypted image: `Checkerboard`

Figure 5: Correlation distribution of the pairs horizontal to adjacent pixels.



(a) Decrypted image: `Lena` (b) Decrypted image: `Black` (c) Decrypted image: `Baboon` (d) Decrypted image: `Checkerboard`

Figure 6: Attack by salt & pepper noise.



(a) Decrypted image: `Lena` (b) Decrypted image: `Black` (c) Decrypted image: `Baboon` (d) Decrypted image: `Checkerboard`

Figure 7: Attack by speckle noise.

that an arbitrarily chosen pixel in an image is generally strongly correlated with adjacent pixels, and its in either horizontal, vertical or diagonal directions. However, a secure image encryption algorithm must produce an encrypted image having low correlation between adjacent pixels. This correlation test consists of randomly selecting N pairs of adjacent pixels (vertical, horizontal, and diagonal) from the original and the encrypted images separately. Then, the correlation coefficient of each pair is calculated using (19)

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{16}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))^2, \tag{17}$$

$$\operatorname{cov}(x, y) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))(y_i - E(y)), \tag{18}$$

$$\gamma_{xy} = \frac{\operatorname{cov}(x, y)}{\sqrt{D(x)}\sqrt{D(y)}} \quad \text{with } D(x) \neq 0,\ D(y) \neq 0, \tag{19}$$

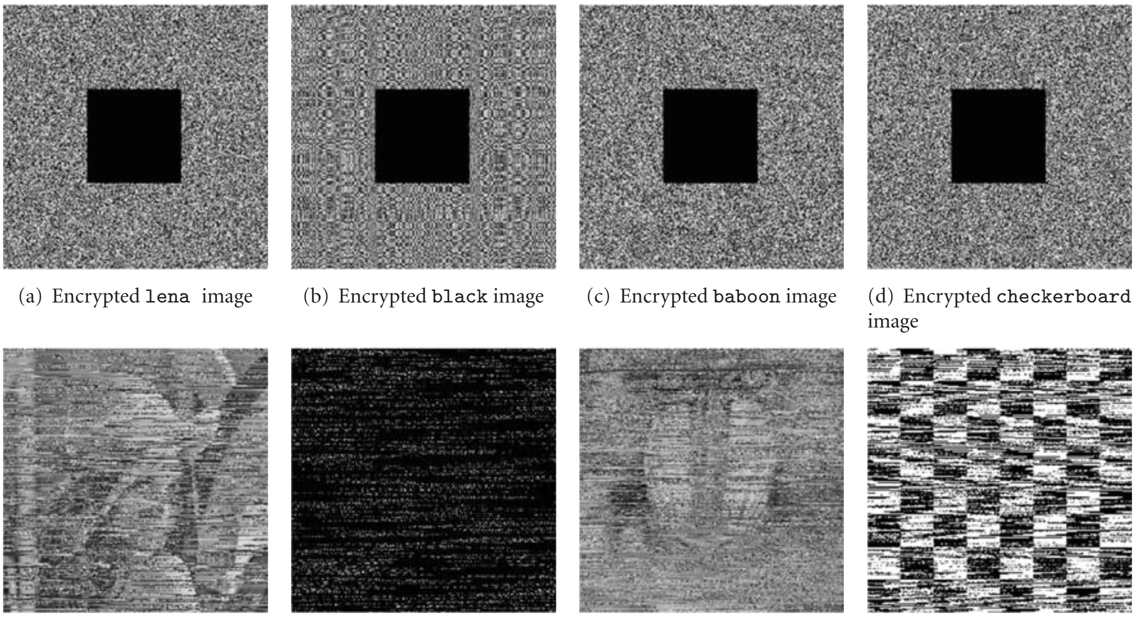
where x_i and y_i are the grayscale values of two adjacent pixels, N is the number of pairs (x_i, y_i), and $E(x)$ and $E(y)$, are respectively, the mean values of x_i and y_i.

Table 3 gives the correlation coefficient values of adjacent pixels in the horizontal, vertical, and diagonal directions of the original images and their encrypted versions. It is clear that for the original images, the coefficient correlation values are very high (close to one) contrary to those observed for the encrypted images. This confirms that adjacent pixels in the original images are strongly correlated. However, for the encrypted images, those values are close to zero, which means that the adjacent pixels (horizontal, vertical and diagonal directions) are very weakly correlated. Figure 5 illustrates the correlation distributions of the horizontal adjacent pixels of the original images and the corresponding encrypted images using the proposed algorithm. One can see from Figure 5 that adjacent pixels in encrypted images are indeed very weakly correlated.

Table 3: Correlation coefficients between adjacent pairs of pixels for original and encrypted images.

Correlation	Horizontal	Vertical	Diagonal
Original image: `lena`	0.9763	0.9453	0.9282
Encrypted image: `lena`	0.0068	0.0091	0.0063
Original image: `black`	1.0000	1.0000	1.0000
Encrypted image: `black`	0.0046	0.0055	0.0056
Original image: baboon	0.8567	0.8772	0.7880
Encrypted image: `baboon`	0.0055	0.0078	0.0042
Original image: `checkerboard`	0.9578	0.9521	0.9118
Encrypted image: `checkerboard`	0.0039	0.0090	0.0045

3.4. Entropy Analysis. The concept of entropy analysis for image encryption algorithm was introduced by Edward [18].



(a) Encrypted lena image (b) Encrypted black image (c) Encrypted baboon image (d) Encrypted checkerboard image

(e) Image (a) decrypted (f) Image (b) decrypted (g) Image (c) decrypted (h) Image (d) decrypted

Figure 8: Encrypted images under cropping attack in center (cropping over 1/8 of the entire image area).

Table 4: Comparison of entropy values of original images and their encrypted version.

Image	Original	Encrypted
Lena	7.4318 Sh	7.9968 Sh
Black	0 Sh	7.9966 Sh
Baboon	7.2279 Sh	7.9974 Sh
Checkerboard	1 Sh	7.9972 Sh

Table 5: Comparison of entropy values of encrypted images under different image encryption algorithms.

Algorithm	Entropy (Sh)
Proposed algorithm	7.9968
Baptista [19]	7.9260
Wong et al. [20]	7.9690
Xiang et al. [21]	7.9950
Lin and Wang [22]	7.9890

For gray-scale images of 256 levels, if each level of gray is assumed to be equiprobable, then the entropy of this image will be theoretically equal to 8 Sh (or bits). Ideally, an algorithm for encryption of images should give an encrypted image having equiprobable gray levels. Table 4 gives the entropy values of the four original images and those of their encrypted versions.

From those entropy values, we note that the entropy values of original images are far from ideal value of entropy since information sources are highly redundant and thus rarely generate uniformly distributed random messages. On the other hand, the entropy values of the encrypted images are very close to the ideal value of 8 Sh, which means that the proposed encryption algorithm is highly robust against entropy attacks.

Table 5 gives a comparison of the entropy values for encrypted image Lena with various image encryption algorithms.

3.5. Analysis against Attacks. An attacker who intercepts encrypted image can easily modify it, while the legitimate user can receive it and decrypt it successfully. This is the principle of attacks in image encryption; these attacks can include additive noise, filtering, rotation and cropping, and so forth.

3.5.1. Additive Noise. To verify the performance of the proposed encrypted algorithm against additive noise attacks, we considered two types of noises: salt and pepper noise and speckle noise. An additive noise attack consists in adding random noise to the intercepted encrypted image. Then, the noisy encrypted image will be decrypted. To measure the robustness of the proposed image encryption algorithm against this attack, mean squared error (MSE) measures are used.

Table 6 gives the MSE values between original images and their decrypted ones under the salt and pepper noise with different noise density values and the speckle noise with different variances.

Figures 6 and 7 illustrate the decrypted images: their encrypted version has been attacked separately by salt & pepper noise with 0.05 density and by speckle noise of variance 0.05. From these results, we can conclude that random noise attacks seriously affect decrypted images.

Table 6: MSE between original images and the decrypted versions under different noise attacks.

Image	Salt and pepper noise		Speckle noise	
	Density	MSE	Variance	MSE
Lena	0.05	3.42×10^3	0.05	5.30×10^3
	0.1	3.68×10^3	0.1	5.79×10^3
Black	0.05	1.06×10^3	0.05	8.11×10^3
	0.1	2.20×10^3	0.1	1.02×10^4
Baboon	0.05	2.38×10^3	0.05	3.97×10^3
	0.1	2.49×10^3	0.1	4.50×10^3
Checkerboard	0.05	2.44×10^4	0.05	2.12×10^4
	0.1	2.50×10^4	0.1	2.19×10^4

Table 7: MSE between original images and the decrypted versions under cropping attack.

Image	Center cropping	Sides cropping
Lena	2.96×10^3	2.90×10^3
Black	2.72×10^3	2.69×10^3
Baboon	2.05×10^3	2.02×10^3
Checkerboard	1.81×10^4	1.64×10^4

Table 8: Speed test results of the proposed algorithm with image Lena using a 2.7 GHz personal computer.

Image size	Encryption time	Decryption time
64 × 64	0.03 s	0.03 s
128 × 128	0.04 s	0.04 s
256 × 256	0.12 s	0.12 s
512 × 512	0.66 s	0.66 s
1024 × 1024	5.40 s	5.40 s

3.5.2. Analysis against Cropping Attacks. The cropping attacks consist of modifying the intercepted encrypted image by deleting one or several areas of the image. Table 7 gives the MSE values between original images and the decrypted and cropped images either in their center or on the image sides with parameter values equal to 1/8. This one indicates the fraction of the encrypted image that has been cropped. Figure 8 represents the encrypted images cropped in the center and their decrypted versions. From these results, we can conclude that the proposed image encryption algorithm resists to this attack lightly.

3.6. Speed Test. Apart from security considerations, another important consideration in the design of image encryption techniques is the actual algorithm execution speed, particularly for real-time applications. The proposed encryption algorithm is indeed very fast compared to other algorithms [10]. Our experimental results show that the average speed for encryption and for decryption is of around 0.9 Mb/s (megabits per second). The peak speed can reach up to 2.2 Mb/s on personnel computer equipped with an AMD Athlon processor with clock speed of 2.70 GHz, 1 GB (gigabytes) of RAM memory and 160 GB hard-disk capacity.

Table 8 illustrates the performance of the proposed algorithm using original image Lena with different sizes ranging from 64 × 64 to 1024 × 1024 pixels. The proposed algorithm was written using the MATLAB software platform.

4. Conclusion

In this paper, a novel image encryption algorithm is proposed. This algorithm is based on the principle of Rubik's cube to permute image pixels. To confuse the relationship between original and encrypted images, the XOR operator is applied to odd rows and columns of image using a key. The same key is flipped and applied to even rows and columns of image. Experimental tests have been carried out with detailed numerical analysis which demonstrates the robustness of the proposed algorithm against several types of attacks such as statistical and differential attacks (visual testing). Moreover, performance assessment tests demonstrate that the proposed image encryption algorithm is highly secure. It is also capable of fast encryption/decryption which is suitable for real-time Internet encryption and transmission applications.

References

[1] Z. Liu, L. Xu, C. Lin, J. Dai, and S. Liu, "Image encryption scheme by using iterative random phase encoding in gyrator transform domains," *Optics and Lasers in Engineering*, vol. 49, no. 4, pp. 542–546, 2011.

[2] Q. Guo, Z. Liu, and S. Liu, "Color image encryption by using Arnold and discrete fractional random transforms in IHS space," *Optics and Lasers in Engineering*, vol. 48, no. 12, pp. 1174–1181, 2010.

[3] Z. Liu, H. Chen, T. Liu et al., "Image encryption by using gyrator transform and Arnold transform," *Journal of Electronic Imaging*, vol. 2, no. 4, pp. 345–351, 1993.

[4] R. Tao, X. Y. Meng, and Y. Wang, "Image encryption with multiorders of fractional fourier transforms," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 734–738, 2010.

[5] R. Zunino, "Fractal circuit layout for spatial decorrelation of images," *Electronics Letters*, vol. 34, no. 20, pp. 1929–1930, 1998.

[6] G. Zhang and Q. Liu, "A novel image encryption method based on total shuffling scheme," *Optics Communications*, vol. 284, no. 12, pp. 2775–2780, 2011.

[7] X.-Y. Zhao and G. Chen, "Ergodic matrix in image encryption," in *Proceedings of the 2nd International Conference on Image and Graphics*, vol. 4875, pp. 394–401, August 2002.

[8] Z.-L. Zhu, W. Zhang, K.-W. Wong, and H. Yu, "A chaos-based symmetric image encryption scheme using a bit-level permutation," *Information Sciences*, vol. 181, no. 6, pp. 1171–1186, 2011.

[9] C. K. Huang and H. H. Nien, "Multi chaotic systems based pixel shuffle for image encryption," *Optics Communications*, vol. 282, no. 11, pp. 2123–2127, 2009.

[10] G. Chen, Y. Mao, and C. K. Chui, "A symmetric image encryption scheme based on 3D chaotic cat maps," *Chaos, Solitons and Fractals*, vol. 21, no. 3, pp. 749–761, 2004.

[11] X. Y. Wang, L. Yang, R. Liu, and A. Kadir, "A chaotic image encryption algorithm based on perceptron model," *Nonlinear Dynamics*, vol. 62, no. 3, pp. 615–621, 2010.

[12] Y. Wang, K. W. Wong, X. Liao, and G. Chen, "A new chaos-based fast image encryption algorithm," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 514–522, 2011.

[13] S. Li, C. Li, G. Chen, N. G. Bourbakis, and K. T. Lo, "A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks," *Signal Processing: Image Communication*, vol. 23, no. 3, pp. 212–223, 2008.

[14] C. K. Huang, H. H. Nien, S. K. Changchien, and H. W. Shieh, "Image encryption with chaotic random codes by grey relational grade and Taguchi method," *Optics Communications*, vol. 280, no. 2, pp. 300–310, 2007.

[15] S. Mazloom and A. M. Eftekhari-Moghadam, "Color image encryption based on Coupled Nonlinear Chaotic Map," *Chaos, Solitons and Fractals*, vol. 42, no. 3, pp. 1745–1754, 2009.

[16] Y. Tang, Z. Wang, and J. A. Fang, "Image encryption using chaotic coupled map lattices with time-varying delays," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 9, pp. 2456–2468, 2010.

[17] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[18] O. Edward, *Chaos in Dynamical Systems*, Cambridge University Press, Cambridge, UK, 2nd edition, 2003.

[19] M. S. Baptista, "Cryptography with chaos," *Physics Letters, Section A*, vol. 240, no. 1-2, pp. 50–54, 1998.

[20] K. W. Wong, S. W. Ho, and C. K. Yung, "A chaotic cryptography scheme for generating short ciphertext," *Physics Letters, Section A*, vol. 310, no. 1, pp. 67–73, 2003.

[21] T. Xiang, X. Liao, G. Tang, Y. Chen, and K. W. Wong, "A novel block cryptosystem based on iterating a chaotic map," *Physics Letters, Section A*, vol. 349, no. 1–4, pp. 109–115, 2006.

[22] Z. Lin and H. Wang, "Efficient image encryption using a chaos-based PWL memristor," *IETE Technical Review*, vol. 27, no. 4, pp. 318–325, 2010.