Article

Efficient Entropic Security with Joint Compression and Encryption Approach Based on Compressed Sensing with Multiple Chaotic Systems

Jingya Wang 1, Xianhua Song 1,∗ and Ahmed A. Abd El-Latif 2,3,∗

1 School of Science, Harbin University of Science and Technology, Harbin 150080, China; 2029000175@stu.hrbust.edu.cn
2 EIAS Data Science Lab, College of Computer and Information Sciences, Prince Sultan University, Riyadh 11586, Saudi Arabia
3 Department of Mathematics and Computer Science, Faculty of Science, Menoufia University, Shebin El-Koom 32511, Egypt
∗ Correspondence: songxianhua@hrbust.edu.cn (X.S.); aabdellatif@psu.edu.sa (A.A.A.E.-L.)

Abstract: This paper puts forward a new algorithm that utilizes compressed sensing and two chaotic systems to complete image compression and encryption concurrently. First, the hash function was utilized to obtain the initial parameters of two chaotic maps, which were the 2D-SLIM and 2D-SCLMS maps, respectively. Second, a sparse coefficient matrix was transformed from the plain image through discrete wavelet transform. In addition, one of the chaotic sequences created by 2D-SCLMS system performed pixel transformation on the sparse coefficient matrix. The other chaotic sequences created by 2D-SLIM were utilized to generate a measurement matrix and perform compressed sensing operations. Subsequently, the matrix rotation was combined with row scrambling and column scrambling, respectively. Finally, the bit-cycle operation and the matrix double XOR were implemented to acquire the ciphertext image. Simulation experiment analysis showed that the compressed encryption scheme has advantages in compression performance, key space, and sensitivity, and is resistant to statistical attacks, violent attacks, and noise attacks.

Keywords: image encryption; compressed sensing; chaotic system; bit-cycle operation; double XOR operation

1. Introduction

In the wake of the development of the internetworking and information technique, digital images are extensively used in numerous domains [1–4]. A great quantity of information is presented in a digital image form, which usually contains private and important information. When important information is falsified or leaked, it can cause acute consequences [5,6], which makes the privacy security issue very prominent. Hence, the information security protection of digital images has aroused widespread attention [7,8]. In this situation, multiple encryption scenarios have emerged.

In the past few years, due to the excellent characteristics of chaotic maps [9,10], various encryption scenarios based on chaotic maps have been created [11–14]. Wang et al. utilized parameter controlled scroll chaotic attractors for encryption [15]. Gao proposed a new 2D hyperchaotic system for image encryption [16]. In addition, chaotic maps can be combined with a variety of methods for encryption. Chen et al. combined chaotic maps and DNA coding for encryption and the results indicated that the effect was better than using chaotic maps alone [17,18]. Yu et al. combined chaotic maps and fractional Fourier transform for optical image encryption [19,20]. Choi et al. combined chaotic maps and cellular automata for encryption [21,22]. Sundarakrishnan et al. used chaotic mapping and
cellular automata to encrypt color images, increased the key space, and used a double permutation and replacement framework, which significantly reduced the correlation and improved the security of the algorithm [23]. Based on the advantage of chaos theory to encryption, many scholars began to use multiple chaotic systems in the encryption framework. Ramasamy et al. achieved secure and efficient encryption using the proposed enhanced logical map, chaotic map, and general encryption framework—scrambling, diffusing, and generating a key stream [24]. Masood et al. used multiple chaotic systems such as two-dimensional Arnold cat mapping, Newton–Leipnik dynamic system and improved Logistic–Gaussian chaotic system, to generate sequences for multiple links of color image encryption, which improved the security of the algorithm [25]. This image encryption scheme using multiple chaotic systems combined with the encryption framework makes full use of the advantages of chaos for encryption, making encryption more secure and efficient, obtaining good encryption effects under various experimental tests, and resisting various attacks. Although the above-mentioned algorithms have achieved good results, none of the above algorithms are applicable due to the bandwidth constraint problem.

To satisfy the bandwidth-constrained demands, the theoretical concept of compressed sensing (CS) was established [26,27]. Soon afterward, multifarious compressed encryption scenarios based on CS were put forward [28,29]. Lu et al., created an image encryption scenario [30] that compressed images by CS and encrypted images by double random phase coding technology. Although this algorithm reduced the amount of data, its method of using the metric matrix as the key takes up a large amount of storage space and is bandwidth-constrained.

To resolve these issues, a new image compression encryption scenario has attracted much attention [31–36]. This scenario combines compressed sensing with chaotic systems, which utilizes compressed sensing to compress the image to meet the bandwidth demands in transmission and can also make full use of the excellent properties of the chaotic system by using the initial parameters of chaotic maps as the key, and using the created sequences to form the measurement matrix. This method resolves the problem that the key occupies a large storage space and is bandwidth-constrained.

To further improve the security of these schemes, many scenarios have adopted scrambling methods [37–41]. According to the position of the scrambling in the algorithm, these can be divided into two categories. One is to perform the scrambling and confusion operation after the measurement value is obtained by compressed sensing [40,41]. The other is to first obtain a sparse coefficient matrix through a sparse transformation of the original image and then perform a scrambling operation on the sparse coefficient matrix [38,39]. Both types of methods can decrease the image correlation and heighten the security of the scenarios, while the latter has the advantage of effectively enhancing the reconstruction quality of the decrypted image [42]. In general, there are two scrambling methods in the encryption process: scrambling using Arnold map [38] and scrambling of the index values obtained by sorting the chaotic sequences [37,38]. Both methods have drawbacks. The Arnold map scrambling method cannot be directly used for non-square images [43]. The second scrambling method is easy to operate and its scrambling effect is determined by the randomness of the chaotic sequence [44], so it is not suitable for a chaotic system with bad randomness. Therefore, a scrambling method called pixel transformation is proposed.

To increase the security and meet the demand of limited bandwidth, a compressed encryption plan based on two chaotic systems and CS was put forward in this paper. First, the SHA-384 of the original image was used to calculate the initial parameters of the 2D-SLIM and 2D-5CLMS system and used as the key, which greatly heightened the relevance between the scheme and the plaintext, and can better resist known plaintext attacks and selective plaintext attacks. Second, the plaintext image is converted into a sparse coefficient matrix. Third, to increase the reconstruction quality of the decrypted image, a new scrambling technology is created. In addition, the chaotic sequence is utilized to create the
measurement matrix and implement the compressed sensing operation, which greatly meets the transmission bandwidth requirements. To further heighten the security, the encryption operation combines matrix rotation with row scrambling and column scrambling, respectively, followed by a bit-cycle operation. Finally, double XOR of the matrix is implemented to acquire the ciphertext image.

The novelties of this paper are: (1) By combining two chaotic systems and compressed sensing, a new image encryption scheme is generated; (2) a new pixel transformation scrambling method is proposed; and (3) the combination of matrix rotation and scrambling improves the security of the algorithm.

The remaining sections are organized as follows. Section 2 presents the related work. Section 3 designs the new compression encryption scenario. Section 4 demonstrates the corresponding decryption algorithms. Section 5 presents the various performance analyses of the compression encryption scenario. Section 6 provides our conclusions.

2. Related Works

2.1. Compressed Sensing

In 2006, Donoho et al. proposed a compressed sensing formulation and processing method for signals [26,27]. This concept smashes the restrictions of Shannon’s sampling theorem by exploiting the sparsity of the natural signal itself or the sparsity of a certain transform domain, allowing for the recovery of the sampled signal with a small amount of samples at lower than the Nyquist sampling rate. Compressed sensing, also known as compressive sampling, allows for sampling, compression, and encryption to be conducted concurrently [43,45].

The pivotal elements of compressed sensing comprises sparse representation, the measurement matrix, and the reconstruction scheme. In general, the signal is not sparse in the time domain, but in some transform domains, the signal may become sparse. Therefore, the classic sparsity representation methods comprise discrete wavelet transform (DWT), fast Fourier transform (FFT), and discrete cosine transform (DCT).

We took a 1D signal to explain the step of compressed sensing. The sparsity expression for a non-sparse signal \( x \) (\( N \times 1 \)) in the transform domain is

\[
x = \Psi s
\]

In Equation (1), \( \Psi (N \times N) \) is known as the normal orthogonal matrix and \( s (N \times 1) \) is a \( K \) sparse vector.

According to Equation (1), the specific expression of compressed sensing is

\[
y = \Phi x = \Phi \Psi s = \Theta s
\]

In Equation (2), \( \Phi (M \times N) \) is the measurement matrix; \( \Theta (M \times N) \) is the sensing matrix; and \( y (M \times 1) \) is the measured value matrix. In particular, \( M < N \).

Compressed sensing demands that \( \Theta \) has the content of the restricted isometry property [46], that is to say, \( \Phi \) and \( \Psi \) are uncorrelated. In addition, the length of \( y \) ought to be

\[
M \geq cK \log \frac{N}{K}
\]

In Equation (3), \( c \) is a constant with a small value.

To exactly recover the \( s \) from the measured value matrix \( y \), theoretically, the problem of \( l_0 \) norm minimization should be solved

\[
\min || s ||_0 \\
\text{s.t.} \quad y = \Phi \Psi s
\]

However, Equation (4) is an \( NP \)-hard problem. Therefore, in general, the problem of \( l_1 \) norm minimization is used to supersede Equation (4).
\[
\begin{align*}
\min & \| s \|_1 \\
n & \text{s.t. } y = \Phi \Psi s
\end{align*}
\] (5)

There are many reconstruction algorithms for compressed sensing, the most common ones are the orthogonal matching pursuit algorithm, subspace pursuit algorithm, and the smooth \( l_0 \) norm (\( Sl_0 \)) algorithm. We chose the \( Sl_0 \) algorithm for the reconstruction in this paper.

2.2. Sigmoid Function

A common S-shaped function, also known as an S-shaped growth curve, is the Sigmoid function [39], whose expression is

\[
y = \frac{a}{1 + e^{-b(x-c)}}
\] (6)

where the range of \( y \) is \([0, a]\). We utilized the sigmoid function for quantization, so we set \( a = 255, b = 80/(15.518 \times (X_{max} - X_{min})), c = (X_{max} + X_{min})/2 \). \( X_{max} \) and \( X_{min} \) are the maximum and minimum values of \( X \), respectively. For different images, \( X_{max} \) and \( X_{min} \) are different, (i.e., the values of \( b \) and \( c \) are taken differently).

2.3. Chaotic System

2.3.1. 2D-SCLMS System

The 2D-SCLMS map is a hyperchaotic system generated based on Logistic and Sine maps [47] with the expression

\[
\begin{align*}
x_{i+1} &= \sin(4\pi^2 (u \sin(4\pi x_i (1-x_i))) + 4uy_i (1-y_i)) \\
y_{i+1} &= \sin(4\pi^2 (u \sin(4\pi y_i (1-y_i))) + 4ux_{i+1} (1-x_{i+1}))
\end{align*}
\] (7)

where \( \mu > 0.1 \) is the parameter. \( x, y \in (-1,1), i = 1, 2, \ldots \)

2.3.2. 2D-SLIM

The 2D-SLIM is a chaotic map with complex properties for image encryption [48]. Its expression is

\[
\begin{align*}
x_{i+1} &= \sin(u_x y_i) \sin(50 / x_i) \\
y_{i+1} &= u_x (1-2x_{i+1}^2) \sin(50 / y_i)
\end{align*}
\] (8)

where \( u, \mu \in (0, +\infty), x, y \in (-1,1), i = 1, 2, \ldots \) We set \( \mu_1 = 2\pi, \mu_2 = 1 \).

3. Image Encryption Process

A new encryption scenario was created and its flow chart is presented in Figure 1.
Figure 1. The flow chart of the proposed encryption algorithm.

3.1. Key Generation

The hash algorithm was utilized to create the initial parameters of the 2D-SCLMS map and the 2D-SLIM, which enhanced the relevance between the ciphertext image and the original image. First, the SHA-384 hash function generates a binary sequence composed of 384 bits and then this sequence is separated into blocks every 8 bits (i.e., 48 decimal numbers \( h_1, h_2, \ldots, h_{48} \)). The 2D-SCLMS system is mainly used for pixel transformation, row scrambling, and column scrambling, and the initial values and parameters are calculated as

\[
x_0 = \text{mod} \left( \sum_{i=1}^{10} k_i, 256 \right) / 256
\]

\[
y_0 = \text{mod} \left( \sum_{i=11}^{20} k_i, 256 \right) / 256
\]

\[
u = \text{mod} \left( \sum_{i=21}^{30} k_i, 256 \right) / 256 + \alpha
\]

The 2D-SLIM is mainly utilized to establish the measurement matrix and perform bit-cycle, where the initial values are calculated as
\[ a = \text{mod}\left(\sum_{i=43}^{48} k_i, 256\right) / 2560 \]
\[ z_0 = \text{mod}\left(\sum_{i=31}^{36} k_i, 256\right) / 256 + a \]
\[ w_0 = \text{mod}\left(\sum_{i=37}^{42} k_i, 256\right) / 256 \] (10)

3.2. Encryption Process

The proposed encryption algorithm is depicted as follows.

**Step 1:** The initial parameters \((x_0, y_0, u)\), obtained in Section 3.1, are entered into the 2D-SCLMS map for \(500 + N^2\) iterations. The first 500 values are removed to acquire the sequences \(X, Y\). Sequence \(X_1\) is obtained

\[ X_1 = \text{mod}(\text{round}(X \times 10^8), 4) \] (11)

\(X_1\) is divided equally into four sequences and each sequence is transformed into an \(N/2 \times N/2\) matrix named \(X_{11}, X_{12}, X_{13}, X_{14}\).

The sequence \(X\) is transformed into a matrix \(X_2 (N \times N)\) and \(X_2\) is divided into \(X_{21}, X_{22}\) by rows, so the matrices \(A, B\) are obtained, respectively.

\[ X_{21} = X_2(1 : N \times CR,:) \]
\[ X_{22} = X_2((1 - CR) \times N + 1 : end,:) \]
\[ A = \text{mod}(\text{floor}(X_{21} \times 10^6), 256) \]
\[ B = \text{mod}(\text{floor}(X_{22} \times 10^6), 256) \] (12)

The sequence \(Y\) is transformed into an \(N \times N\) matrix and is then divided into two parts \(Y_1, Y_2\) according to the number of rows. The matrix \(Y_1\) is arrayed in descending order by the columns, and the matrix \(Y_2\) is arrayed in ascending order by rows to obtain the index matrix \(L_1, L_2\), respectively.

\[ Y_1 = Y(1 : N \times CR,:) \]
\[ Y_2 = Y((1 - CR) \times N + 1 : end,:) \]
\[ [~, L_1] = \text{sort}(Y_1, 2, '\text{descend}') \]
\[ [~, L_2] = \text{sort}(Y_2) \] (13)

**Step 2:** The plaintext image \(P (N \times N)\) generates a discrete coefficient matrix \(P_1\) through DWT, and then matrix \(P_1\) is divided equally into four small matrices \(P_{11}, P_{12}, P_{13}, \) and \(P_{14}\).

\[ P_1 = \Psi \times P \times \Psi^T \] (14)

**Step 3:** Perform pixel transformation on \(P_{11}, P_{12}, P_{13}, P_{14}\) using matrices \(X_{11}, X_{12}, X_{13}, X_{14}\). Take \(X_{11}\) as an example for illustration.
If \(X_{11}(i, j) = 0\), then
\[
\text{temp} = P_{11}(i, j)
\]
\[
P_{11}(i, j) = P_{12}(i, j)
\]
\[
P_{12}(i, j) = \text{temp}
\]
If \(X_{11}(i, j) = 1\), then
\[
\text{temp} = P_{11}(i, j)
\]
\[
P_{11}(i, j) = P_{13}(i, j)
\]
\[
P_{13}(i, j) = \text{temp}
\]
If \(X_{11}(i, j) = 2\), then
\[
\text{temp} = P_{11}(i, j)
\]
\[
P_{11}(i, j) = P_{14}(i, j)
\]
\[
P_{14}(i, j) = \text{temp}
\]
If \(X_{11}(i, j) = 3\), then
\[
\text{temp} = P_{11}(i, j)
\]
\[
P_{11}(i, j) = P_{11}(\frac{N}{2} + 1 - i, \frac{N}{2} + 1 - j)
\]
\[
P_{13}(\frac{N}{2} + 1 - i, \frac{N}{2} + 1 - j) = \text{temp}
\]

Similarly, pixel transformation was performed again based on the values of \(X_{12}, X_{13}, X_{14}\), respectively. When the pixel transformation was over, the four matrices were combined to acquire \(P_2\).

**Step 4:** The initial values \((z_0, w_0)\), created in Section 3.1 and the parameters, are entered into the 2D-SLIM iterating \(500 + d \times M \times N\) times to produce two chaotic sequences. The first 500 values of the two sequences are removed to obtain the chaotic sequence \(Z, W\). \(M = CR \times N\), wherein \(CR\) is the compression rate and \(d\) is the sampling distance.

Sequence \(Z_1\) is acquired by sampling from sequence \(Z\) according to the sampling distance \(d\). The measurement matrix \(\Phi(M \times N)\) is generated.

\[
Z_i' = 1 - 2Z_{1id}, i = 1, 2, \ldots, MN
\]
\[
\Phi = \sqrt{\frac{2}{M}} \text{reshape}(Z_i', M, N)
\]

(16)

Take the \(MN\) values from the sequence \(W\) and transform it into a matrix \(W_i\). According to Equation (17), \(W_2\) and \(C\) can be generated.

\[
W_2 = \text{mod}(\text{floor}(W_i \times 10^6), 8)
\]
\[
C = \text{mod}(\text{floor}(W_i \times 10^6), 256)
\]

(17)

**Step 5:** Compress \(P_2\) to obtain the measurement results \(P_3\).

\[
P_3 = \Phi \times P_2
\]

(18)

**Step 6:** Quantize \(P_3\) according to the sigmoid function introduced in Section 2.2 and round the quantized result to obtain \(P_4\).

\[
P_4 = \frac{a}{1 + e^{-b(P_3 - c)}}
\]

(19)

**Step 7:** Rotate \(P_4\) counterclockwise by 180° and then scramble the columns according to the index matrix \(L_1\) to obtain \(P_5\).
\[
\begin{align*}
P_{41} &= \text{rot}90(\text{rot}90(P_4)) \\
P_i(i, j) &= P_{41}(i, L_2(i, j))
\end{align*}
\] (20)

**Step 8:** Rotate \(P_5\) counterclockwise by 180° and then scramble the rows according to the index matrix \(L_2\) to obtain \(P_6\).

\[
\begin{align*}
P_{51} &= \text{rot}90(\text{rot}90(P_5)) \\
P_6(i, j) &= P_{51}(L_2(i, j), j)
\end{align*}
\] (21)

**Step 9:** Rotate \(P_6\) counterclockwise by 180° and then perform the bit-cycle operation according to \(W_2\). If \(W_2(i, j) = 1\), then \(P_6(i, j)\) is shifted left by one bit. If \(W_2(i, j) = 2\), then \(P_6(i, j)\) is shifted left by two bits. Similarly, if \(W_2(i, j) = 7\), then \(P_6(i, j)\) is shifted left by seven bits, and finally \(P_7\) is obtained.

\[
\begin{align*}
P_{61} &= \text{rot}90(\text{rot}90(P_6)) \\
P_7 &= P_6(\text{bit-cycle})
\end{align*}
\] (22)

**Step 10:** The final ciphertext image \(P_8\) is obtained by double XOR of \(P_7\).

\[
P_8 = \text{bitxor}(\mod(\text{bitxor}(P_7, C) + A, 256), B)
\] (23)

4. Decryption Process

The specific decryption method is demonstrated below and its flow chart is presented in Figure 2.

**Figure 2.** The flow chart of the decryption scenario.
Step 1: The initial parameters are brought into the two chaotic systems. The specific method is the same as Steps 1 and 4 in Section 3.2.

Step 2: Perform the reverse operation of the double XOR on the ciphertext image $P_8$ to obtain $P_7$, then perform the inverse operation of the bit cycle and rotate $180^\circ$ counterclockwise to obtain $P_6$.

$$P_7 = IXOR(P_8)$$
$$P_{61} = P_7(Ibit\ -\ cycle)$$
$$P_6 = rot90(rot90(P_{61}))$$

(24)

Step 3: Perform the inverse scrambling operation on the rows of $P_6$ according to the index matrix $L_2$ and then rotate $180^\circ$ counterclockwise to obtain $P_5$.

$$P_5(L_2(i, j), j) = P_6(i, j)$$
$$P_5 = rot90(rot90(P_{51}))$$

(25)

Step 4: Perform the inverse scrambling operation on the columns of $P_5$ according to the index matrix $L_1$ and then rotate $180^\circ$ counterclockwise to obtain $P_4$.

$$P_4(i, L_1(i, j)) = P_5(i, j)$$
$$P_4 = rot90(rot90(P_{41}))$$

(26)

Step 5: Perform inverse quantization on $P_4$ according to the sigmoid function introduced in Section 2.2 to obtain $P_3$.

$$P_3 = -\log(\frac{a}{P_4} - 1) \times \frac{1}{b} + c$$

(27)

Step 6: Use the smooth $l_0$ norm method to reconstruct $P_2$.

$$P_2 = SL_0(P_3, \Phi)$$

(28)

Step 7: Divide $P_2$ into four blocks on average and perform inverse pixel transformation to obtain $P_1$.

$$P_1 = IPT(P_2)$$

(29)

Step 8: Perform the reverse DWT on $P_1$ to acquire the decrypted image $P$.

$$P = \Psi^T \times P_1 \times \Psi$$

(30)

5. Simulation Experiment and Performance Analysis

Multiple experiments were conducted to prove the performance of the newly presented compressed encryption scenario. The operating system used for all experiments was Windows 10 Ultimate with AMD Ryzen 2.00 GHz CPU, 8 G RAM, and 1 TB hard disk and the operating software was MATLAB R2020a. The test selected six images with a size of $512 \times 512$ (“Lena”, “Cameraman”, “Cattle”, “Einstein”, “Boat” and “Couple”) and three images with a size of $256 \times 256$ (“Barbana”, “Lena”, “Cameraman”).

5.1. Simulation Results

Figure 3 displays the original images, compressed encrypted images, and decrypted images for all of the test images. All experiments were verified with a compression ratio of 0.5 as an example. Hereon, the original images in lines (a)-(f) are $512 \times 512$ and the original images in lines (g)-(i) are size of $256 \times 256$. 
The ciphertext images were similar in noise and were smaller than the original images in Figure 3, which indicates that this scheme has a good compression and encryption effect. Furthermore, the decrypted images were of high quality and were the same size as the plaintext images, which showed that the scenario had a good reconstruction and decryption effect.

5.2. Compression Performance Analysis

5.2.1. Peak Signal-to-Noise Ratio (PSNR)

PSNR [49] was utilized for the assessment of the compression performance. This is expressed as

\[
MSE = \frac{1}{N \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} (X(i, j) - Y(i, j))^2 \]

(31)

\[
PSNR = 10 \times \log_{10}\left(\frac{255 \times 255}{MSE}\right)
\]

In Equation (31), X and Y are the plaintext and the decrypted image, respectively. The larger the PSNR value, the better the compression performance. Figure 4 shows the
simulation of “Lena” under different CRs and their corresponding PSNR values. It can be concluded that even if the CR = 0.25, the PSNR exceeded 30 db. Table 1 lists the PSNR of different images. The PSNR of the tested images exceeded 30 db, which indicates that the compression characteristic of the scenario was excellent and stable. Table 2 compares the PSNR of different compression encryption algorithms for “Lena” (256 × 256). The PSNR of our algorithm was 32.6176, which was higher than the other scenarios, which showed that the newly proposed scenario was better.

![Figure 4. The encryption and decryption results of “Lena” with different CRs. (a) CR = 0.25; (b) CR = 0.5; (c) CR = 0.75; (d) PSNR = 33.3376 db; (e) PSNR = 33.9387 db; (f) PSNR = 32.6499 db.](image)

<table>
<thead>
<tr>
<th>Image</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>33.9387</td>
</tr>
<tr>
<td>Cameraman</td>
<td>32.3925</td>
</tr>
<tr>
<td>Boat</td>
<td>31.1828</td>
</tr>
<tr>
<td>Couple</td>
<td>31.1743</td>
</tr>
<tr>
<td>Einstein</td>
<td>32.4941</td>
</tr>
<tr>
<td>Peppers</td>
<td>32.4268</td>
</tr>
</tbody>
</table>

Table 1. The PSNR (db) of the proposed algorithm for different images.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena (256 × 256)</td>
<td>30.71</td>
<td>29.23</td>
<td>31.2302</td>
<td>32.6176</td>
</tr>
</tbody>
</table>

Table 2. The PSNR (db) comparison for several schemes.

5.2.2. Structural Similarity Index Measurement (SSIM)

A momentous indicator to survey the similarity of two images is SSIM, and its range is [0, 1]. The larger the SSIM [51], the greater the similarity of the two images. The expression of SSIM is

$$SSIM(X, Y) = \frac{(2\mu_X \mu_Y + (K_1 L)^2)(2\sigma_{XY} + (K_2 L)^2)}{\left(\mu_X^2 + \mu_Y^2 + (K_1 L)^2\right)\left(\sigma_X^2 + \sigma_Y^2 + (K_2 L)^2\right)}$$

(32)
In Equation (32), $X$ and $Y$ are the plaintext and reconstructed image. $\mu_X$ and $\sigma_X^2$ are the mean value and the variance of $X$, respectively. $\mu_Y$ and $\sigma_Y^2$ are the mean value and the variance of $Y$, respectively. $\sigma_{XY}$ is the covariance of $X$ and $Y$. $M$ is the total number of windows. $L = 255$. $K_1 = 0.01$, $K_2 = 0.03$. We tested the SSIM values for multiple images, as shown in Table 3. The SSIM of the images was close to 1, which indicates that the plaintext image and reconstructed image had high similarity (i.e., the reconstruction algorithm achieved good results). Table 4 compares the SSIM of different compression encryption algorithms for “Lena” (256 × 256). The SSIM calculated by the newly proposed scenario was larger, which shows that the image reconstructed by the new scenario was more similar to the plaintext image.

Table 3. The SSIM for different images.

<table>
<thead>
<tr>
<th>Image</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>0.9001</td>
</tr>
<tr>
<td>Cameraman</td>
<td>0.8323</td>
</tr>
<tr>
<td>Boat</td>
<td>0.8447</td>
</tr>
<tr>
<td>Couple</td>
<td>0.8313</td>
</tr>
<tr>
<td>Einstein</td>
<td>0.8704</td>
</tr>
<tr>
<td>Cattle</td>
<td>0.8021</td>
</tr>
<tr>
<td>Peppers</td>
<td>0.7082</td>
</tr>
</tbody>
</table>

Table 4. The SSIM comparison for different algorithms.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena (256 × 256)</td>
<td>0.7129</td>
<td>0.6475</td>
<td>0.7337</td>
</tr>
</tbody>
</table>

5.3. Key Space Analysis

The key space of a scenario must be larger than $2^{100}$ to ensure that the algorithm is good and secure enough against brute force attacks [52].

The new algorithm has an internal key $\alpha$ and utilizes the hash-384 algorithm. Assuming that the computer has a computational precision of $10^{-14}$, the entire key space is $10^{14} + 2^{384}$, which is much larger than $2^{100}$. Thus, the scenario has a large key space and can resist violent attacks.

5.4. Key Sensitivity Analysis

A good encryption scenario is sensitive to the key, that is, even though the key changes very little, the encrypted image has a great difference.

The number of pixel change rate (NPCR) and the unified average change intensity (UACI) can be used to test the sensitivity of the scenario. These are expressed as

$$NPCR = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} | \text{Sign}(C_1(i, j) - C_2(i, j)) |$$

$$UACI = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{| C_1(i, j) - C_2(i, j) |}{255}$$

where $C_1$ and $C_2$ are two different cipher images. Table 5 lists the NPCR and UACI for multiple images.
Table 5. The key sensitivity.

<table>
<thead>
<tr>
<th>Image</th>
<th>NPCR</th>
<th>UACI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>99.5911%</td>
<td>33.4233%</td>
</tr>
<tr>
<td>Einstein</td>
<td>99.6094%</td>
<td>33.5026%</td>
</tr>
<tr>
<td>Couple</td>
<td>99.6117%</td>
<td>33.4125%</td>
</tr>
<tr>
<td>Cattle</td>
<td>99.6017%</td>
<td>33.3956%</td>
</tr>
<tr>
<td>Boat</td>
<td>99.6056%</td>
<td>33.4358%</td>
</tr>
<tr>
<td>Cameraman</td>
<td>99.5834%</td>
<td>33.5923%</td>
</tr>
<tr>
<td>Peppers</td>
<td>99.6307%</td>
<td>33.5688%</td>
</tr>
<tr>
<td>Barbana</td>
<td>99.5880%</td>
<td>33.5250%</td>
</tr>
</tbody>
</table>

The NPCR and UACI were close to 99.6094% and 33.4635%, respectively, which indicates that the scenario is sensitive to key.

5.5. Statistical Attack Analysis
5.5.1. Histogram Analysis

A momentous index to appraise the performance of encryption scenarios is the histogram. Figure 5 displays the histogram of multiple plaintext images and ciphertext images.
Figure 5. Histograms. (a) Plain image Cattle; (b) plain image Einstein; (c) plain image Boat; (d) encrypted image Cattle; (e) encrypted image Einstein; (f) encrypted image Boat.

The histograms of the plaintext images were uneven, but those of the cipher images were similar to the uniform distribution, which illustrates that the scenario resisted statistical attacks.

In addition, we utilized the histogram variance to survey the effectiveness of this algorithm.

\[
\text{Var}(Z) = \frac{1}{256^2} \sum_{i=0}^{255} \sum_{j=0}^{255} (z_i - z_j)^2
\]  

(34)

In Equation (34), \(z_i\) and \(z_j\) represent the number of pixel values corresponding to \(i\) and \(j\). The histogram variance of the plaintext images were very large, and the maximum could reach \(10^6\), while those of the ciphertext images were small, only \(10^2\), and the minimum was 115.8203 in Table 6. This shows that the histogram of the ciphertext images was flatter.

Table 6. The histogram variance of multiple images.

<table>
<thead>
<tr>
<th>Image</th>
<th>Original Image</th>
<th>Encrypted Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>1.0827 (\times 10^6)</td>
<td>461.9766</td>
</tr>
<tr>
<td>Couple</td>
<td>1.1955 (\times 10^6)</td>
<td>515.0078</td>
</tr>
<tr>
<td>Cameraman</td>
<td>1.6741 (\times 10^6)</td>
<td>584.1484</td>
</tr>
<tr>
<td>Boat</td>
<td>1.5359 (\times 10^6)</td>
<td>558.7188</td>
</tr>
<tr>
<td>Einstein</td>
<td>1.1987 (\times 10^6)</td>
<td>455.4297</td>
</tr>
<tr>
<td>Cattle</td>
<td>7.5077 (\times 10^5)</td>
<td>466.6719</td>
</tr>
<tr>
<td>Barbana</td>
<td>6.0765 (\times 10^5)</td>
<td>136.4609</td>
</tr>
<tr>
<td>Peppers</td>
<td>3.6777 (\times 10^4)</td>
<td>115.8203</td>
</tr>
</tbody>
</table>

Table 7 compares the histogram variance of “Lena” (256 \(\times\) 256) with different algorithms. The histogram variance of the new scenario was smaller, explaining that the histogram was flatter. That is, the newly proposed algorithm was more resistant to statistical attacks.

Table 7. The histogram variance comparison of “Lena” (256 \(\times\) 256) using different algorithms.

<table>
<thead>
<tr>
<th>Plain Image</th>
<th>Ref. [37]</th>
<th>Ref. [50]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>3.0665 (\times 10^4)</td>
<td>181.7109</td>
<td>121.4063</td>
</tr>
</tbody>
</table>

To appraise the performance of the new scenario to resist statistical attacks, this paper utilized the chi-square [33], the expression of which is

\[
\chi^2 = \sum_{i=0}^{N-1} \frac{(u_i - u_0)^2}{u_0}
\]  

(35)

In Equation (35), \(u_i\) is the frequency of value \(i\). \(u_0 = MN/2^8\). Table 8 enumerates the chi-square results of multiple images. The values for seven images were less than 293.2478 (255 degrees of freedom and 5% confidence), which shows that this algorithm has good effects and can resist statistical attack. Table 9 compares the results of several scenarios for “Lena” (256 \(\times\) 256). The chi-square value of the newly proposed encryption scenario was the smallest, which shows that this scenario was more resistant to statistical attacks.

Table 8. The chi-square values.

<table>
<thead>
<tr>
<th>Image</th>
<th>Lena</th>
<th>Couple</th>
<th>Einstein</th>
<th>Cattle</th>
<th>Boat</th>
<th>Peppers</th>
<th>Barbana</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chi-square</td>
<td>230.9883</td>
<td>257.5039</td>
<td>227.7148</td>
<td>233.3359</td>
<td>279.3594</td>
<td>231.6406</td>
<td>272.9219</td>
</tr>
</tbody>
</table>
Table 9. The chi-square of “Lena” (256 × 256) using different algorithms.

<table>
<thead>
<tr>
<th>Plain Image</th>
<th>Ref. [37]</th>
<th>Ref. [50]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>30665.7</td>
<td>253.3125</td>
<td>242.8125</td>
<td>211.2656</td>
</tr>
</tbody>
</table>

5.5.2. Correlation Analysis

The encryption scenario is to break the correlation of the original image. The evaluation index to assess the effectiveness of the scenario is the correlation coefficient, the expression of which is

\[
\rho_{xy} = \frac{cov(x, y)}{\sqrt{D(x)D(y)}}
\]

\[
cov(x, y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - \frac{1}{N} \sum_{i=1}^{N} x_i)(y_i - \frac{1}{N} \sum_{i=1}^{N} y_i)
\]

\[
D(x) = \frac{1}{N} \sum_{i=1}^{N} (x_i - \frac{1}{N} \sum_{i=1}^{N} x_i)^2
\]

In Equation (36), \(x\) and \(y\) are the image adjacent pixels. Table 10 enumerates the correlation coefficients of different original images and ciphertext images. The comparison values of “Lena” (256 × 256) with several encryption scenarios are enumerated in Table 11.

Table 10. The correlation coefficients for different images.

<table>
<thead>
<tr>
<th>Image</th>
<th>Horizontal</th>
<th>Vertical</th>
<th>Diagonal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>0.9840</td>
<td>0.9835</td>
<td>0.9717</td>
</tr>
<tr>
<td></td>
<td>−0.0032</td>
<td>−0.0037</td>
<td>−0.00085</td>
</tr>
<tr>
<td>Cameraman</td>
<td>0.9853</td>
<td>0.9870</td>
<td>0.9765</td>
</tr>
<tr>
<td></td>
<td>−0.00014</td>
<td>−0.0017</td>
<td>0.0030</td>
</tr>
<tr>
<td>Boat</td>
<td>0.9833</td>
<td>0.9727</td>
<td>0.9617</td>
</tr>
<tr>
<td></td>
<td>−0.0015</td>
<td>0.0058</td>
<td>−0.0018</td>
</tr>
<tr>
<td>Couple</td>
<td>0.9624</td>
<td>0.9650</td>
<td>0.9386</td>
</tr>
<tr>
<td></td>
<td>0.0037</td>
<td>0.0045</td>
<td>0.0014</td>
</tr>
<tr>
<td>Einstein</td>
<td>0.9687</td>
<td>0.9644</td>
<td>0.9548</td>
</tr>
<tr>
<td></td>
<td>−0.0014</td>
<td>−0.00043</td>
<td>−0.0021</td>
</tr>
<tr>
<td>Cattle</td>
<td>0.8573</td>
<td>0.9103</td>
<td>0.8423</td>
</tr>
<tr>
<td></td>
<td>−0.0057</td>
<td>−0.0016</td>
<td>−0.0015</td>
</tr>
<tr>
<td>Peppers</td>
<td>0.9490</td>
<td>0.9452</td>
<td>0.9039</td>
</tr>
<tr>
<td></td>
<td>−0.0048</td>
<td>−0.0021</td>
<td>−0.0014</td>
</tr>
<tr>
<td>Barbana</td>
<td>0.9056</td>
<td>0.7542</td>
<td>0.7158</td>
</tr>
<tr>
<td></td>
<td>−0.00066</td>
<td>0.0057</td>
<td>0.0018</td>
</tr>
</tbody>
</table>

Table 11. The correlation coefficients of “Lena” (256 × 256) using different schemes.

<table>
<thead>
<tr>
<th>Direction</th>
<th>Horizontal</th>
<th>Vertical</th>
<th>Diagonal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain image</td>
<td>0.9746</td>
<td>0.9651</td>
<td>0.9539</td>
</tr>
<tr>
<td>Ref. [31]</td>
<td>0.0009</td>
<td>−0.0062</td>
<td>−0.0087</td>
</tr>
<tr>
<td>Ref. [37]</td>
<td>−0.0160</td>
<td>−0.0044</td>
<td>−0.0052</td>
</tr>
<tr>
<td>Ref. [48]</td>
<td>−0.0015</td>
<td>0.0041</td>
<td>0.0069</td>
</tr>
<tr>
<td>Ref. [50]</td>
<td>0.0076</td>
<td>−0.0066</td>
<td>−0.0035</td>
</tr>
</tbody>
</table>
The correlation coefficients of the plaintext images were close to 1, but those of the ciphertext images were about 0 in Table 8, and that value in our scenario was smaller in Table 11, which shows that the scenario had better resistance to statistical attacks.

The correlation of “Lena” is presented in Figure 6 for clear observation. The correlation of the plaintext image in three directions was diagonal, but those of the ciphertext image were interspersed over the whole range. This shows that the encryption scenario effectively abated the correlation.

![Figure 6](image)

**Figure 6.** Correlation. (a–c) Horizontal, vertical, and diagonal directions of plain “Lena”; (d–f) Horizontal, vertical, and diagonal directions of encrypted “Lena”.

5.5.3. Information Entropy (IE)

The quota to assess the overall randomness of images is the IE and its expression is

$$H(s) = -\sum_{i=0}^{M} p(s_i) \log_2 p(s_i)$$  \hspace{1cm} (37)

In Equation (37), $M = 255$, $p(s)$ is the probability of $s$. The closer the IE is to 8, the better the algorithm [54].

Table 12 lists the IE of several original and ciphertext images. Table 13 lists the comparison value of “Lena” (256 × 256) using different algorithms.
Table 12. The IE of several images.

<table>
<thead>
<tr>
<th>Image</th>
<th>Original Image</th>
<th>Encryption Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>7.3920</td>
<td>7.9987</td>
</tr>
<tr>
<td>Couple</td>
<td>7.2010</td>
<td>7.9986</td>
</tr>
<tr>
<td>Cameraman</td>
<td>7.0480</td>
<td>7.9987</td>
</tr>
<tr>
<td>Boat</td>
<td>7.1914</td>
<td>7.9985</td>
</tr>
<tr>
<td>Einstein</td>
<td>7.2655</td>
<td>7.9987</td>
</tr>
<tr>
<td>Cattle</td>
<td>7.3579</td>
<td>7.9987</td>
</tr>
<tr>
<td>Peppers</td>
<td>7.5327</td>
<td>7.9949</td>
</tr>
<tr>
<td>Barbana</td>
<td>5.0030</td>
<td>7.9940</td>
</tr>
</tbody>
</table>

Table 13. The information entropy of “Lena” (256 × 256) using different algorithms.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5683</td>
<td>7.9944</td>
<td>7.9935</td>
<td>7.9946</td>
<td>7.9954</td>
</tr>
</tbody>
</table>

The IE of the ciphertext image was extremely close to 8 in Table 12. The IE of the newly encryption algorithm was higher than the other algorithms in Table 13. Therefore, this algorithm had very good results. The encrypted image had stronger randomness and was resistant to statistical attacks.

5.5.4. Local Information Entropy (LIE)

A momentous metric for analyzing the randomness of the local image is the LIE. Some non-overlapping image blocks are randomly selected and the LIE can be obtained by calculating the IE of each block and then taking the average value. The expression is

$$\overline{LH_{k,T_i}}(P) = \frac{1}{k} \sum_{i=1}^{k} H(S_i)$$  \hspace{1cm} (38)

where $H(S_i)$ is the IE of sub-block $S$. Let $k = 30$, $T_i = 1936$ for calculation. When the confidence level is 0.05, the range of LIE is [7.901901305, 7.903037329] [32]. Table 14 lists the LIE of different images (512 × 512). The LIE of all test images passed the experiment, which shows that the local image had good randomness.

Table 14. The LIE (512 × 512).

<table>
<thead>
<tr>
<th>Image</th>
<th>LIE</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>7.902316286</td>
<td>Pass</td>
</tr>
<tr>
<td>Cameraman</td>
<td>7.902787296</td>
<td>Pass</td>
</tr>
<tr>
<td>Boat</td>
<td>7.902113612</td>
<td>Pass</td>
</tr>
<tr>
<td>Cattle</td>
<td>7.902520981</td>
<td>Pass</td>
</tr>
<tr>
<td>Einstein</td>
<td>7.902336151</td>
<td>Pass</td>
</tr>
<tr>
<td>Couple</td>
<td>7.902842150</td>
<td>Pass</td>
</tr>
</tbody>
</table>

5.6. Differential Attack Analysis

An excellent encryption scenario is sensitive to the plaintext image, in other words, even though the original image has very small changes, the encrypted image can be completely different.

The NPCR and UACI are indicators used to measure whether the algorithm can resist differential attacks. When the NPCR > NPCR*, the NPCR passes the test. When the UACI is between [UACI*−α, UACI*+α], the UACI passes the test [55]. The NPCR and UACI statistical tests are shown in Tables 15 and 16.
Table 15. The NPCR statistical test.

<table>
<thead>
<tr>
<th>Image</th>
<th>NPCR</th>
<th>Theoretical NPCR Critical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>512 × 512</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lena</td>
<td>99.5941%</td>
<td>Pass</td>
</tr>
<tr>
<td>Einstein</td>
<td>99.6460%</td>
<td>Pass</td>
</tr>
<tr>
<td>Couple</td>
<td>99.5987%</td>
<td>Pass</td>
</tr>
<tr>
<td>Cattle</td>
<td>99.6185%</td>
<td>Pass</td>
</tr>
<tr>
<td>Boat</td>
<td>99.6185%</td>
<td>Pass</td>
</tr>
<tr>
<td>Cameraman</td>
<td>99.6048%</td>
<td>Pass</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Image</th>
<th>NPCR</th>
<th>Theoretical NPCR Critical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>256 × 256</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lena</td>
<td>996368%</td>
<td>Pass</td>
</tr>
<tr>
<td>Barbana</td>
<td>996368%</td>
<td>Pass</td>
</tr>
<tr>
<td>Peppers</td>
<td>996154%</td>
<td>Pass</td>
</tr>
</tbody>
</table>

Table 16. The UACI statistical test.

<table>
<thead>
<tr>
<th>Image</th>
<th>UACI</th>
<th>Theoretical UACI Critical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>512 × 512</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lena</td>
<td>33.4078%</td>
<td>Pass</td>
</tr>
<tr>
<td>Einstein</td>
<td>33.5236%</td>
<td>Pass</td>
</tr>
<tr>
<td>Couple</td>
<td>33.4989%</td>
<td>Pass</td>
</tr>
<tr>
<td>Cattle</td>
<td>33.5140%</td>
<td>Pass</td>
</tr>
<tr>
<td>Boat</td>
<td>33.4173%</td>
<td>Pass</td>
</tr>
<tr>
<td>Cameraman</td>
<td>33.4373%</td>
<td>Pass</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Image</th>
<th>UACI</th>
<th>Theoretical UACI Critical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>256 × 256</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lena</td>
<td>33.4428%</td>
<td>Pass</td>
</tr>
<tr>
<td>Barbana</td>
<td>33.4929%</td>
<td>Pass</td>
</tr>
<tr>
<td>Peppers</td>
<td>33.5082%</td>
<td>Pass</td>
</tr>
</tbody>
</table>

The NPCR and UACI of all test images were very close to the ideal values, and all passed the NPCR and UACI tests. Therefore, the algorithm could effectively resist differential attacks.

5.7. NIST SP 800-22 Analysis

The NIST SP 800-22 statistical test suite is published by the National Institute of Standards and Technology for testing sequences for randomness [56]. Therefore, we set the confidence level to 0.01 to evaluate the randomness of the ciphertext image. The results are listed in Table 17. All data passed the test, indicating that the ciphertext image had good randomness.
Table 17. The NIST SP 800-22 test.

<table>
<thead>
<tr>
<th>Test Items</th>
<th>p-Value</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency test</td>
<td>0.332829</td>
<td>Pass</td>
</tr>
<tr>
<td>Block frequency test</td>
<td>0.589821</td>
<td>Pass</td>
</tr>
<tr>
<td>Cusum-forward test</td>
<td>0.577516</td>
<td>Pass</td>
</tr>
<tr>
<td>Cusum-reverse test</td>
<td>0.201550</td>
<td>Pass</td>
</tr>
<tr>
<td>Runs test</td>
<td>0.315933</td>
<td>Pass</td>
</tr>
<tr>
<td>Longest run test</td>
<td>0.471291</td>
<td>Pass</td>
</tr>
<tr>
<td>Rank test</td>
<td>0.452825</td>
<td>Pass</td>
</tr>
<tr>
<td>FFT test</td>
<td>0.510298</td>
<td>Pass</td>
</tr>
<tr>
<td>Non-overlapping template test</td>
<td>0.510816</td>
<td>Pass</td>
</tr>
<tr>
<td>Overlapping template test</td>
<td>0.387884</td>
<td>Pass</td>
</tr>
<tr>
<td>Universal test</td>
<td>0.545638</td>
<td>Pass</td>
</tr>
<tr>
<td>Approximate entropy test</td>
<td>0.463226</td>
<td>Pass</td>
</tr>
<tr>
<td>Random-excursions test (x = −1)</td>
<td>0.159822</td>
<td>Pass</td>
</tr>
<tr>
<td>Random-excursions variant test (x = 1)</td>
<td>0.124450</td>
<td>Pass</td>
</tr>
<tr>
<td>Serial1 test</td>
<td>0.550327</td>
<td>Pass</td>
</tr>
<tr>
<td>Serial2 test</td>
<td>0.584547</td>
<td>Pass</td>
</tr>
<tr>
<td>Linear complexity test</td>
<td>0.403982</td>
<td>Pass</td>
</tr>
</tbody>
</table>

5.8. Time Complexity

Time complexity is an important quantitative criterion to evaluate the feasibility of an encryption algorithm, and it requires the algorithm to be easy to execute. If the running time of the algorithm is too long, it does not meet the requirements of real-time performance. This paper tested the encryption time of multiple images, which are presented in Table 18. The time of all 256 × 256 images was less than 1 s, and the time of 512 × 512 images was less than 3 s, which greatly proves that the algorithm is real-time.

Table 18. The encryption runtime (Unit: s).

<table>
<thead>
<tr>
<th>Image</th>
<th>Lena</th>
<th>Couple</th>
<th>Einstein</th>
<th>Cattle</th>
<th>Boat</th>
<th>Lena</th>
<th>Peppers</th>
<th>Barbana</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>2.8783</td>
<td>2.9142</td>
<td>2.8466</td>
<td>2.9652</td>
<td>2.8521</td>
<td>0.9598</td>
<td>0.9463</td>
<td>0.9421</td>
</tr>
</tbody>
</table>

5.9. Anti-Noise Attack Analysis

As it is subject to various noise interference during transmission, an excellent encryption scenario should resist noise attacks. The salt and pepper noise is tested at intensities of 0.005%, 0.05%, and 0.1% in “Lena”, as shown in Figure 7.

![Noise attack](image)

Figure 7. Noise attack: (a) 0.005% noise; (b) 0.05% noise; (c) 0.1% noise.

Even though the added noise intensity was 0.1%, the cipher image could be decrypted and information could be viewed. This shows that the scenario resisted noise attacks.
In order to measure the anti-noise ability of the encryption algorithm more accurately, this paper tested the PSNR. For three different noise intensities, their corresponding PSNR are presented in Table 19. When the noise intensity was 0.005%, the PSNR was 33.2311, even if the noise intensity increased to 0.1%, the PSNR was greater than 29, which shows that the algorithm had a strong resistance to noise.

Table 19. The PSNR test for noise resistance.

<table>
<thead>
<tr>
<th>Noise</th>
<th>0.005%</th>
<th>0.05%</th>
<th>0.1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>33.2311</td>
<td>29.5916</td>
<td>29.1613</td>
</tr>
</tbody>
</table>

6. Discussion

The encryption algorithm based on the chaotic system and compressed sensing proposed in this paper could resist various attacks, and had security and timeliness. However, it also has certain limitations. The measurement matrix is generated by the universal method, that is, the chaotic sequence generated by the chaotic system constitutes the measurement matrix. We should conduct further research in the future to make better use of the chaotic characteristics of the chaotic system to construct a better measurement matrix to make the compression and encryption more convenient and obtain better compression and encryption effects.

7. Conclusions

The paper proposed a new image compression and encryption scenario based on CS and two chaotic maps. The pixel transform operation was performed before the compressed sensing first, which is beneficial to increase the image reconstruction quality. In the quantization process, we made full use of the performance of the sigmoid function to quantize the matrix to the interval \([0, 255]\). In the scrambling process, we combined rotation with row and column scrambling, which tremendously reduced the correlation. Finally, the cipher image was created by double XOR after the bit-cycle operation.

After a series of tests and experimental analysis, the new scenario had a huge key space and was sensitive to keys. In addition, various experiments against statistical analysis attacks were carried out in this paper such as histograms and their statistical analysis, information entropy, correlation, and local information entropy. The information entropy was very close to 8, and the correlation coefficient was close to 0. Subsequently, the algorithm was also resistant to differential attacks, brute force attacks, and noise attacks. All of the test images were close to the standard values of the NPCR and UACI and passed the statistical analysis test, and their PSNR exceeded 29 for 0.1% intensity noise. The bit sequence of the ciphertext image passed the NIST randomness test.

The significance of this paper was to combine the two chaotic systems with compressed sensing, which can not only fully utilize the practicability of chaos theory for image encryption, but can also compress ciphertext images to meet the needs of the transmission bandwidth. The encryption algorithm proposed in this paper is not only resistant to various attacks, but also has real-time performance and is a secure encryption scheme.

In the future, we should focus on the further combination of the chaotic system and compressed sensing and its application in medicine or larger fields.

**Author Contributions:** Conceptualization, J.W.; Methodology, J.W. and A.A.A.E.-L.; Software, J.W.; Formal analysis, J.W. and A.A.A.E.-L.; Investigation, J.W. and A.A.A.E.-L.; Data curation, X.S.; Writing—original draft preparation, J.W. and X.S.; Writing—review and editing, A.A.A.E.-L.; Visualization, X.S. and A.A.A.E.-L.; Supervision, X.S.; Project administration, X.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Postdoctoral Research Foundation of China (2018M631914) and the Heilongjiang Provincial Postdoctoral Science Foundation (CN) (LBH-
Z17042). This work was also supported by the EIAS Data Science Lab, College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data sharing not applicable.

**Conflicts of Interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**References**


