Hindawi
Scientific Programming
Volume 2020, Article ID 6612390, 22 pages
https://doi.org/10.1155/2020/6612390

Research Article

Improved Chaos-Based Cryptosystem for Medical Image Encryption and Decryption

Mohamed Gafsi,[1] **Nessrine Abbassi**,[1] **Mohamed Ali Hajjaji**,[1,2] **Jihene Malek**,[1,2] **and Abdellatif Mtibaa**[1,3]

[1]*Université de Monastir, Laboratoire d'Electronique et de Microélectronique, LR99ES30, Monastir 5000, Tunisia*
[2]*Higher Institute of Applied Sciences and Technology, Sousse University, Sousse, Tunisia*
[3]*Université de Monastir, Ecole Nationale d'Ingénieurs de Monastir, Monastir 5000, Tunisia*

Correspondence should be addressed to Mohamed Ali Hajjaji; daly_fsm@yahoo.fr

Received 3 October 2020; Revised 9 November 2020; Accepted 26 November 2020; Published 18 December 2020

Academic Editor: Zhaoqing Pan

Copyright © 2020 Mohamed Gafsi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the medical sector, the digital image is multimedia data that contain secret information. However, designing an efficient secure cryptosystem to protect the confidential images in sharing is a challenge. In this work, we propose an improved chaos-based cryptosystem to encrypt and decrypt rapidly secret medical images. A complex chaos-based PRNG is suggested to generate a high-quality key that presents high randomness behaviour, high entropy, and high complexity. An improved architecture is proposed to encrypt the secret image that is based on permutation, substitution, and diffusion properties. In the first step, the image's pixels are randomly permuted through a matrix generated using the PRNG. Next, pixel's bits are permuted using an internal condition. After that, the pixels are substituted using two different *S*-boxes with an internal condition. In the final step, the image is diffused by XORing pixels with the key stream generated by the PRNG in order to acquire an encrypted image. *R* rounds of encryption can be performed in a loop to increase the complexity. The cryptosystem is evaluated in depth by his application on several medical images with different types, contents, and sizes. The obtained simulation results demonstrate that the system enables high-level security and performance. The information entropy of the encrypted image has achieved an average of 7.9998 which is the most important feature of randomness. The algorithm can take full advantage of parallelism and pipeline execution in hardware implementation to meet real-time requirements. The PRNG was tested by NIST 800-22 test suit, which indicates that it is suitable for secure image encryption. It provides a large key space of 2^{192} which resists the brute-force attack. However, the cryptosystem is appropriate for medical image securing.

1. Introduction

In the medical sector, the digital image is multimedia data that contain secret information. However, designing an effective cryptosystem to protect medical image content is a challenge. Using public or shared digital networks, images are vulnerable to potentially more destructive attacks such as replay or human-based attacks, brute-force, and statistical attack. The need for effective cryptographic solutions for medical image requires the development of an improved algorithm and implementation. To protect the image against new generations of attacks, encryption solutions should guarantee the confidentiality of the image. Confidentiality is achieved by encryption to make data unintelligible and unusable even if the data is lost or hacked. Among encryption schemes, symmetric encryption is the best cryptographic solution that permits the confidentiality of large volume data. In this innovative idea, chaos is an effective axis of modern cryptography challenging existing traditional symmetric encryption systems like the Advanced Encryption System (AES) [1]. Chaos systems have many significant advantages such as highly sensitive to initial conditions, deterministic random numbers, ergodicity, structure complexity, large key space, flexibility, and large periodicity.

Unlike asymmetric encryption, symmetric encryption has several modes of operation. Since 2001, five modes have been recommended by NIST which are ECB, CBC, OFB, CTS, and CTR. Among these, the CTR mode of encryption is commonly used in high-speed networks thanks to its high performance. The CTR architecture can take full advantage of parallel and pipelining execution and can achieve a high-level performance using reasonable hardware resources.

After this innovative idea was investigated, many researchers turned to design chaos-based symmetric cryptosystem algorithms for ordinary and medical image encryption. This is using different types of chaotic models such as the Lorenz and Chen system, skew tent map, and logistic map [2, 3]. Jeevitha [4] presented a cryptosystem algorithm for medical image encryption. In the first step, the medical image was decomposed into some planes using discrete wavelet transform. Edge maps were generated with the same or different thresholds from the original image and the binary images of equal size considered as with the original planes. Then, the XOR diffusion between the edge maps and the planes was carried out. Next, the positions of the obtained plane at the last step were scrambled. Finally, planes were combined to form the encrypted image. Jizhao [5] put forward a simple cryptosystem algorithm for medical image protection. A PRNG based on a four-dimensional chaotic system was proposed to generate the key. The original medical image was encrypted using a diffusion-confusion as architecture. The confusion property was done by a simple substitution *S*-box. However, the diffusion property was obtained by XORing the image pixel with a key stream. Tsafack et al. [6] presented an image encryption system based on a simple chaotic system. To generate the encryption key, an electronic circuit based on a dynamical four-dimensional chaotic system is designed. This implementation mainly increases the execution time. For image encryption, a simple confusion and diffusion architecture was designed. Firstly, the substitution *S*-box and a key stream were produced by utilizing the PRNG. Then, the image's pixels were substituted using the *S*-box. Finally, the result pixels were XORed by the key stream in order to obtain the encrypted image. Their proposed cryptosystem was tested on some medical images. Xingyuan [7] proposed a chaos-based cryptosystem for colour image encryption. Firstly, the image was divided into four blocks. Then, the blocks were scrambled by employing Arnold's chaotic map. Next, the image's blocks were diffused by a combination method between the Boolean network and semitensor products. Finally, the blocks were combined to form the encrypted image. Yasir et al. [8] suggested an image cryptosystem based on confusion and compression. Firstly, the simple lossless Lempel–Ziv–Welch (LZW) data compression algorithm was used to compress the original image. Then, the Chebyshev chaotic map was used to select an *S*-box from a collection of 40,320 available *S*-boxes. Finally, the compressed image was encrypted by substituting the pixels with the selected *S*-box. Using only a confusion process to encrypt the secret image is not secure against attacks. Huijuan [9] proposed a simple cryptosystem algorithm for image encryption. A PRNG based on the two-dimensional logistic-adjusted sine map was designed for encryption key generation. Two mechanisms were used for orbit perturbation and dynamic state variable selection. Their proposed image encryption algorithm includes a permutation and a XOR diffusion procedure. Hongjun [10] put forward an image cryptosystem based on DNA sequence and two chaotic maps. The scheme is symmetric and they adopt a confusion-diffusion as encryption architecture. The initial parameters of chaos maps are generated using the MD5 hash. The image was confused using PWLCM map and confused using DNA and Chebyshev map. Zhou [11] suggested an image cryptosystem based on a combination between the 3D orthogonal Latin squares (3D-OLSs) and a matching matrix. Firstly, the 3D sine map was used to generate three chaotic sequences. Next, a 3D orthogonal Latin square and a matching matrix were produced by using the chaotic sequences. Then, the 3D-OLSs and the matching matrix were jointly used to permute the original image. After that, all planes of the permuted matrix were divided into sixteen blocks of the same size. The chaotic sequence was sorted and a position matrix was generated. According to the position matrix, the blocks of each plane were linked and shifted by using a cyclic shift operation, and then, a new matrix was generated. Finally, the encrypted image was generated by executing a diffusion operation for the new matrix. Zhang [12] proposed a simple cryptosystem algorithm for image encryption. In designing, he has used transformation, permutation, and XOR diffusion as encryption architecture. The transformation was obtained by employing the discrete wavelet transform of the image. The permutation was carried out by substitution *S*-box. However, the diffusion property was obtained by scrambling the image's pixels with a key stream that was generated by a simple PRNG based on the PWLCM chaotic map. In the substitution step, Yong adopts the *S*-box of the AES standard, which was not a secure substitution method because the *S*-box of AES was not dynamic.

The challenge is that traditionally, key generation, encryption, authentication, and integrity have been complex and computationally costly to execute while keeping in mind the issue related to the security level. All mentioned image cryptosystems have many weaknesses. It is sequential, too long in design and calculation, which greatly increases the execution time. In this work, we propose an improved chaos-based symmetric cryptosystem for fast image encryption and decryption. The goal is to achieve high-level security and high performance with low computational complexity and reasonable resources. Our contribution is as follows:

(i) Design of improved chaos-based PRNG with the goal to enlarge the key space, increase the entropy, randomness, and complexity, and avoid the key's sequence relationship and determinism. This permits the generating of high-quality key streams with high randomness behaviour, unpredictability, and complexity.

(ii) Design of fast and secure encryption and decryption architecture based on permutation, substitution, and diffusion properties. This permits enhancing the randomness and decreasing the correlation. The goal is to achieve a high-level performance with low computational cost and with reasonable resources.

(iii) Undertake in-depth experimental measurements for medical images with different type, content, and size to evaluate the strength of the proposed cryptosystem against the new generation of attacks.

(iv) Undertake an evaluation study of the performance of the execution and compare the result with other recent works.

This paper is planned in four parts as follows: in Section 2, the proposed image cryptosystem algorithm is described. The simulation, analysis, evaluation, and validation of the proposed algorithm are given in Section 3. Section 4 concludes the work.

2. Proposed Cryptosystem Algorithm

The symmetric cryptographic scheme is the best solution to encrypt and decrypt large volume data. The proposed cryptosystem algorithm is a symmetric scheme based on confusion and diffusion properties. To generate a high-quality key, a complex chaos-based PRNG is suggested. The general view of the proposed cryptosystem is depicted in Figure 1.

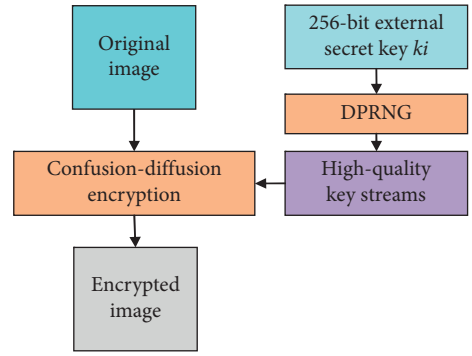


Figure 1: General view of the proposed cryptosystem.

2.1. Pseudorandom Number Generator. PRNGs are used to generate key useful for encryption. The proposed PRNG is a chaos-based key generator. A complex PRNG architecture is designed with the goal to increase the key complexity, entropy, randomness, sensitivity, and key space and to avoid determinism, correlation, and key dependence.

The proposed PRNG is illustrated in Figure 2. It includes three data processing blocks: a dynamical state generator (DSG), a complex chaotic design (CCD), and a convertor block. Three different chaotic systems are employed in designing to enhance the complexity of the key. The chaotic systems are maintained in parallel. This permits generating many key streams at a time that enlarge the key length and the key space. The PRNG requires a 256-bit external secret key to initialize the underlying system (equation (1)). This secret key is used to generate the initial state of the PRNG. However, the use of the same initial key permits always obtaining the same random number sequence. To avoid transient effects, key sequence relationship, dependence, and determinism, a random state generator is attached to the complex chaotic design to perturb the underlying system dynamically. However, the PRNG exhibits complex chaotic behaviour. It depends not only on the initial key but also on intermediate random states. This permits to increase the complexity of the key against attacks.

$$Ki = k_1|k_2|k_3|\cdots|k_{32}. \tag{1}$$

After random values generation by the chaotic design, the convertor block is used that permits modulating the generated values into 32-bit numbers (equation (2)). As a result, a sequence of independent numbers PRNS is obtained and its properties are statistically independent, uniformly distributed, and unpredictable. In addition, the proposed PRNG depends not only on the initial secret key but also on internal random states generated dynamically:

$$\text{PRNS} = \left(n_i \times 10^{12}\right) \bmod 2^{32}. \tag{2}$$

The used chaotic systems are the Henon map, 2D logistic map in a complex set, and the Baker map. Their mathematical models are described in equations (3)–(5), respectively:

$$\begin{cases} x_{n+1} = 1 - ax_n^2 + y_n, \\ y_{n+1} = bx_n, \end{cases}$$
$$x_0 = \frac{(k_1 \oplus k_2 \oplus \cdots \oplus k_5)}{2^8}, \tag{3}$$
$$y_0 = \frac{(k_6 \oplus k_7 \oplus \cdots \oplus k_{10})}{2^8}.$$

The Henon map has a state of two variables (x, y), and a and b are the system parameters. It exhibits chaotic behaviour for certain parameter values and initial conditions. When $a = 1.4$ and $b = 0.3$, the system has chaotic behaviour [13]. The initial state (x_0, y_0) of the Henon map is derived from the initial key ki:

$$\begin{cases} x_{i+1} = yx_i(1 - x_n) + \lambda y_i^2, \\ y_{i+1} = \lambda y_i(1 - 2x_i), \end{cases}$$
$$x_0 = \frac{(k_{11} \oplus k_{12} \oplus \cdots \oplus k_{15})}{2^8}, \tag{4}$$
$$y_0 = \frac{(k_{16} \oplus k_{17} \oplus \cdots \oplus k_{20})}{2^8}.$$

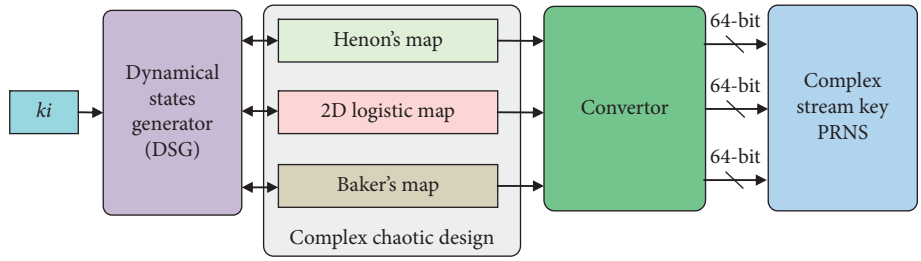


Figure 2: General view of the proposed PRNG.

The 2D logistic map in a complex set has a state of two variables (x, y) and one parameter λ. For $\lambda \epsilon [0, 4]$, the system has chaotic behaviour [14]. The initial state (x_0, y_0) of the 2D logistic map is derived from the initial key ki:

$$(x_{n+1}, y_{n+1}) = \begin{cases} \left(\frac{x_n}{p}, p y_n\right), \\ \left(\frac{x_n - p}{1-p}, (1-p) y_n + 1 - p\right), \end{cases} \quad (5)$$
$$x_0 = \frac{(k_{21} \oplus k_{22} \oplus \cdots \oplus k_{26})}{2^8},$$
$$y_0 = \frac{(k_{27} \oplus k_{28} \oplus \cdots \oplus k_{32})}{2^8}.$$

The Baker map has a state of two variables (x, y) and one parameter p. For $p = 0.5$, the system has chaotic behaviour [15]. The initial state (x_0, y_0) of the Baker map is derived from the initial key ki.

2.2. Encryption Phase. A symmetric scheme is adopted for image encryption. The cryptosystem uses the Secure Hash Algorithm (SHA-256) to generate a unique 256-bit hash value fully related to the secret image I as follows:

$$\text{Hi} = \text{SHA} - 256\,(\text{I}). \quad (6)$$

The image's hash value is considered as the initial secret key of the cryptosystem that is named ki. This key is used to initialize the PRNG. An improved PRNG-based symmetric scheme is designed to encrypt the secret image. The general architecture is depicted in Figure 3. Both confusion and diffusion properties are employed in encryption architecture. The confusion property is obtained by pixel permutation and substitution. However, the diffusion property is obtained by XORing the image's pixels with a key stream. Data encryption steps are as follows:

(i) Step 1: read a medical image I with any size $S = N \times M \times O$. N and M are the image's dimensions and O is the number of layers. For colour image $O = 3$, the image is decomposed firstly into red, blue, and green components, and then, each component is encrypted separately using the encryption system.

(ii) Step 2: bits permutation of pixels. The image's pixels are permuted by cycling right shift or cycling left shift according to the pixel's position parity. Figure 4 illustrates the process.

(1) If the position of the pixel is pair, then the pixel's bits are permuted by cycling right shift of 2 bits:

$$\text{PP} = \text{pixel} \ll 2. \quad (7)$$

(iii) Step 3: random permutation of the pixel's position. Here, a permutation matrix (PM) of size $M \times N$ is generated using the PRNG. The matrix contains random indices that to be followed to permute the position of the image's pixels. The principle to generate the matrix PM is illustrated in Figure 5. Firstly, the PRNG is iterated to produce a sequence of $M \times N$ random numbers. Then, the numbers are sorted in ascending order, while keeping the index of each random number. Next, reshape the sequence of indices into $M \times N$ cases to obtain the MP matrix. Finally, the image's pixels are permuted according to the indices in the PM. This process is detailed for 4×4 pixels in Figure 6.

(2) If the position of the pixel is impair, then the pixel's bits are permuted by cycling left shift of 2 bits:

$$\text{IP} = \text{pixel} \gg 2. \quad (8)$$

(iv) Step 4: image's pixels substitution using two different S-boxes. Here, two S-boxes are generated using the PRNG. The same idea used for PM generation is used again for the generation of S-boxes. However, the PRNG is iterated $256 + 256$ times in order to generate two streams of 256 pseudorandom 8-bit numbers $N1$ and $N2$, respectively. The pseudorandom numbers are used to create two different S-boxes. Figure 7 illustrates the process.

(v) Afterward, an internal condition is used for block's pixel permutation that permits utilizing the S-box1 or the S-box2 according to the permuted pixel obtained at the last step:

(3) If the position of the permuted pixel is pair, then the pixel is substituted by the state of the S-box1:

$$\text{PSP} = S - \text{box1}\,(\text{PP}). \quad (9)$$

(4) If the position of the permuted pixel is impair, then the pixel is substituted by the state of the S-box2:

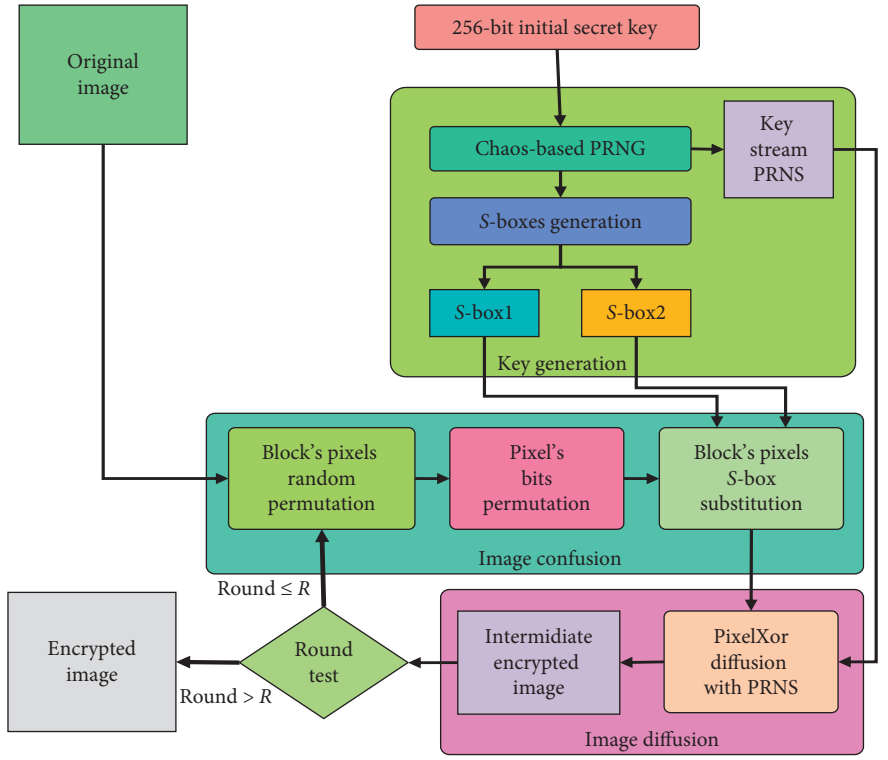


Figure 3: General architecture of the image encryption algorithm.

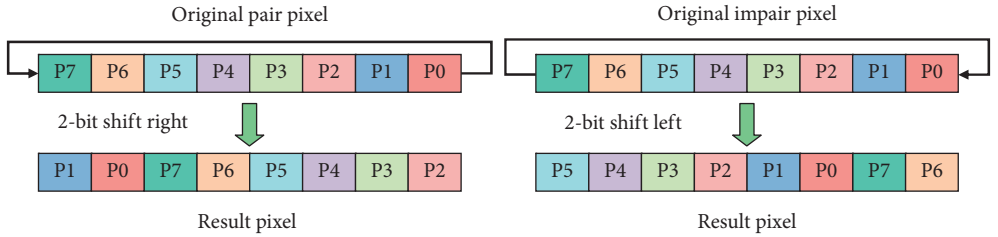


Figure 4: Process of pixel bits permutation.

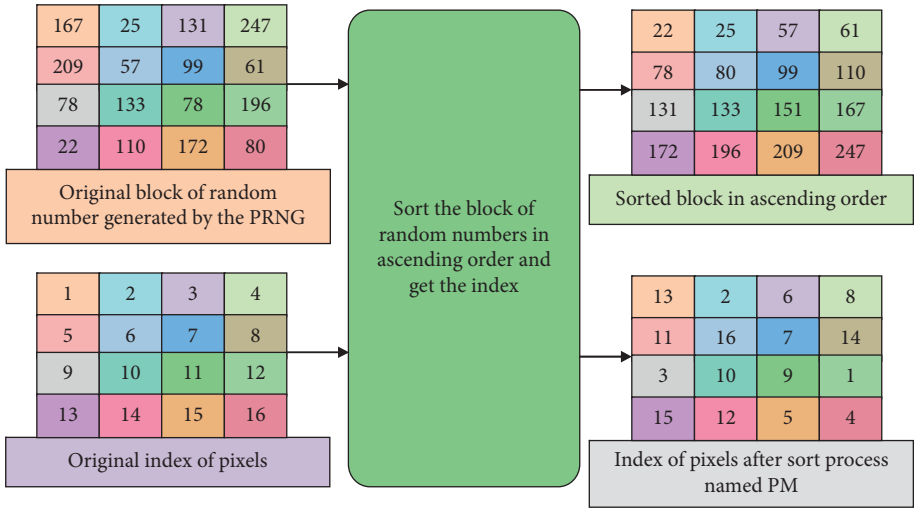


Figure 5: Generation of the PM matrix.

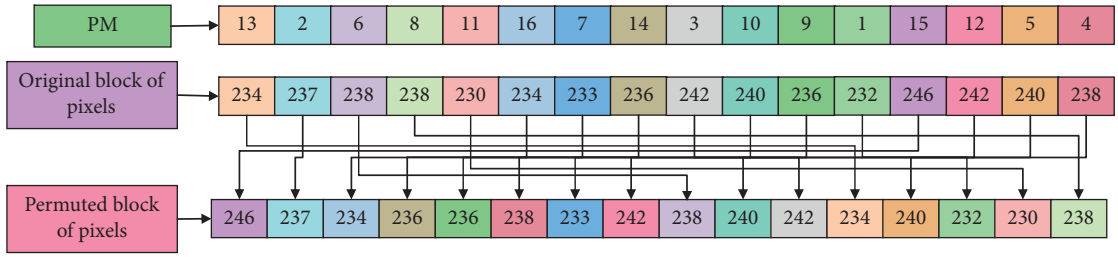


Figure 6: Process of random permutation of pixels.

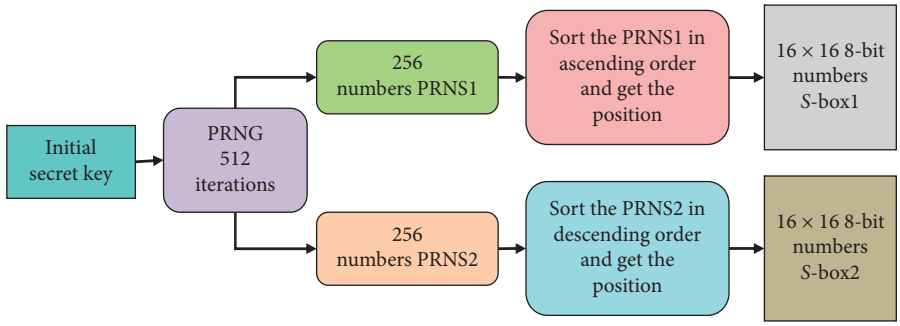


Figure 7: Process of generation of S-boxes.

$$\mathrm{PSP} = S - \mathrm{box2}\,(\mathrm{IP}). \tag{10}$$

(vi) The S-box is a 256-case substitution table. Let I be an 8-bit coded image per pixel, X and Y are binary numbers obtained from the pixel of the image as follows:

$$\begin{aligned} I_{ij} &= p_7p_6p_5p_4p_3\ p_2p_1p_0, \\ X &= p_3p_2p_1p_0, \\ Y &= p_7p_6p_5p_4. \end{aligned} \tag{11}$$

(vii) Each pixel of the image block is substituted by the state of the table which corresponds to the intersection of X with Y. Table 1 shows an example of S-box. Let us take an example: $I_{ij} = (134)_{10} = (10000110)_2$, so $X = 0110$ and $Y = 1000$; the value of the image pixel is substituted by the value of the state S.

(viii) Step 5: pixels XOR diffusion with a key stream. Thus, The PRNG is iterated again for $N \times M$ times in order to generate a key stream PRNS. N and M are the image's dimensions. Then, the obtained image in the last step is diffused by XORing the pixels with the PRNS. Following this process, an intermediate encrypted image (IEI) is obtained as follows:

$$\mathrm{IEI} = \mathrm{Block} \oplus \mathrm{PRNS}. \tag{12}$$

(ix) Step 6: repeat all last steps R rounds in order to produce the final encrypted image.

2.3. Decryption Phase. After the encryption step, the encrypted image can be stored or transmitted to a well-defined destination using an insecure network (diffusion step). At the reception, the image must be processed by the decryption system to find the plain image. The decryption system is an inverse algorithm of the encryption algorithm. In the substitution step, inverse S-boxes are used. Table 2 shows the inverse S-box of the S-box presented in Table 1.

3. Experimental Results and Interpretation

In this section, a thorough assessment of the proposed cryptosystem is detailed. Several indicators are used, which are the most used in the image cryptography community. Using the proposed cryptosystem, we can perform R rounds of encryption to improve the complexity of the encrypted image against hackers. However, we evaluate the cryptosystem with only one round of encryption. Several ordinary and medical images with different types, contents, and sizes are used for the test. For ordinary colour images, we use the standard Lena, peppers, and baboon images of size $512 \times 512 \times 3$ (Figure 8). For medical images, height different types of images are selected that are illustrated in Figure 9: medical image obtained by magnetic resonance device (MRI), 3D scanner, X-ray, radiography, endoscopy, computerized tomography (CT) scan, and ultrasound device. Simulation results and performance analysis for the selected images are given in this section. This part includes qualitative analysis of encrypted images, statistical analysis, noise and data loss analysis, key analysis, and algorithm performance analysis.

3.1. Analysis of the Encrypted Image Quality. Here, we make objective measurements of the encrypted image quality where the original image is the reference. Peak signal-to-noise ratio (PSNR) and structural similarity index measure

Table 1: *S*-box1 substitution.

S-box		*X* = 4-bit															
	X/*Y*	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Y = 4-bit	0000	137	140	253	68	34	133	135	175	12	35	120	119	232	74	78	233
	0001	41	240	31	46	69	37	127	136	79	20	184	193	207	3	24	157
	0010	36	221	73	236	57	167	144	77	118	215	220	226	249	108	156	172
	0011	255	45	165	201	248	49	198	163	58	80	189	30	242	94	237	254
	0100	17	23	114	235	247	18	152	180	15	16	44	204	100	141	224	244
	0101	158	109	149	208	81	199	42	166	39	28	105	32	54	103	150	21
	0110	256	65	178	25	203	125	187	251	112	123	145	174	6	89	87	90
	0111	138	82	111	121	241	102	104	110	47	211	106	214	86	98	117	197
	1000	61	107	179	92	188	128	22	75	142	146	246	195	33	93	225	9
	1001	76	143	162	238	181	53	116	200	229	59	52	63	239	139	177	202
	1010	124	194	96	132	51	161	252	155	38	192	50	29	67	1	56	97
	1011	147	227	115	206	148	216	62	186	40	70	171	230	2	151	48	173
	1100	13	223	55	130	219	72	85	185	218	183	213	14	7	83	191	228
	1101	153	169	4	101	205	26	126	10	160	84	182	250	210	60	5	99
	1110	245	11	88	209	243	64	134	190	19	131	95	234	170	212	27	66
	1111	159	164	231	196	43	71	176	129	217	122	91	8	222	154	168	113

Table 2: Inverse *S*-box1.

S-box		*X* = 4-bit															
	X/*Y*	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Y = 4-bit	0000	173	188	29	210	222	108	204	251	143	215	225	8	192	203	72	73
	0001	64	69	232	25	95	134	65	30	99	213	238	89	171	59	18	91
	0010	140	4	9	32	21	168	88	184	16	86	244	74	49	19	120	190
	0011	53	170	164	154	149	92	194	174	36	56	153	221	128	182	155	229
	0100	97	239	172	3	20	185	245	197	34	13	135	144	39	14	24	57
	0101	84	113	205	217	198	124	110	226	109	111	250	131	141	61	234	162
	0110	175	125	223	76	211	117	93	118	90	122	129	45	81	119	114	104
	0111	255	66	178	150	126	40	11	10	115	249	105	160	101	214	22	133
	1000	247	195	233	163	5	230	6	23	0	112	157	1	77	136	145	38
	1001	106	137	176	180	82	94	189	70	208	253	167	46	31	80	240	216
	1010	165	146	55	241	50	87	37	254	209	236	186	47	191	107	7	246
	1011	158	98	130	71	148	218	201	26	199	183	102	132	58	231	206	169
	1100	27	161	139	243	127	54	85	151	51	159	100	75	212	179	28	83
	1101	227	220	121	237	202	123	41	181	248	200	196	42	33	252	193	78
	1110	142	43	177	207	152	187	242	12	15	235	67	35	62	147	156	17
	1111	116	60	228	79	224	138	68	52	44	219	103	166	2	63	48	96

(SSIM) are used for that [16–18]. Table 3 introduces the simulation results found for each encrypted image.

From Table 3, the PSNR value of encrypted images is lower than 8 dB, and the SSIM value is close to 0. This indicates that the encrypted image produced by the proposed cryptosystem has a very poor quality. As a result, it is very difficult to predict the plain image from the encrypted one.

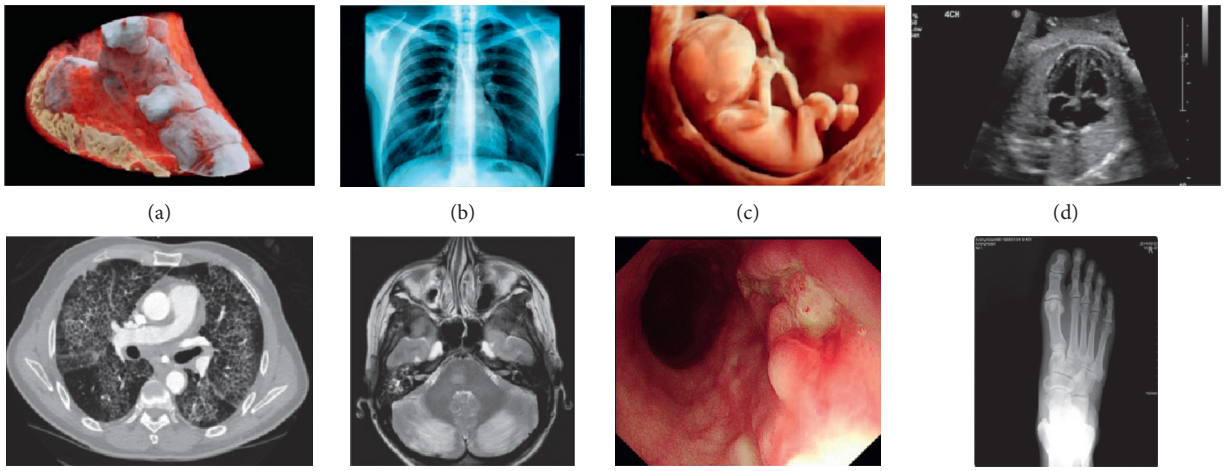
3.2. Statistical Analysis. The statistical analysis of the plain and encrypted image includes the analysis of histogram, entropy, two-dimensional normalized correlation (NC), and the correlation coefficient (ρ) [19–21].

3.2.1. Histogram Analysis. The image histogram is a two-dimensional statistical curve showing the distribution of gray scales according to their values. Figure 10 shows the original images and their corresponding encrypted images, and histograms of the original images and their corresponding encrypted images.

As seen in Figure 10.4, Figure 10.8, Figure 10.12, Figure 10.16, Figure 10.20, Figure 10.24, Figure 10.28, Figure 10.32, Figure 10.36, Figure 10.40, and Figure 10.44, we note that the histogram of the resultant encrypted image is uniformly distributed and dissimilar compared to the histogram of the original image in Figure 10.2, Figure 10.6, Figure 10.10, Figure 10.14, Figure 10.18, Figure 10.22, Figure 10.26, Figure 10.30, Figure 10.34, Figure 10.38, and Figure 10.42 which contains large spikes. Therefore, the original image's pixels and the encrypted image's pixels are completely different.

(a) (b) (c)

Figure 8: Standard Lena, peppers, and baboon images used for the test. (a) Colour Lena.jpg ($512 \times 512 \times 3$). (b) Peppers.jpg ($512 \times 512 \times 3$). (c) Colour baboon.jpg ($512 \times 512 \times 3$).



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 9: Eight different medical images selected for the test: (a) 3D medical scanner ankle ($1080 \times 1920 \times 3$); (b) 3D X-ray chest image ($3816 \times 2832 \times 3$); (c) 3D ultrasound baby ($625 \times 410 \times 3$); (d) 1D ultrasound ($1200 \times 700 \times 1$); (e) 3D CT-scan chest image ($800 \times 600 \times 3$); (f) 1D MRI ($456 \times 456 \times 1$); (g) 1D endoscopy ($634 \times 549 \times 1$); (h) 3D radiography foot ($2400 \times 2956 \times 3$).

Table 3: PSNR and SSIM values of the encrypted image.

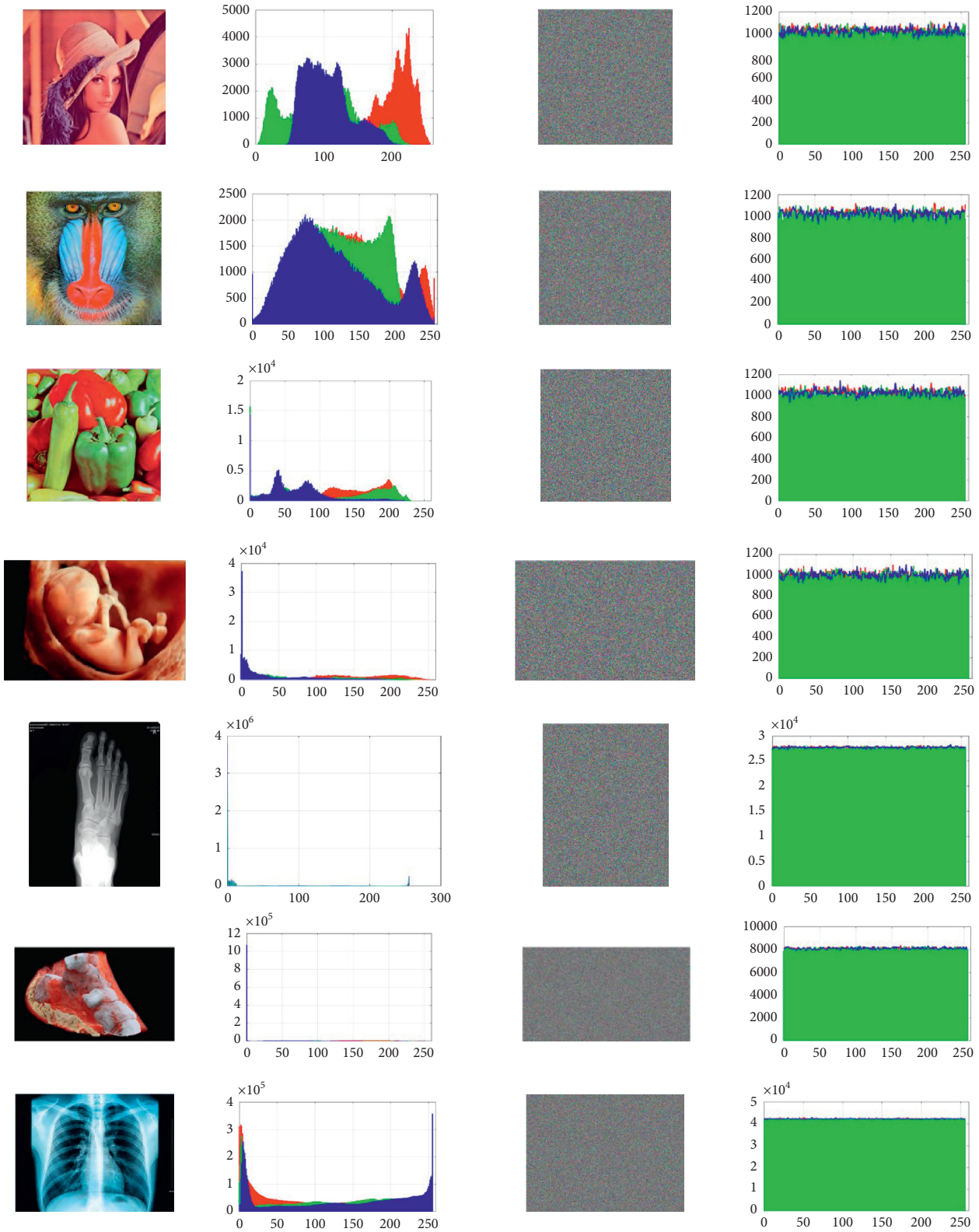
Image	PSNR	SSIM
Lena	7.6051	0.0059
Peppers	6.1244	0.0080
Baboon	7.0725	0.0061
3D ultrasound	7.5974	0.0064
1D ultrasound	7.1465	0.0072
3D scanner	6.1943	0.0040
3D radiography foot	6.4712	0.0063
3D X-ray	6.9839	0.0065
3D CT-scan	7.4483	0.0072
1D RMI	6.7421	0.0054
1D endoscopy	7.3805	0.0075

3.2.2. NC Analysis. The 2D NC is a measure of the degree of reliability between two images. After encrypting the original images, the NC is computed between the original image and its corresponding encrypted image. From Table 4, the NC value between the original image and its corresponding encrypted image produced by the proposed system is highly close to zeros. This indicates that the original image and the encrypted one are dissimilar and have not a relationship. As a consequence, the proposed system is safe against statistical attacks.

3.2.3. Global and Local Shannon Entropy Analysis. Shannon entropy is a measure of the degree of randomness associated with an image. It is defined as follows:

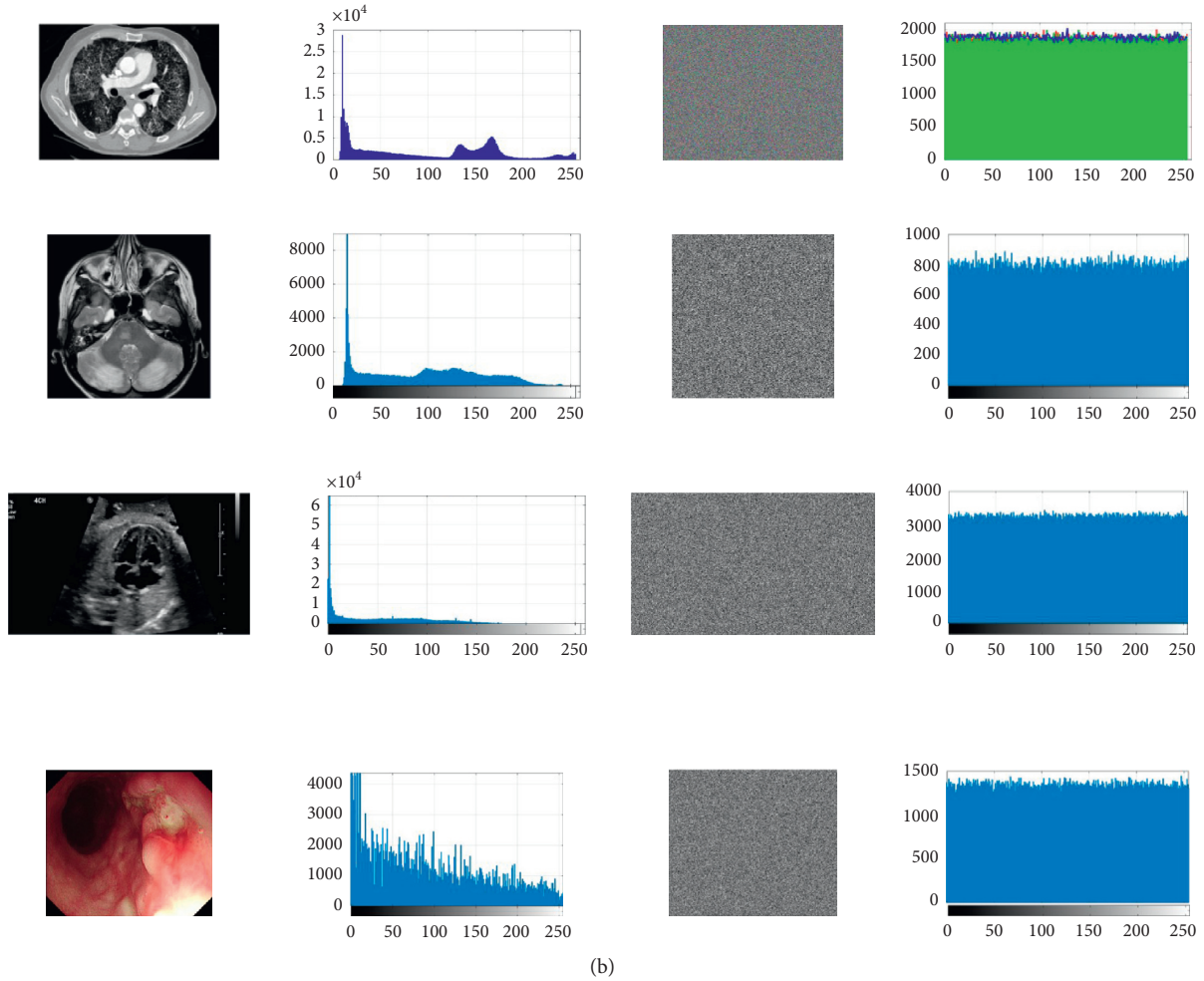
$$E(I) = \sum_{i=1}^{N} \left[P(I_i)\log_2\left(P(I_i)\right)\right]. \quad (13)$$

The global Shannon entropy is measured by applying equation (16) to the whole image. This way fails to measure the real degree of randomness of an image. It has many weaknesses such as unfair random comparisons between images of different sizes, the inability to discern the



(a)

Figure 10: Continued.



(b)

Figure 10: Histogram of the original images and their corresponding encrypted images.

randomness of images before and after image encrypting, and possible inaccurate scores for the synthesized images. However, it cannot be used for universal measures of randomness. To overcome this problem, local Shannon should be applied. The local entropy is measured by computing the mean of global Shannon entropies over all the nonoverlapping blocks of size 1936 pixels in the image [3]. Table 5 introduces the simulation results of global and local Shannon entropy found for each image.

Analysing the results, the encrypted image's global entropy value is highly close to the ideal value 8 and the mean of local entropy is very important. This indicates that the pixels of the encrypted image are random. As a consequence, the proposed system is safe against entropy and statistical attacks. Table 6 introduces a comparative study of image entropy with several other recent works. The proposed system gives the best result.

3.2.4. Correlation Coefficient Analysis. The ρ tool computes the correlation coefficient in the horizontal, vertical, and diagonal directions of an image. Let x and y two grayscale values of two adjacent pixels in the image, and the correlation of the adjacent pixels is computed using equation (14):

$$\begin{aligned} \rho_{xy} &= \frac{\operatorname{cov}(x, y)}{\sqrt{D(x)}\sqrt{D(y)}}, \\ D(x) &= \frac{1}{N}\sum_{i=1}^{N}\left(x_i - E(x_i)\right)^2, \\ \operatorname{cov}(x, y) &= \frac{1}{N}\sum_{i=1}^{N}\left(x_i - E(x)\left(y_i - E(y)\right)\right), \\ E(x) &= \frac{1}{N}\sum_{i=1}^{N} x_i, \end{aligned} \tag{14}$$

where $E(x)$ is the expectation of x, $D(x)$ is the estimation of the variance in x, and $\operatorname{cov}(x, y)$ is the estimation of the covariance between x and y.

Table 7 shows that the correlation coefficient of the original images is close to 1, while the encrypted images are close to zeros. This indicates that the original image's pixels are correlated, unlike the encrypted image's pixels are not correlated. Figure 11 shows the distributions of 3000 pairs of randomly selected adjacent pixels of the original image Lena in the horizontal, vertical, and diagonal directions,

Table 4: NC results of encrypted images.

Image	NC		
	Red	Blue	Green
Lena	−0.0041	−0.0024	−0.00065
Peppers	−0.0025	−0.00018	−0.00079
Baboon	−0.0027	−0.0028	−0.00063
3D ultrasound	0.00003	−0.00021	−0.00029
3D ankle	−0.00009	0.00003	−0.00024
3D *X*-ray	−0.00041	−0.00042	−0.0015
3D radiography	−0.0039	−0.00367	−0.0043
3D CT-scan	−0.0028	−0.00056	−0.00073
1D ultrasound		−0.0047	
MRI		−0.0036	
Endoscopy		−0.00042	

Table 5: Global and local Shannon entropy values of encrypted images.

Image	Local Shannon entropy			Global Shannon entropy		
	Red	Blue	Green	Red	Blue	Green
Lena	7.9548	7.9546	7.9542	7.9998	7.9998	7.9997
Peppers	7.9546	7.9549	7.9544	7.9998	7.9997	7.9997
Baboon	7.9554	7.9557	7.9553	7.9998	7.9998	7.9998
3D ultrasound	7.9557	7.9552	7.9548	8.0000	8.0000	8.0000
3D ankle	7.9563	7.9559	7.9559	8.0000	8.0000	8.0000
3D X-ray	7.9568	7.9563	7.9564	8.0000	8.0000	8.0000
3D radiography	7.9562	7.9563	7.9560	8.0000	8.0000	8.0000
3D CT-scan	7.9567	7.9567	7.9565	8.0000	8.0000	8.0000
1D ultrasound		7.9556			8.0000	
MRI		7.9553			7.99998	
Endoscopy		7.9549			7.99998	

Table 6: Comparative study of entropy values.

Image	Ordinary image		Medical image	
Entropy	Global	Local	Global	Local
Reference [3]	—	—	7.99950	7.90300
Reference [5]	—	—	7.99930	—
Reference [6]	—	—	7.99954	—
Reference [7]	7.99930	—	—	—
Reference [8]	7.98910	—	—	—
Reference [9]	7.99932	—	—	—
Reference [10]	7.98830	—	—	—
Reference [11]	7.99930	—	—	—
Reference [12]	7.99720	—	—	—
Reference [22]	—	—	7.99740	—
Proposed algorithm	7.99985	7.95486	7.99998	7.95627

respectively. Also, Figure 12 presents the distribution of pixels of the original medical image ankle. Figure 13 shows the distributions of 2000 pairs of randomly selected adjacent pixels of the encrypted Lena image in the horizontal, vertical, and diagonal directions, respectively. Also, Figure 14 presents the distribution of pixels of the encrypted medical image ankle. We note that the distribution of the pixels of the plain image is consistent, unlike the encrypted image is inconsistent. This indicates that the encrypted image's pixels are not correlated. As a result, we cannot predict any information using a statistical correlation between the encrypted image pixels that permits recovering of the original image.

3.3. Noise and Data Loss Analysis. Using digital networks for transmission, image is vulnerable to several types of noise and loss. However, having any noise or loss in the encrypted image can result in difficulty to recover the clear image using the decryption algorithm. Noise and loss refer to random errors in pixels values of the image acquired during image transmission. A good cryptosystem algorithm should recover the plain image when the encrypted image was affected by any treatment. In this part, we evaluate the robustness of the proposed cryptosystem against Gaussian white noise and "salt and pepper" data loss. Firstly, we produce an encrypted image using the encryption system. Then, we attack it with an attack which results in a modified encrypted image. Afterward, we try decrypting the modified encrypted image by the decryption system. Finally, we evaluate the decrypted image using the NC, PSNR, and SSIM tools where the original image is the reference [23–27]. The selected attacks are the most common and have been used with different intensities to properly test the robustness of the algorithm.

Salt and pepper noise is added to an image by the addition of both random on and off pixels, i.e., random bright with a pixel value of 255 and random dark with 0 pixel value, all over the image. Table 8 introduces the simulation results of the PSNR, SSIM, and NC where the encrypted image was affected by "salt and pepper" data loss. Figures 15(a)–15(d) show the encrypted images attacked with "salt and pepper" loss intensities 0.01, 0.05, 0.01, and 0.5, respectively. Figures 15(e)–15(h) show the corresponding decrypted images, respectively. An intensity of 0.5, i.e., 50% of the encrypted image pixels, has been lost.

Table 9 introduces the result of PSNR, SSIM, and NC where the encrypted image is noisy with the Gaussian white noise. Figures 16(a)–16(d) are the encrypted images noisy with the variances 0.01, 0.05, 0.1, and 0.5, respectively. Figures 16(e)–16(h) show the corresponding decrypted image, respectively.

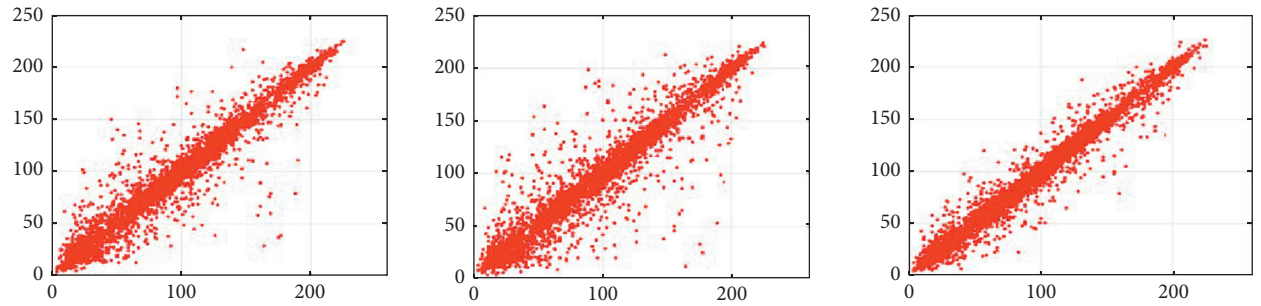
Following the obtained results, the proposed algorithm proves its performance to a certain extent. This is due to the main feature that our algorithm does not allow any propagation error.

3.4. Known-Plaintext and Chosen-Plaintext Attack Analysis. In the proposed algorithm, the diffusion process is performed by the XOR operation. Thus, it is very essential to evaluate its robustness against the chosen-plaintext attack. This type of attack uses the encrypted image with arbitrary plaintext data to crack the cryptosystem algorithm. According to reference [28], if equation (15) is determined, the algorithm will be vulnerable to chosen-plaintext attacks. Otherwise, the algorithm resists chosen-plaintext attacks:

$$C_1(x, y) \oplus C_2(x, y) = P_1(x, y) \oplus P_2(x, y). \quad (15)$$

Table 7: ρ values of the original image and its corresponding encrypted image.

Image	Direction	Original image			Encrypted image		
		Red	Blue	Green	Red	Blue	Green
Lena (512 × 512 × 3)	H	0.9588	0.9358	0.9160	−0.0196	−0.0145	−0.0546
	V	0.9818	0.9665	0.9522	−0.0162	−0.0184	−0.0082
	D	0.9900	0.9810	0.9737	−0.0078	−0.0012	−0.0049
Peppers (512 × 512 × 3)	H	0.9617	0.9658	0.9443	−0.0136	−0.0070	−0.0014
	V	0.9814	0.9789	0.9671	−0.0288	−0.0054	0.0046
	D	0.9666	0.9655	0.9455	−0.0310	−0.0294	0.0084
Baboon (512 × 512 × 3)	H	0.8656	0.9291	0.8399	0.0053	0.0063	−0.0040
	V	0.7897	0.8848	0.7629	0.0020	0.0076	0.0098
	D	0.8855	0.9309	0.8619	0.0077	0.0049	−0.0075
3D ultrasound baby (2844 × 4044 × 3)	H	0.8989	0.9008	0.8360	−0.0022	−0.0024	−0.0020
	V	0.9720	0.9723	0.9559	−0.0056	−0.0090	−0.0013
	D	0.9789	0.9815	0.9662	−0.0019	−0.0082	−0.0002
3D scanner ankle (1080 × 1920 × 3)	H	0.9981	0.9983	0.9963	−0.0073	−0.0187	−0.0100
	V	0.9975	0.9984	0.9956	−0.0319	−0.0259	−0.0061
	D	0.9993	0.9992	0.9984	−0.0024	−0.0051	−0.0179
3D radiography (2400 × 2956 × 3)	H	0.9993	0.9994	0.9994	−0.0022	−0.0018	−0.0007
	V	0.9989	0.9916	0.9945	−0.0029	−0.0036	−0.0032
	D	0.9946	0.9944	0.9901	−0.0015	−0.0019	−0.0002
3D X-ray (3816 × 2832 × 3)	H	0.9981	0.9971	0.9955	−0.0241	−0.0051	−0.0164
	V	0.9995	0.9990	0.9986	−0.0071	−0.0034	−0.0180
	D	0.9995	0.9990	0.9987	−0.0043	−0.0108	−0.0336
3D CT-scan (800 × 600 × 3)	H	0.9946	0.9944	0.9901	−0.0018	−0.0066	−0.0011
	V	0.9946	0.9944	0.9891	−0.0012	−0.0033	−0.0039
	D	0.9944	0.9941	0.9897	−0.0046	−0.0018	−0.0026
1D ultrasound (1200 × 700 × 1)	H		0.9865			−0.00012	
	V		0.9843			−0.00084	
	D		0.9839			−0.00069	
MRI (456 × 456 × 1)	H		0.9144			−0.0022	
	V		0.9346			−0.0041	
	D		0.99422			−0.0019	
Endoscopy (634 × 549 × 1)	H		0.9882			−0.0260	
	V		0.9869			−0.0022	
	D		0.9847			−0.0047	



(a) (b) (c)

Figure 11: Continued.

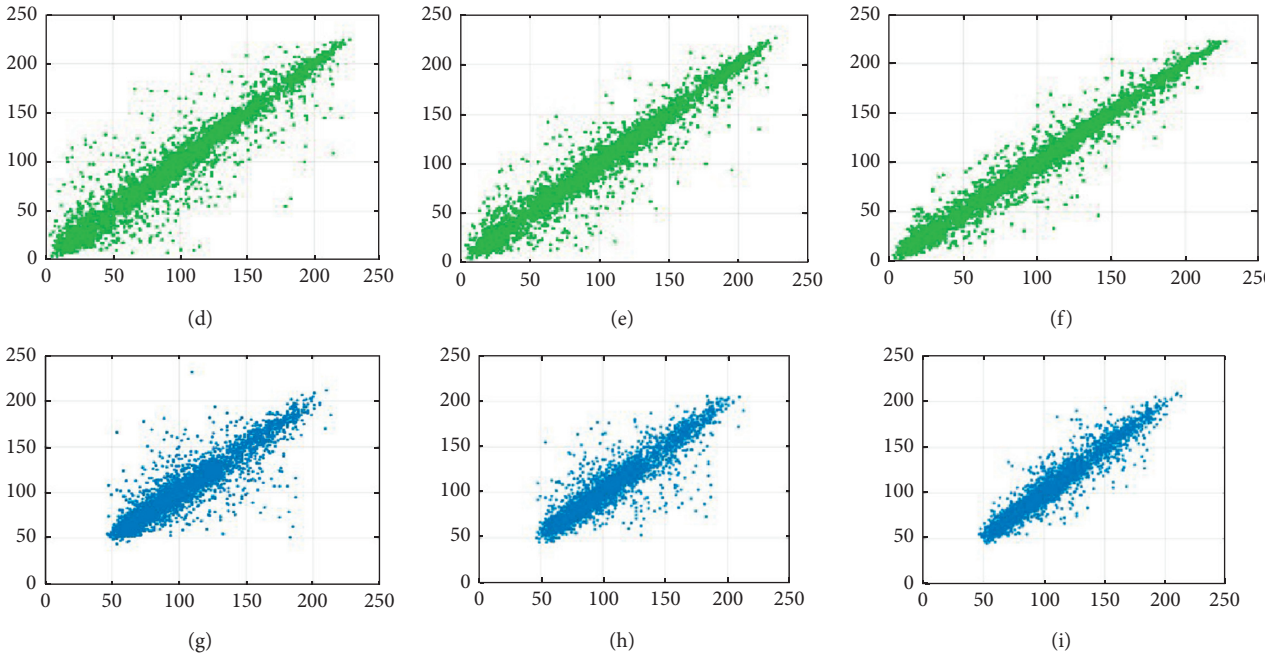


Figure 11: Distribution of 3000 pairs of randomly selected adjacent pixels of the original Lena image in the horizontal, vertical, and diagonal directions.

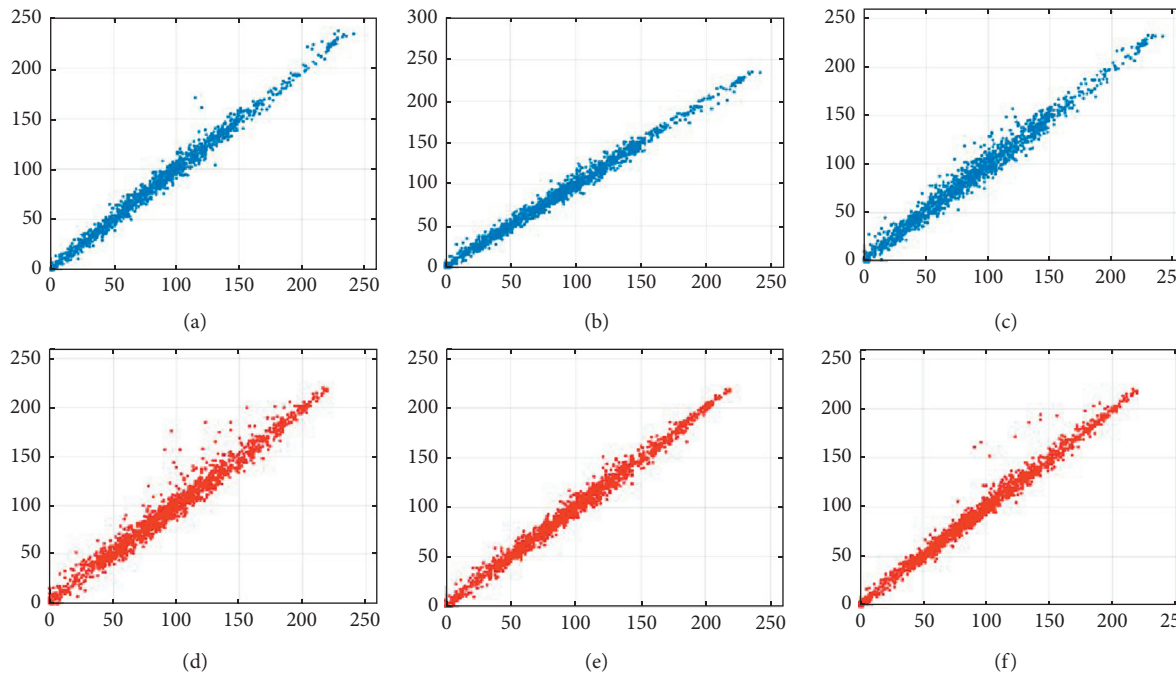


Figure 12: Continued.

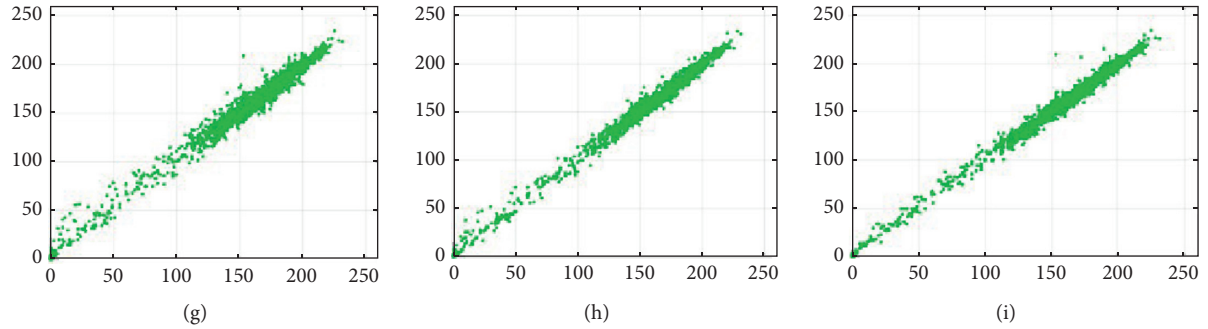


Figure 12: Distribution of 3000 pairs of randomly selected adjacent pixels of the encrypted Lena image in the horizontal, vertical, and diagonal directions.

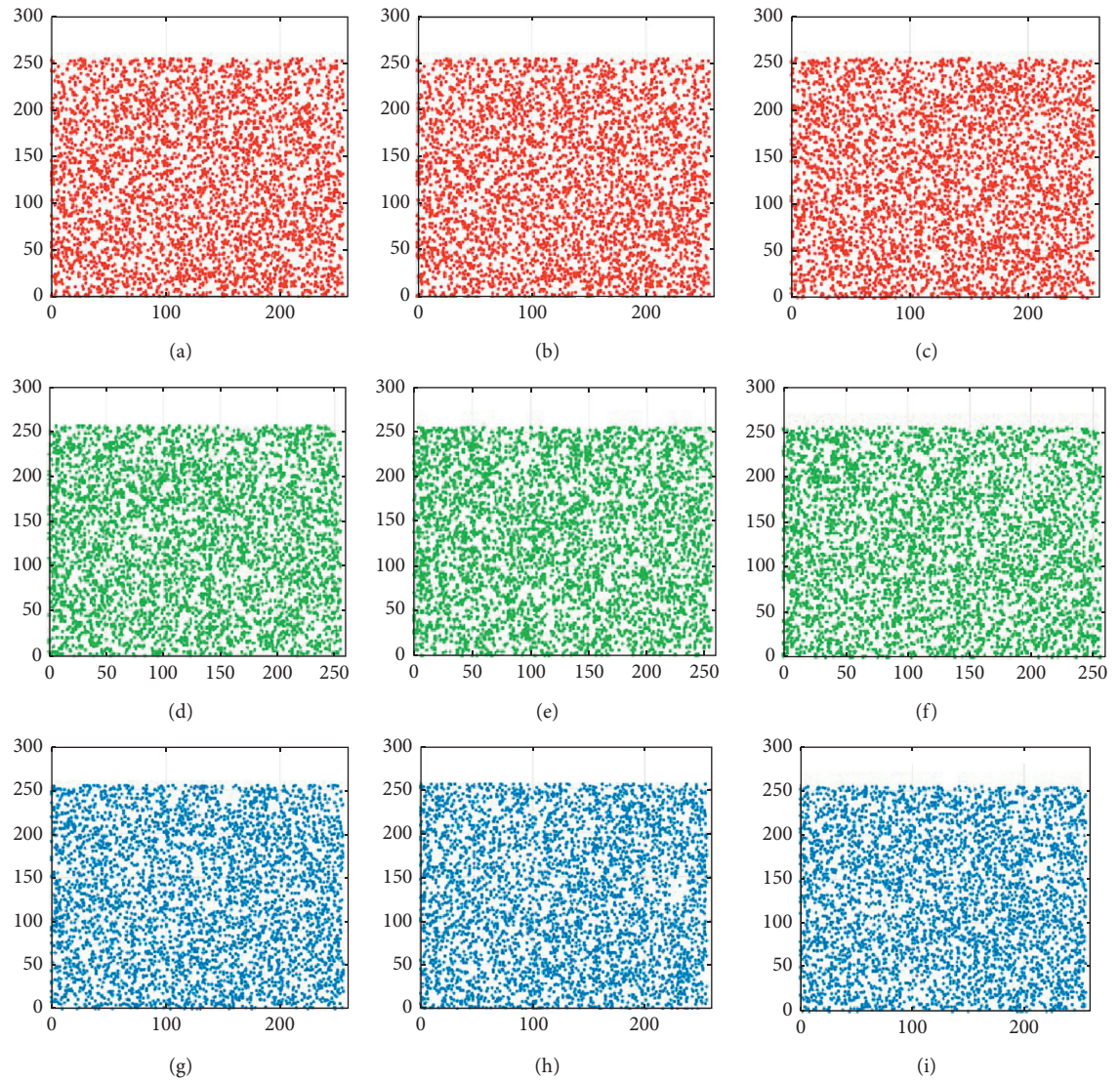


Figure 13: Distribution of 3000 pairs of randomly selected adjacent pixels of the medical encrypted ankle image in the horizontal, vertical, and diagonal directions.

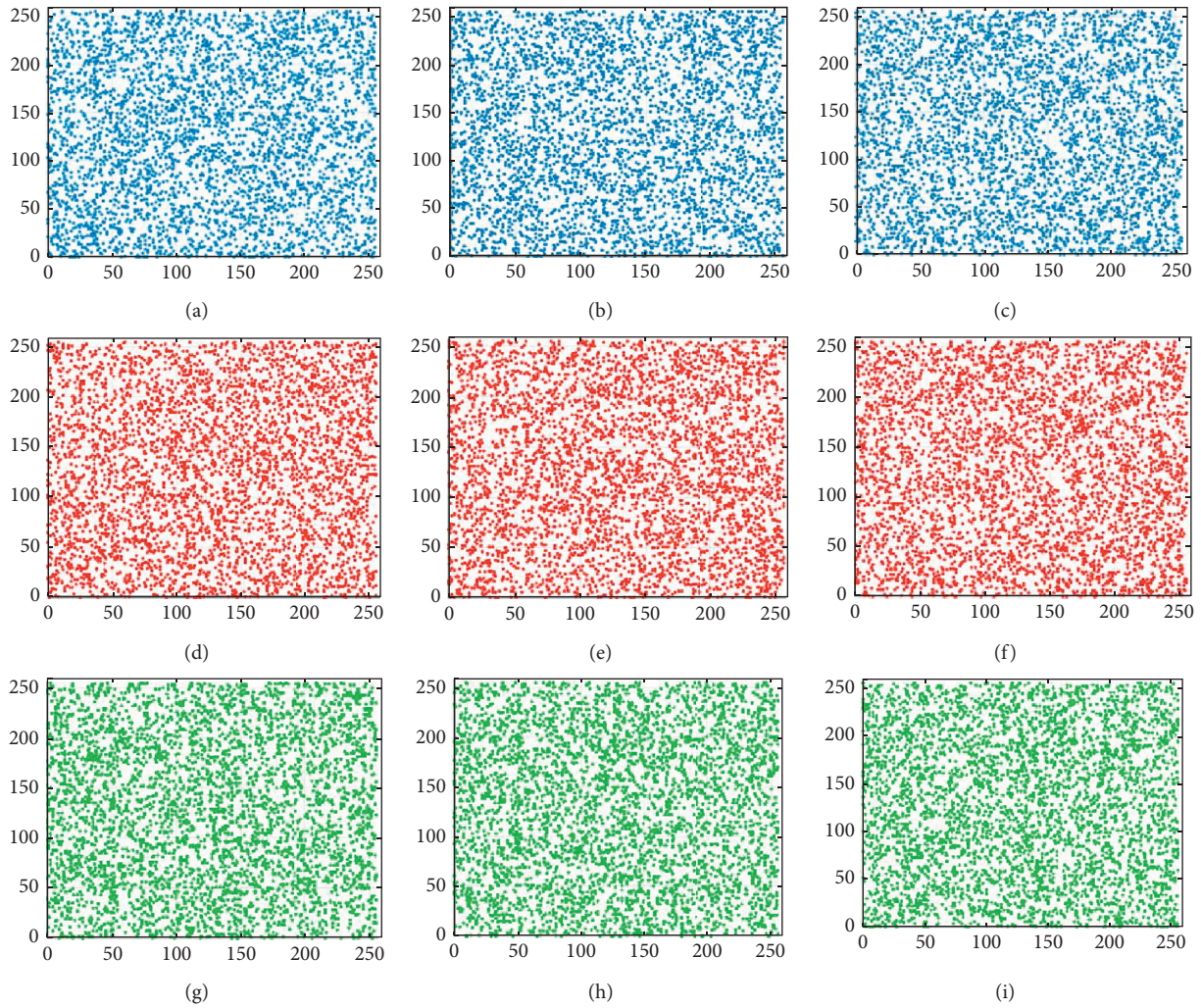


Figure 14: Distribution of 3000 pairs of randomly selected adjacent pixels of the medical original ankle image in the horizontal, vertical, and diagonal directions.

Table 8: PSNR, SSIM, and NC results between the original and the corresponding decrypted image under different intensities of salt and pepper noise.

Image	Data loss		Salt and pepper					
	Intensity		0.009	0.01	0.05	0.09	0.1	0.5
Lena $512 \times 512 \times 3$	PSNR	39.51	37.96	35.68	32.66	29.89	28.67	26.61
	SSIM	0.982	0.969	0.967	0.856	0.765	0.748	0.479
	NC	0.988	0.980	0.979	0.899	0.828	0.814	0.557
Peppers $512 \times 512 \times 3$	PSNR	39.08	37.58	35.02	31.11	29.54	28.05	26.09
	SSIM	0.983	0.969	0.967	0.857	0.771	0.751	0.489
	NC	0.990	0.982	0.980	0.906	0.840	0.826	0.564
Baboon $512 \times 512 \times 3$	PSNR	39.81	38.20	36.93	32.81	30.27	28.83	26.82
	SSIM	0.977	0.959	0.957	0.821	0.722	0.701	0.458
	NC	0.992	0.987	0.986	0.931	0.878	0.867	0.521
3D ultrasound baby $2844 \times 4044 \times 3$	PSNR	40.92	38.40	37.98	35.90	31.34	29.89	26.92
	SSIM	0.936	0.902	0.894	0.713	0.623	0.604	0.420
	NC	0.994	0.991	0.990	0.949	0.907	0.897	0.496

Table 8: Continued.

Image	Data loss		Salt and pepper					
	Intensity		0.009	0.01	0.05	0.09	0.1	0.5
3D scanner ankle 1080 × 1920 x 3	PSNR	40.26	37.69	36.21	35.17	31.65	29.21	25.20
	SSIM	0.986	0.937	0.917	0.856	0.782	0.571	0.437
	NC	0.994	0.990	0.989	0.945	0.904	0.894	0.501
3D radiography 2400 × 2956 × 3	PSNR	41.21	38.85	37.32	35.79	32.63	29.85	25.81
	SSIM	0.981	0.916	0.896	0.805	0.663	0.549	0.425
	NC	0.994	0.990	0.989	0.946	0.905	0.893	0.487
3D X-ray 3816 × 2832 × 3	PSNR	40.26	38.47	36.96	34.98	31.75	28.72	25.34
	SSIM	0.985	0.979	0.968	0.871	0.782	0.766	0.421
	NC	0.976	0.963	0.945	0.883	0.769	0.747	0.453
3D CT-scan 800 × 600 × 3	PSNR	40.64	38.35	36.45	34.66	30.74	27.76	25.17
	SSIM	0.979	0.973	0.841	0.795	0.632	0.473	0.437
	NC	0.979	0.968	0.946	0.891	0.787	0.761	0.507
1D ultrasound 1200 × 700	PSNR	39.54	37.26	36.08	32.17	28.95	27.21	25.14
	SSIM	0.987	0.942	0.909	0.867	0.831	0.782	0.587
	NC	0.993	0.989	0.953	0.851	0.704	0.694	0.501
MRI 456 × 456 × 1	PSNR	39.75	37.81	35.52	31.62	29.98	27.83	24.92
	SSIM	0.991	0.975	0.860	0.713	0.689	0.626	0.516
	NC	0.995	0.983	0.974	0.898	0.826	0.711	0.501
Endoscopy 634 × 549	PSNR	39.67	38.14	37.32	35.23	31.07	29.17	25.72
	SSIM	0.984	0.924	0.887	0.812	0.674	0.589	0.477
	NC	0.995	0.987	0.982	0.943	0.910	0.879	0.513

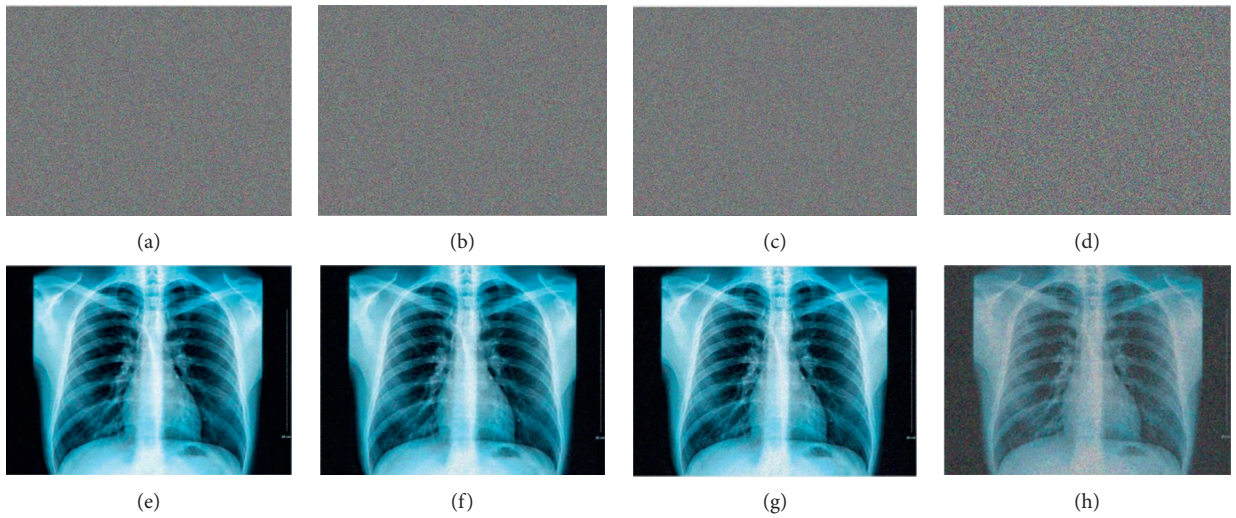


Figure 15: Result decrypted chest X-ray image under different intensities of salt and pepper data loss.

P_1 and P_2 are the plain Lena and pepper images, while C_1 and C_2 are their corresponding encrypted images, respectively. Figure 17 shows that the XOR of encrypted image and clear image is not equal, i.e., the proposed cryptosystem algorithm resists chosen-plaintext attack.

In general, an attacker uses whole black or whole white images to find out the possible patterns in the cryptosystem algorithm. However, the whole white and whole black images of $512 \times 512 \times 3$ size are encrypted by the proposed algorithm. Figure 18 presents the encrypted images and no pattern can be perceptible. The entropy value of images is selfsame as other encrypted images and correlation coefficients are highly close to zeros. Table 10 shows the result simulation of global and local image entropy and correlation coefficient.

3.5. Key Analysis. The analysis of the key includes the key space, key sensitivity, and randomness analysis test to evaluate the strength of the cryptosystem against brute-force and differential hackers.

3.5.1. Key Space. The key space of a safety encryption scheme should be very large to resist the brute-force attack. The designed PRNG has six outputs each with 32-bit length. Thus, there is a $2^{3\times64} = 2^{192}$ possible key. Following reference [29], the key brute-force attacks are computationally infeasible. Table 11 gives a comparative study of the key space with other recent encryption algorithms.

Table 9: PSNR, SSIM, and NC results between the original and the corresponding decrypted image under different variance of Gaussian white noise.

Image	Noise	Gaussian white						
	Variance	0.005	0.009	0.01	0.05	0.09	0.1	0.5
Lena 512 × 512 × 3	PSNR	38.90	37.81	35.61	33.19	29.11	28.23	23.61
	SSIM	0.906	0.837	0.824	0.725	0.658	0.548	0.409
	NC	0.947	0.893	0.781	0.719	0.695	0.663	0.465
Peppers 512 × 512 × 3	PSNR	38.75	12.72	12.54	10.21	9.56	9.45	8.17
	SSIM	0.932	0.882	0.832	0.754	0.622	0.541	0.415
	NC	0.954	0.889	0.847	0.783	0.637	0.515	0.457
Baboon 512 × 512 × 3	PSNR	39.19	38.15	36.07	33.83	29.75	28.54	23.77
	SSIM	0.922	0.876	0.763	0.604	0.571	0.530	0.405
	NC	0.983	0.891	0.800	0.751	0.540	0.508	0.443
3D ultrasound baby 2844 × 4044 × 3	PSNR	39.43	38.53	36.10	33.71	29.02	28.90	23.82
	SSIM	0.971	0.877	0.849	0.732	0.650	0.635	0.412
	NC	0.982	0.925	0.872	0.747	0.636	0.622	0.397
3D scanner ankle 1080 × 1920 x 3	PSNR	39.72	38.13	37.81	32.76	29.15	27.42	23.85
	SSIM	0.981	0.893	0.867	0.759	0.673	0.648	0.415
	NC	0.996	0.953	0.891	0.756	0.647	0.631	0.421
3D chest radiography 2400 × 2956 × 3	PSNR	39.86	38.57	36.22	33.72	29.33	28.94	24.03
	SSIM	0.981	0.895	0.862	0.743	0.659	0.643	0.425
	NC	0.993	0.936	0.895	0.786	0.671	0.655	0.443
3D X-ray 3816 × 2832 × 3	PSNR	39.59	38.42	36.86	33.61	29.22	28.47	23.69
	SSIM	0.982	0.879	0.850	0.737	0.656	0.641	0.419
	NC	0.989	0.981	0.864	0.791	0.677	0.669	0.453
3D CT-scan 800 × 600 × 3	PSNR	38.78	38.05	35.91	33.11	29.23	28.15	23.39
	SSIM	0.975	0.853	0.824	0.718	0.629	0.631	0.402
	NC	0.982	0.968	0.837	0.745	0.653	0.646	0.420
1D ultrasound 1200 × 700	PSNR	39.14	38.42	35.98	33.47	28.87	27.97	23.52
	SSIM	0.972	0.846	0.815	0.698	0.609	0.522	0.389
	NC	0.980	0.959	0.828	0.719	0.627	0.569	0.390
MRI 456 × 456 × 1	PSNR	38.72	37.85	35.21	32.71	28.18	27.46	23.31
	SSIM	0.973	0.851	0.809	0.692	0.589	0.519	0.382
	NC	0.986	0.965	0.829	0.714	0.631	0.573	0.387
Endoscopy 634 × 549	PSNR	38.68	37.87	35.25	32.68	28.12	27.37	23.22
	SSIM	0.975	0.846	0.811	0.697	0.585	0.5116	0.379
	NC	0.984	0.960	0.826	0.705	0.629	0.569	0.386

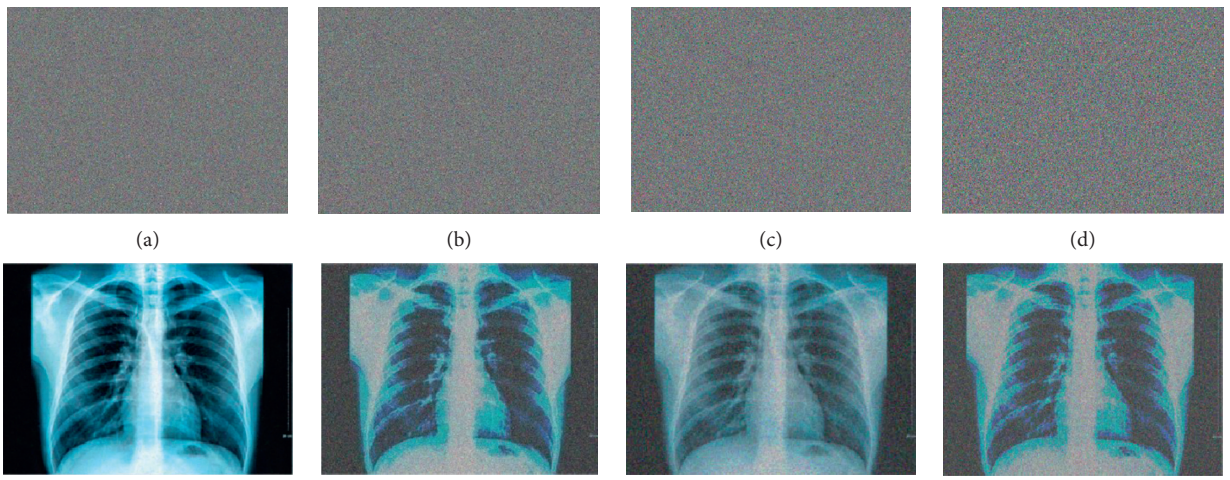
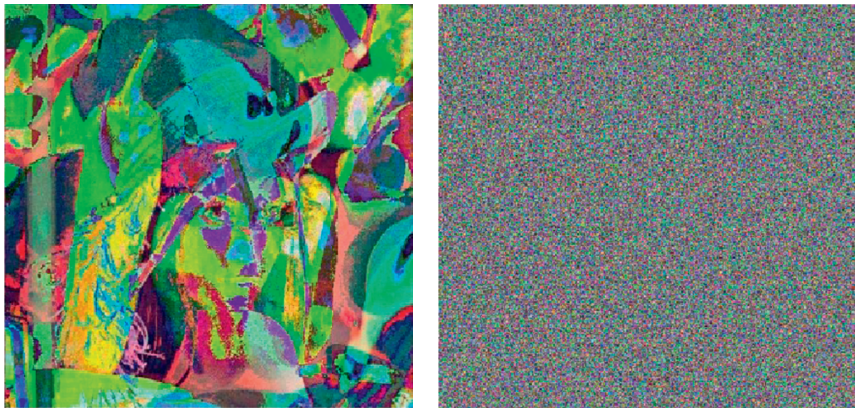
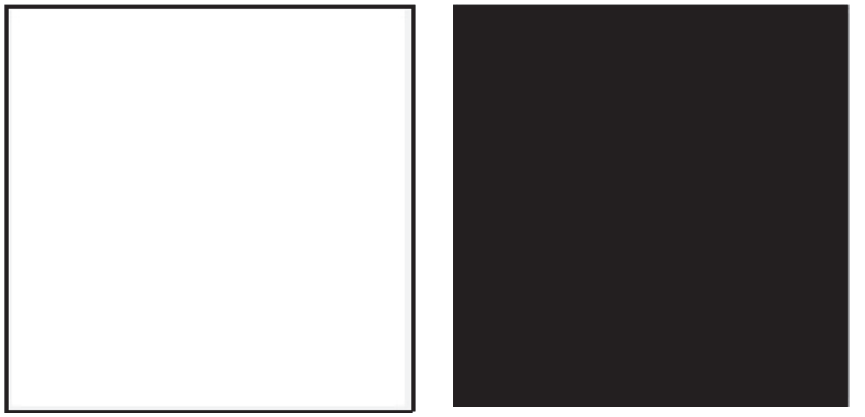


Figure 16: Result decrypted chest X-ray image under different variances of Gaussian white noise.



(a) (b)

Figure 17: Chosen-plaintext analysis. (a) P_1 XOR P_2. (b) C_1 XOR C_2.



(a) (b)

(c) (d)

Figure 18: (a) Clear white image. (b) Clear black image. (c) Encrypted white image. (d) Encrypted black image.

3.5.2. Key Sensitivity. The PRNG should be sensitive to a small change in the initial key *ki*. Exactly, a change of 1 bit in *ki* will cause a considerably significant change in the encrypted image. The sensitivity test is applied at the encryption phase, as much as at the decryption phase. The sensitivity test can be achieved using the NPCR and UACI tests to assess the robustness of the encrypted image against differential hackers [30]. NPCR and UACI are described as follows:

$$\begin{aligned} \text{NPCR} &= \frac{1}{S}\sum D(i,j) \times 100\%, \\ \text{UACI} &= \frac{1}{S}\sum \frac{|d|}{G} \times 100\%. \end{aligned} \tag{16}$$

Table 10: Entropy value of encrypted black and white images.

Tool	Entropy		Correlation coefficient		
	Global	Local	H	V	D
Plain black image	0	0	—	—	—
Encrypted black image	7.9998	7.9574	−0.0023	−0.0042	−0.00354
Plain white image	0	0	—	—	—
Encrypted white image	7.9998	7.9574	−0.0035	−0.0029	−0.00173

Table 11: Comparative study of key space.

Work	Key space
Reference [4]	2^{150}
Reference [8]	10^{42}
Reference [10]	3.4×10^{38}
Proposed algorithm	2^{192}

where S is the size of the image and $D(i, j)$ is a logical value affected by the following cases:

$$D(i,j) = \begin{cases} 0, & \text{if } I_1(i,j) = I_2(i,j), \\ 1, & \text{if } I_1(i,j) \neq I_2(i,j). \end{cases} \quad (17)$$

d is the difference between two pixels on the image with the same coordinates:

$$d = p_1(i,j) - p_2(i,j). \quad (18)$$

Encryption Phase: in this phase, a change of one bit in *ki* must provide a considerable change in the encrypted image. For the test, let us consider two initial secret keys *ki*1 and *ki*2, where *ki*2 is different by one bit from *ki*1:

(i) *ki*1 = 5C2D5DA1B3B91F884A20FC7E18C644C2ED 4EA2F05D2DEBD98A14E20906E4C1CD

(ii) *ki*2 = 6C2D5DA1B3B91F884A20FC7E18C644C2ED 4EA2F05D2DEBD98A14E20906E4C1CD

Therefore, we encrypt the same Lena image using the *ki*1 and the *ki*2, respectively. Figures 19(a) and 19(b) show the result images of each key, respectively. Figure 19(c) shows the difference between image Figures 19(a) and 19(b). Table 12 gives the simulation results of the NPCR and UACI values found between the two encrypted images.

Like the aforementioned results, NPCR and UACI percentages are important. In addition, the NC coefficient is very weak; i.e., the images are dissimilar. We conclude that the proposed cryptosystem is highly sensitive to a one-bit change in the given initial key.

Decryption Phase: at the decryption step, a change of one bit in *ki* must provide a considerable change in the decrypted image. For the test, *ki*1 is used to decrypt the image in Figure 15(b); i.e., we try to decrypt an encrypted image by a wrong key which is different by one bit from the right key. Figure 20 shows the result decrypted image.

The NC value between the original and the image in Figure 20 is close to zero, NC = −0.0037. This indicates that the recovered image and the original image are completely different and have not a relationship. As a consequence, it is impossible to recover the original image using the wrong key which is different by one bit from the right key.

3.5.3. NIST 800-22 Test. The analysis of the randomness of a key stream can be achieved using the NIST 800-22 test. The test is useful to test random and pseudorandom number generators to determine whether or not a PRNG is appropriate for data encryption [31]. The analysis contains 15 tests that assess key streams to meet important necessities. It focuses on different nonrandom aspects that can be found in a key sequence.

The test results of a sequence of 262400 bytes generated by the proposed PRNG-CTR are shown in Table 13. The sequence passes successfully all parts of the test. This demonstrates that the generated pseudorandom numbers have good statistical properties: unpredictable, random, independent, and uniformly distributed.

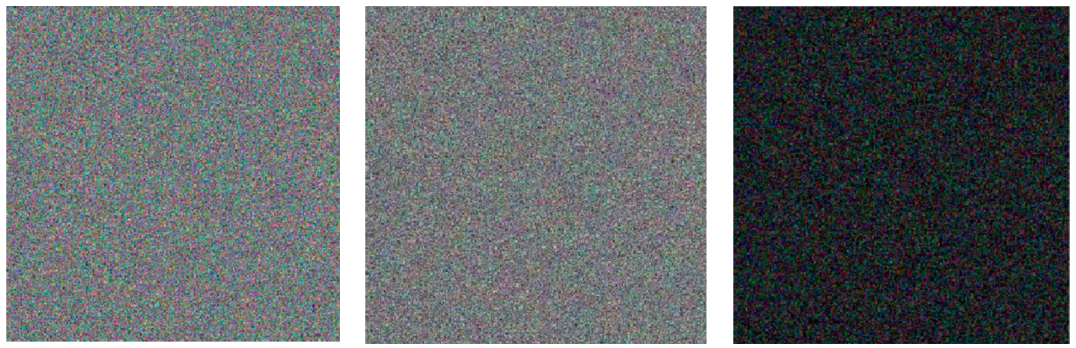
3.6. Cryptosystem Performance and Discussion. In real-time image processing, the execution time is a major constraint. In a software implementation, the speed of execution mainly depends on CPU performance. The proposed algorithm is implemented using the Matlab R2017a software running on a personal computer with CPU Intel Core i7-3770 3.4 GHz frequency. We can use the approximate equations (19) and (20) to compute the speed (S) and the number of cycles per byte (CpB) taken by an encryption algorithm running on a specific processor [32]:

$$S = \frac{\mathrm{DS}}{T} \frac{\mathrm{MB}}{s}, \quad (19)$$

$$\mathrm{CpB} = \frac{\mathrm{CpS}}{S}, \quad (20)$$

where DS is the data size, T is the time taken to execute the algorithm on a CPU, and CpS is the CPU frequency.

The proposed cryptosystem executes four processes in each encryption round: pixel's bit permutation, random permutation of pixel's position, S-box substitution of pixels, and XOR diffusion. However, it uses two permutation processes, one substitution process, and one diffusion process. Each process takes a time that to be executed. In Table 14, we have introduced the time taken by each process to encrypt the colour Lena image of size $512 \times 512 \times 3$ that presents 786432 megabytes of volume. Analysing the results, we note that the permutation process takes more time than substitution, while the XOR diffusion takes less time. The XOR process is a simple operation that can be done in parallel and pipeline processing. We suggest comparing our work to the works presented in reference [5–7, 10, 11]. To properly compare the systems, we introduce the comparison Table 15 that remembers the time taken by an encryption algorithm according to the number of processes. From reference [7, 10], the designers employ only the XOR diffusion process. However, it is logical that their cryptosystems take less execution time than other more complex architecture, but their algorithms cannot permit secure encryption according to the Shannon theory [33]. From reference



(a) (b) (c)

Figure 19: Key sensitivity test applied at the encryption phase.

Table 12: Simulation results of the NPCR and UACI test.

Image	NPCR (%)			UACI (%)		
	Red	Blue	Green	Red	Blue	Green
Lena	99.68109	99.69172	99.68452	33.83002	34.03204	33.69463
Peppers	99.75642	99.70531	99.69347	33.68726	33.59820	33.65629
Baboon	99.74926	99.69642	99.70562	34.03204	33.64322	33.58726
3D ultrasound	99.87215	99.81436	99.75443	33.65629	33.70929	33.66517
3D scanner	99.89427	99.83960	99.89441	34.07125	34.08311	34.19787
3D radiography	99.89853	99.88799	99.89711	34.09614	34.09556	34.08134
3D X-ray	99.87732	99.89493	99.89467	34.09847	34.09597	34.09613
3D CT-scan	99.79369	99.65950	99.76318	33.80594	33.89826	33.89117
1D ultrasound		99.84620			33.82922	
1D brain		99.75471			33.79463	
1D endoscopy		99.79556			33.89125	

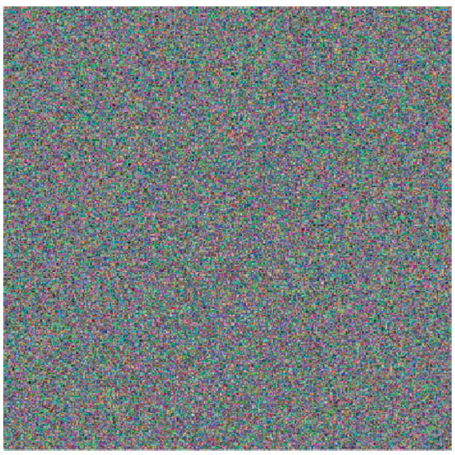


Figure 20: Key sensitivity test applied at the decryption phase.

[5, 11], the designers employ a permutation and a XOR diffusion process. According to the architecture complexity of the permutation, the proposed permutation process takes less time than their algorithms. Completing with reference [6], the designer employs two processes: a substitution and XOR diffusion. The proposed substitution algorithm takes less time than this algorithm.

In the proposed algorithm, the permutation, substitution, and diffusion are not complex that can be done with reasonable resources and low computational cost. In addition, they are independent that can be performed in parallel execution. This reduces significantly the execution time. The proposed scheme provides high-level security with high performance and reasonable resources.

Table 13: Simulation results of the NIST 800-22 test for the proposed PRNG.

Statistical	P value	Status
Status frequency	0.4372742	Pass
Block frequency ($m = 128$)	0.4372742	Pass
Forward CUSUM	0.4372742	Pass
Reverse CUSUM	0.4372742	Pass
Runs	0.4372742	Pass
Long runs of ones	0.4372742	Pass
Binary matrix rank	0.4372742	Pass
Spectral DFT	0. 8755390	Pass
Nonoverlapping template ($m = 9$)	0.7070707	Pass
Overlapping template ($m = 9$)	0.5449921	Pass
Universal	0.0713232	Pass
Approximate entropy ($m = 10$)	0.0125474	Pass
Random excursions ($x = +1$)	0.9030558	Pass
Random excursions variant ($x = -1$)	0.3974291	Pass
Linear complexity ($M = 500$)	0.1922722	Pass

Table 14: Performance of the proposed encryption algorithm.

Process	Key generation	Random permutation	S-box substitution	XOR diffusion
Time (s)	0.0192	0.2426	0.1880	0.0378
Total time (s)		0.4876		
Speed (Mb/s)		12.9		

Table 15: Comparative study of encryption algorithm speed.

Work	Permutation	Substitution	Diffusion	S (Mb/s)
Reference [2]	√	—	√	0.2
Reference [3]	—	√	√	9.6
Reference [4]	—	—	√	3.4
Reference [7]	—	—	√	13.52
Reference [8]	√	—	√	2.4
Proposed algorithm	√	√	√	12.9

4. Conclusion

In this work, we have proposed an improved chaos-based symmetric cryptosystem for medical image encryption and decryption. The SHA-256 is used to generate a 256-bit key of the cryptosystem. A complex chaos-based PRNG is designed to generate a high-quality encryption key. The generated key presents high randomness behaviour, high entropy, and high complexity. Improved architecture based on confusion and diffusion property is proposed for image encryption. The image undergoes a processing cycle of four operations in order to produce the encrypted image: random permutation of the position of pixels, position permutation of pixel's bits, S-box pixels substitution, and XOR diffusion. R rounds of encryption can be performed in a loop to enjoy a high-level performance. In-depth measurements are taken with several medical images to assess the strength of the proposed cryptosystem against the most known attacks. The results demonstrate that the algorithm offers high performance and enhanced security with low computational complexity. The obtained image entropy is equal to 8 which is an important measure of randomness. The NIST test indicates that the proposed PRNG is appropriate for secure image encryption. In addition, the architecture is easily parallelizable to speed up execution and meet real-time application requirements. The comparative study with recent work indicates that the proposed algorithm provides the best performance. However, it is extremely adapted to protect and authenticate images, which can be used in several domains.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] S. Ajili, "Joint watermarking\encryption image for safe transmission: application on medical imaging," in *Proceedings of the International Conference: Global Summit on Computer and Information Technology*, London, UK, 2014.

[2] M. Ali Hajjaji, "A medical image crypto–compression algorithm based on neural network and PWLCM," *Journal of Multimedia Tools and Applications*, vol. 78, no. 11, pp. 14379–14396, 2019.
[3] D. Manel, "An enhancement crypto–compression scheme for image based on chaotic system," *International Journal of Applied Engineering Research*, vol. 11, no. 7, pp. 4718–4725, 2016.
[4] D. Jeevitha, "Novel medical image encryption using DWT block–based scrambling and edge maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, 2020.
[5] J Liu, "A novel fourth order chaotic system and its algorithm for medical image encryption," *Journal of Multidimensional Systems and Signal Processing*, vol. 1, pp. 1637–1657, 2019.
[6] N. Tsafack, J. Kengne, B. Abd–El–Atty, A. M. Iliyasu, K. Hirota, and A. A. Abd EL–Latif, "Design and implementation of a simple dynamical 4–D chaotic circuit with applications in image encryption," *Information Sciences*, vol. 515, pp. 191–217, 2020.
[7] X. Gao, "Image encryption algorithm based on the matrix semi–tensor product with a compound secret key produced by a Boolean network," *Information Sciences*, vol. 539, pp. 195–214, 2020.
[8] Y. Naseer, T. Shah, F. Attaullah, and A. Javeed, "Advance image encryption technique utilizing compression, dynamical system and *S*–boxes," *Mathematics and Computers in Simulation*, vol. 178, pp. 207–217, 2020.
[9] H. Li, "Chaos–based image encryption algorithm with orbit perturbation and dynamic state variable selection mechanisms," *Journal of Optics and Lasers in Engineering*, vol. 115, pp. 197–207, 2019.
[10] L. Hongjun, "Chaos–based fast colour image encryption scheme with true random number keys from environmental noise," *Journal of Image Processing, IET*, vol. 11, no. 5, pp. 324–332, 2017.
[11] J. Zhou, "Fast color image encryption scheme based on 3D orthogonal Latin squares and matching matrix," *Journal of Optics and Laser Technology*, vol. 131, 2020.
[12] Y. Zhang, "The fast image encryption algorithm based on lifting scheme and chaos," *Information Sciences*, vol. 520, 2020.
[13] P. F. Ping, Y. Mao, and Z. Wang, "Designing permutation–substitution image encryption networks with Henon map," *Neurocomputing*, vol. 283, pp. 53–63, 2018.
[14] V. Xu, "A 2D logistic map and Lorenz–Rossler chaotic system based RGB image encryption approach," *Journal of Multimedia Tools and Applications*, vol. 12, 2020.
[15] L. Liu, "An image encryption algorithm based on Baker map with varying parameter," *Journal of Multimedia Tools and Applications*, vol. 76, pp. 16511–16527, 2020.
[16] M. Gafsi, "High securing cryptography system for digital image transmission," *Smart Innovation: Systems and Technologies*, vol. 146, pp. 311–322, 2020.
[17] M. Ali Hajjaji, "FPGA Implementation of digital images watermarking system based on discrete haar wavelet transform," *Journal of Security and Communication Networks, Hindawi*, vol. 2019, Article ID 1294267, 2019.
[18] A. Horé and D. Ziou, "Image quality metrics: PSNR vs. SSIM," 2010.
[19] G. Ramzi, "A novel chaos–based image encryption using DNA sequence operation and Secure Hash Algorithm SHA–2," *Nonlinear Dynamics*, vol. 83, no. 3, pp. 1123–1136, 2016.
[20] M. Dridi, "Cryptography of medical images based on a combination between chaotic and neural network," *Journal of Image Processing, IET*, vol. 11, no. 5, pp. 324–332, 2016.
[21] Y. Xian, "Image encryption based on chaotic Sub–Block scrambling and chaotic digit selection diffusion," *Journal of Optics and Lasers in Engineering*, vol. 134, 2020.
[22] S. Kumar, "Medical image encryption using fractional discrete cosine transform with chaotic function," *Journal of Medical & Biological Engineering & Computing*, vol. 57, pp. 2517–2533, 2019.
[23] M. Gafsi, "Efficient encryption system for numerical image safe transmission," *Journal of Electrical and Computer Engineering, Hindawi*, vol. 12, 2020.
[24] M. Gafsi, "XSG for hardware implementation of a robust watermarking system," 2016.
[25] M. Ali Hajjaji, "Discrete cosine transform space for hiding patient information in the medical images," in *Proceedings of the IEEE International Conference on Design & Test of Integrated Micro & nano–Systems (DTS'19)*, Gammarth, Tunisia, 2019.
[26] M. Ali Hajjaji, "Combining DWT/KLT for secure transfer of color images," in *Proceedings of the IEEE International Conference on Design & Test of Integrated Micro & nano–Systems (DTS'19)*, Gammarth, Tunisia, 2019.
[27] M. Ali Hajjaji, "Real time implementation of numerical watermarking system using xilinx system generator," in *Proceedings of the 16th International Conference on Sciences and Techniques of Automatic Control & Computer Engineering-STA'2015*, Monastir, Tunisia, 2015.
[28] S. Aashiq Banu, "Tri–level scrambling and enhanced diffusion for DICOM image cipher–DNA and chaotic fused approach," *Journal of Multimedia Tools and Applications*, vol. 79, pp. 28807–28824, 2020.
[29] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos–based cryptosystems," *International Journal of Bifurcation and Chaos*, vol. 16, no. 8, pp. 2129–2151, 2006.
[30] W. Yue, "NPCR and UACI randomness tests for image encryption, cyber journals: multidisciplinary journals in science and technology," *Journal of Selected Areas in Telecommunications (JSAT)*, vol. 79, 2011.
[31] A. Rukhin, "A statistical test suite for random and pseudorandom number generators for cryptographic applications, special publication 800–22 revision 1a," *National Institute of Standards and Technology (NIST)*, vol. 79, 2010.
[32] A. Safwan El, "A new chaos–based image encryption system," *Journal of Signal Processing: Image Communication*, vol. 41, pp. 144–157, 2016.
[33] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.