mathematics

MDPI

Article

A Universal Image Compression Sensing–Encryption Algorithm Based on DNA-Triploid Mutation

Yinghong Cao, Linlin Tan, Xianying Xu and Bo Li *

School of Information Science and Engineering, Dalian Polytechnic University, Dalian 116034, China; caoyinghong@dlpu.edu.cn (Y.C.); 230520854000583@xy.dlpu.edu.cn (L.T.); xuxiany@dlpu.edu.cn (X.X.)
* Correspondence: libo_15@dlpu.edu.cn

Abstract: With the fast growth of information technology (IT), the safety of image transmission and the storing of images are becoming increasingly important. Traditional image encryption algorithms have certain limitations in transmission and security, so there is an urgent need for a secure and reliable image encryption algorithm. A universal compression sensing (CS) image encryption (IE) algorithm based on DNA-triploid mutation (DTM) is presented in this paper. Firstly, by using the CS algorithm, an image is compressed while obtaining a range of chaotic sequences by iteration of a chaotic map. Then, DNA sequences are generated by encoding the image and, based on the DTM, new mutant DNA sequences are generated according to specific rules. Next, the chaotic sequences are operated at the DNA level to perform confusion and diffusion operations on the image to ensure the security of the data. Finally, DNA decoding is carried out to obtain the compressed encrypted image. The simulation results show that the algorithm can effectively complete encryption and decryption of images. The performance test results show that the algorithm has a sufficiently large key space of 10^{587}. The information entropy of the cipher image is close to 8. In summary, both simulation experiments and performance tests fully show that a high level of security and reliability for the proposed algorithm in protecting image privacy is achieved.

Keywords: privacy protection; DNA-triploid mutation; compression sensing; image encryption

MSC: 37N99

check for updates

Citation: Cao, Y.; Tan, L.; Xu, X.; Li, B. A Universal Image Compression Sensing–Encryption Algorithm Based on DNA-Triploid Mutation. *Mathematics* **2024**, *12*, 1990. https://doi.org/10.3390/math12131990

Academic Editor: Jonathan Blackledge

Received: 28 May 2024
Revised: 25 June 2024
Accepted: 26 June 2024
Published: 27 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

As IT develops rapidly, digital images are applied in many fields [1], such as medical images, security monitoring, digital art and so on. No matter which field it is applied to, the privacy protection of digital images is always a problem that needs to be focused on. In order to protect private information, researchers have designed traditional encryption algorithms such as AES [2], RSA [3], DES [4] and so on, but due to the large amount of data contained by digital images, the effectiveness of the algorithms needs to be improved [5]. So, researchers have proposed many new encryption algorithms to improve the performance of encryption.

Recently, chaotic systems have been widely exploited in many fields due to their excellent characteristics, such as IE [6,7], chaotic radar system [8], and so on. And the design of systems with complex dynamics has received widespread attention [9–11]. The chaotic sequences produced by the iteration of chaotic systems feature unpredictability [12], sensitivity to initial values, and randomness [13–17] and are therefore often used as keys for IE [18,19]. However, classical chaotic systems have some limitations such as small key space and phase space and can be easily destroyed [20]. Because many nonlinear dynamic behaviors that can be generated by neural networks [21–27], IE algorithms can be provided with randomness and security. The Tabu Neuron Network (TNN) [28] is utilized for the presented algorithm. High efficiency, security, and reliability can be provided to the

encryption algorithm by the TNN with those characteristics, which can better protect the transmission of private information.

Many encryption algorithms have been proposed based on chaos theory [29–33]. For instance, Mohamed et al. raised a substitution matrix created by using chaotic state variables, using it as the key and the encryption algorithm itself for encryption [34]. Based on the combination of IE and chaotic systems, some researchers have combined encryption algorithms with DNA rules [35–40] to create a more secure and efficient encryption algorithm that provides a more reliable algorithm for privacy protection. Fan. Et al. raised an algorithm based on horizontal confusion and diffusion of eight-base DNA [41]. Wang. Et al. employed the look-up table algorithm to quickly obtain the results of DNA arithmetic and save encryption time [42]. Furthermore, a nonlinear feedback shift register and image encryption with a DNA computation algorithm were utilized to provide high security to the algorithm [43]. In addition, the encryption process can be made random and unpredictable by performing mutation operations on DNA sequences, thus increasing the difficulty of cracking. In this algorithm, an effective algorithm based on triplet mutation [44] is proposed and a series of encryption operations are performed at the DNA level by using chaotic sequences. The algorithm is able to resist attack tests well and protects image information without being destroyed, which improves the safety of image encryption.

The quantity of data in an image is increased due to the need to convert binary data into a sequence of four bases when encrypting at the DNA level. To reduce the amount of data when encrypting, the image can be compressed before performing the encryption algorithm. Gao et al. presented a compression technique using a back propagation neural network [45], but because of the "jaggedness phenomenon" and step size selection, the training speed was slow. Liu et al. suggested a modified image compression algorithm that combined Huffman coding, integer wavelet transform, and linear prediction [46], but it had slow compression speed, while the CS [47] technique performs better in terms of compression speed and this algorithm uses CS. It utilizes the sparsity or low dimensional structure of the signal to efficiently rebuild the original signal with few sampled data [48]. The Discrete Cosine Transform (DCT) [49] algorithm is utilized in this algorithm to sparsely represent the original signal by combining the Hadamard matrix and the chaotic sequences to generate an observation matrix. In the stage of signal reconstruction, to reconstruct the compressed signal, the Orthogonal Matching Pursuit (OMP) [50] algorithm is applied. The size of the image data is significantly reduced by this algorithm, which helps to reduce transmission and storage costs.

For some traditional image encryption methods, there may be the risk that the encryption algorithm has been cracked or has loopholes, resulting in the encrypted image being easy to decrypt and causing the risk of information leakage. The algorithm is based on the DTM and the image is compressed first and then encrypted, which can effectively avoid the risk of cracking or loopholes that may exist in traditional encryption methods and therefore improve the security of the image data. Moreover, the chaotic sequences generated by this algorithm using the TNN system iteration have highly complex nonlinear characteristics, which can provide higher encryption strength than traditional encryption methods, making decryption much more difficult and effectively preventing information from being cracked or stolen.

In conclusion, a universal compression–encryption algorithm using DTM is presented. Key highlights from this paper are:

1. The universal image is first compressed before encryption to decrease the size of the image to be encrypted and increase the encryption efficiency.
2. The chaotic sequences generated by TNN through iteration are fully utilized to provide pseudo-randomness to the encryption algorithm by combining it with the encryption algorithm.
3. Due to the high randomness in the way DNA sequences combine, confusion and diffusion operations at the DNA level provide a strong randomization.

The paper is organized as follows. The rich dynamics of the TNN chaotic map, the process of CS, and the details of DTM are shown in Section 2. The specific steps of the IE algorithm are described in Section 3. The effects of this algorithm are illustrated in Section 4. Performance tests and analysis are carried out in Section 5. What is done is summed up in the final section.

2. Preliminaries

2.1. Chaotic System

2.1.1. Chaotic Map

The 2D hyperchaotic map that the TNN used is shown in Equation (1):

$$\begin{cases} x(i+1) = -ax(i) + b\sin(gx(i)) + y(i) \\ y(i+1) = -cy(i) - d\sin(gx(i)) \end{cases} \quad (1)$$

where the activation gradients are g, a, b, c, and d for the system parameters and the initial values are x_0 and y_0.

The phase diagrams (PDs) of the NN are given in Figure 1, with the parameters (a, b, c, d, g) = (-0.4, 1, 0.9, -0.8, 3.1) and $y_0 = 0$, $x_0 = -0.1$ the initial values. The PDs for different values of parameter a are shown for a= -0.4, -0.6, and -0.8 in Figure 1a–c, respectively. The PDs show that the system has a multi-structured phenomenon as the value of a increases. This phenomenon can generate high-quality random sequences. The bifurcation diagram (BD) and Lyapunov exponential spectrum (LEs) of the TNN are shown in Figure 2a,b. When the parameter range is $a \in [-0.98, 0.98]$, it is in the full domain hyperchaotic state. In the full domain hyperchaotic state, the state variables of the system have high randomness, which can generate more random and complex key sequences and increase the security of the IE algorithm. The complexity diagram for parameters a and d can be seen in Figure 2c. The complexity of the system is close to 1, which indicates that the chaotic sequences iterated by the hyperchaotic map have high complexity.

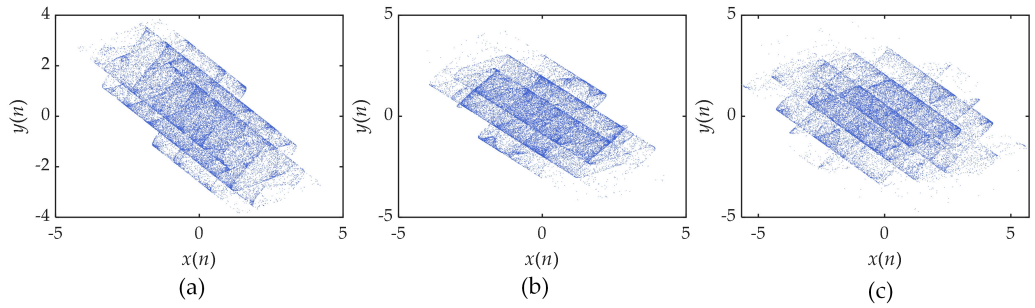


Figure 1. Three attractors with the variation of parameter a: (**a**) $a = -0.4$, (**b**) $a = -0.6$, and (**c**) $a = -0.8$.

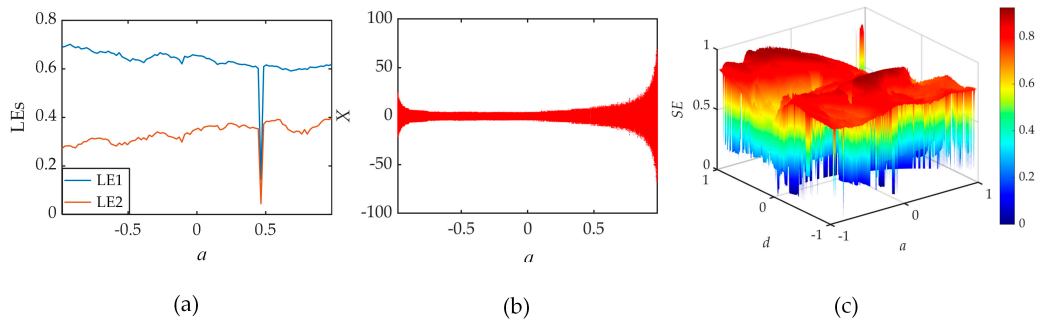


Figure 2. TNN: (**a**) LEs of parameter a, (**b**) BD of parameter a, and (**c**) complexity of parameters a and d.

2.1.2. Randomness Test

To prove the reliability of the chaotic map in IE, NIST tests are performed, which represent a usual method to check the randomness of sequences with 15 different tests. In evaluating the randomness of chaotic keys, the NIST tests help to ensure that the generated keys are sufficiently random and secure when image encryption is performed by using chaotic sequences generated by iteration of the system. Therefore, the NIST tests are important in evaluating the randomness of chaotic keys to help ensure the security and reliability of the image encryption system. The results of the tests are analyzed using the standards of p-value and pass rate (PR). Sequences are considered to be random when the 15 tests have all p-values equal to 0.01 or more and all PRs are greater than 96%. The results obtained after the NIST tests are presented in Table 1, from which it is clear that all the tests pass the standards and the chaotic sequences have a high degree of randomness, which makes them well suited for application in the IE algorithm. In the tests, the chaotic sequences are generated by the system by iterating 6.25×10^6 times, and the two obtained chaotic sequences are converted into a binary bit stream, at which point the total length of the bit stream is 10^8.

Table 1. The results of the NIST tests.

Items	Designed Algorithm		
	***p*-Value**	**PR**	**Pass/Fall (P/F)**
Frequency	0.020548	100%	P
Block Frequency	0.437274	99%	P
Cumulative Sums	0.071177	100%	P
Runs	0.021999	98%	P
Longest Run	0.678686	98%	P
Rank	0.759756	100%	P
FFT	0.911413	98%	P
Nonoverlapping Template	0.011791	100%	P
Overlapping Template	0.657933	99%	P
Universal	0.719747	97%	P
Approximate Entropy	0.798139	100%	P
Random Excursions	0.032923	100%	P
Random Excursions Variant	0.016717	97%	P
Serial Linear Complexity	0.779188	98%	P
	0.275709	98%	P

2.2. Compression Sensing Technology

The three processes of the CS are represented in Figure 3. The DCT algorithm is used for representing the original signal sparsely and the Hadamard matrices and chaotic sequences are combined to generate observation matrices. Sparse observations can be seen in Equation (2).

$$y = \varphi x = \varphi \psi x, \tag{2}$$

where φ is the observation matrix, x is a bunch of signals of length N, Ψ is the sparse matrix that transforms the signal into the frequency domain S, and y is the observation and is the final sampling result.

The reconstruction algorithm of CS reconstructs sparse signals from a small number of linear observations. The signal reconstruction is as presented in Equation (3). Specifically, to find a sparse signal x making the l_0-parameter of x minimal while satisfying that the measurement matrix φx agrees with the observed signal y. After obtaining the reconstructed signal, the original signal is reconstructed with transformed base values. In the signal reconstruction stage, the OMP algorithm is utilized, which has less complexity and is closely related to the value in each iteration. The basic idea of the OMP algorithm is to

approximate a sparse representation of the signal by progressively selecting the atoms that best match the residuals.

$$x = \text{argmin}\|x\|_0 \; s.t. \varphi x = y \tag{3}$$

where $\|x\|_0$ denotes the l_0 paradigm, the number of non-zero elements in the vector x.

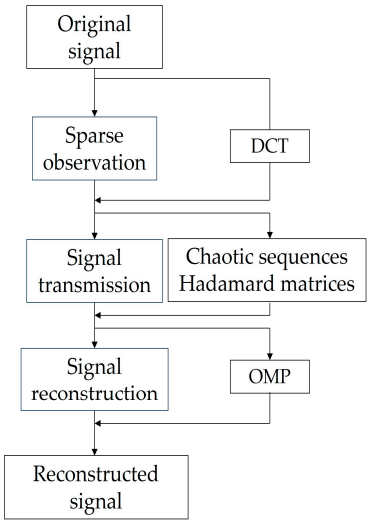


Figure 3. Compression sensing process.

2.3. DNA-Triploid Mutation

In the nucleotide structure, four bases are included, guanine (G), cytosine (C), adenine (A), and thymine (T). Based on the Watson–Crick rule of complementarity, complementary base pairs are formed by A and T and C and G. In the binary coding of DNA, binary codes (11, 10, 01, 00) are assigned to bases (T, A, G, C) to generate 24 coding rules in total. However, only eight coding rules are in line with the base complementarity rules, as presented in Figure 4. Furthermore, the mutation of bases may be caused by the self-replication of DNA during the transmission of genetic information.

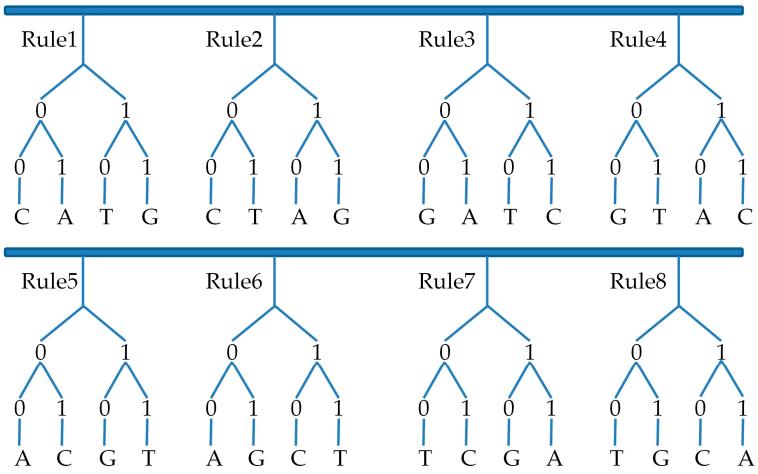


Figure 4. DNA rules.

Under the rules of the DNA triploid replication stage, a design for the DNA mutation rule is performed. During triploid replication, each base is copied three times, so the presence of each base is more stable in the new DNA strand after mutation. This helps to reduce base-pairing errors, improving the accuracy of DNA replication. In triploid organisms, DNA mutation occurs by copying a single DNA strand into three strands and then synthesizing new mutant DNA strands according to specific rules. The synthesis

rule defined by the algorithm takes the binary numbers denoted by the basis and adds them together to generate a new binary number that corresponds with the base number. In addition, eight DTM rules based on eight DNA coding rules are listed in Figure 5. The DTM provides a good algorithm for IE.

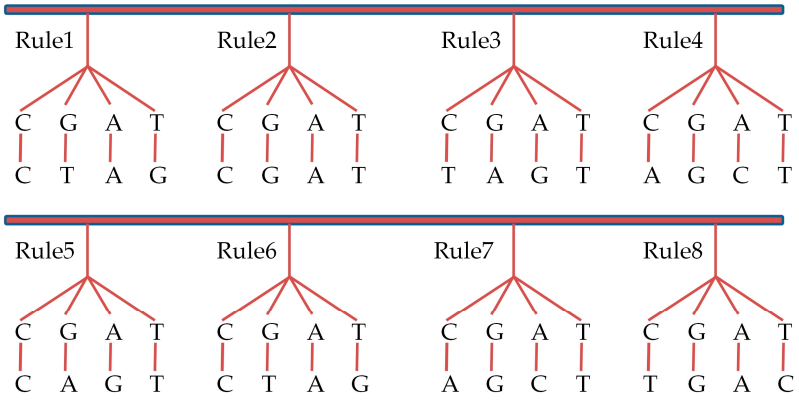


Figure 5. DNA-triploid mutation rules.

3. Designed Algorithm

For securing the image, an algorithm for encryption at the DNA level after compression is proposed. Taking a color image as an example, first an image is input and then compressed using the compression sensing (CS) technique in a certain compression ratio (CR). Then, encode it into DNA sequences and apply DNA-triploid mutation (DTM) to the coded image. Finally, perform confusion and diffusion operations at the DNA level by using chaotic sequences; then, it is decoded to obtain the encrypted image. Chaotic sequences are generated by iteration of the Tabu Neuron Network (TNN). The specific algorithm is illustrated in Figure 6.

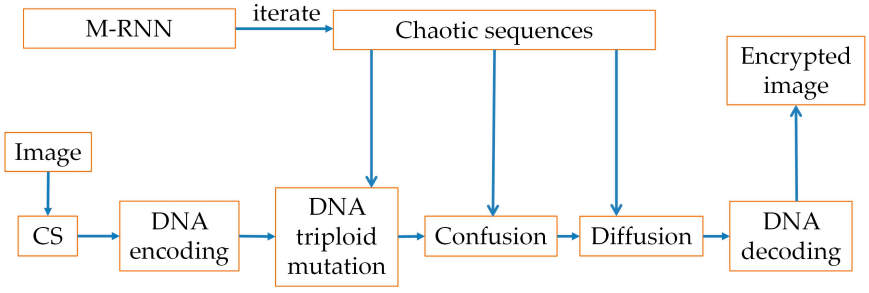


Figure 6. Encryption algorithm flowchart.

3.1. Encryption Algorithm

(1) Input an image and record the image S sized $H \times W \times L$.

(2) CS. A certain CR is set and the following (a), (b), and (c) are the detailed image compression process.

(a) Sparse representation. Set a CR and the sparse basis ψ is obtained by inputting the size of H into the sparse basis function DCT. The signal is transformed into the frequency domain and an expression of sparse representation is presented in Equation (4). R, G, and B, respectively, for the sparse representation channels, with the corresponding results denoted as R_1, G_1, and B_1.

$$\begin{cases} R_1 = \psi R \psi' \\ G_1 = \psi G \psi' \\ B_1 = \psi B \psi' \end{cases} \tag{4}$$

(b) Signal observation. Input the parameters and initial values of chaotic map, then perform the pre-iteration process. Observation matrices φ_1, φ_2, and φ_3 are obtained by combining Hadamard matrices with partial pre-iterative results. These observation matrices are utilized with R_1, G_1, and B_1 to obtain the observation results in S_1, S_2, and S_3, as shown in Equation (5).

$$\begin{cases} S_1 = \varphi_1(\varphi_1 R_1)' \\ S_2 = \varphi_2(\varphi_2 G_1)' \\ S_3 = \varphi_3(\varphi_3 B_1)' \end{cases} \tag{5}$$

(c) Quantitative processing. The minimum and maximum observations min_i and max_i by R, G, and B groups are taken separately. According to those results, three groups of results D_1, D_2, and D_3 are quantified and presented in Equation (6). Finally, RGB images are reconstructed by quantization results; after CS, the result is called SS_1 with its size $HH \times WW \times LL$.

$$\begin{cases} D_1 = \text{round}(255\frac{S_1 - \min_R}{\max_R - \min_R}) \\ D_2 = \text{round}(255\frac{S_2 - \min_G}{\max_G - \min_G}) \\ D_3 = \text{round}(255\frac{S_3 - \min_B}{\max_B - \min_B}) \end{cases} \tag{6}$$

(3) Generation of chaotic sequences. Input SS_1 into the hash-256 to obtain four hash values h_a, h_b, h_c, and h_d. Combine the hash values with the initial values to generate new initial values. Three new chaotic sequences $chaoSeq_1$, $chaoSeq_2$, and $chaoSeq_3$ are obtained by iterating through the TNN chaotic map. The reshaping results are $Seqq_1$, $Seqq_2$, and $Seqq_3$, which are shown in Equation (7).

$$\begin{cases} Seqq_1 = \lfloor (chaoSeq_1 + 100) \times 10^{10} \rfloor \% 256 + 1 \\ Seqq_2 = \lfloor (chaoSeq_2 + 100) \times 10^{10} \rfloor \% 256 + 1 \\ Seqq_3 = \lfloor chaoSeq_3 + 100) \times 10^{10} \rfloor \% 256 + 1 \end{cases} \tag{7}$$

(4) DNA encoding. Divide the compressed image SS_1 into three channels for encoding operation, respectively. The specific operation steps are as follows.

(a) Convert decimal pixel values to binary.

(b) Encode the binary numbers into DNA sequences; the specific encoding rules are as given above.

(c) Reshape them into three channels DD_1, DD_2, and DD_3 of size $HH_1 \times WW_1$.

Since both the confusion and diffusion are operated at the DNA level, the chaotic sequences $Seqq_1$, $Seqq_2$, and $Seqq_3$ need to be subjected to DNA encoding operations to obtain three chaotic sequence matrices, XX, YY, and ZZ, of size $HH_1 \times WW_1$.

(5) DNA mutation. DNA mutation is performed at each position in the three channels according to the eight mutation rules that are listed above. The value DD_i (i, j) of each channel after encoding and the same position of the three chaotic sequences $Seqq_1$, $Seqq_2$, and $Seqq_3$ are mutated based on certain mutation rules, and the results are stored on D_1, D_2, and D_3.

(6) Confusion operation. Recombine D_1, D_2, and D_3 into NNN of size $3HH_1 \times 4WW_1$. Perform point-to-point confusion on the reorganized matrix. The confusion operation is performed by using an Arnold pseudo-random matrix. The new coordinate position vector k is calculated from the coordinates (i, j) and then the elements at coordinates (i, j) are exchanged with the elements at coordinates (k (1), k (2)) to obtain the confused matrix. Then, reorganize the confused matrix into three matrices U_1, U_2, and U_3 of size $HH_1 \times 4WW_1$.

(7) Diffusion operation. Reshape the three matrices into XX, YY, and ZZ of the same size to obtain SS_1, SS_2, and SS_3. Respectively perform the xor operation at the DNA level between SS_1, SS_2, SS_3, and XX, YY, and ZZ. The xor operation is performed in parity. Take SS_1 as an example, set a flag i, when i is even, the xor operation is performed by SS_1 and XX. When i is odd, the xor operation is performed by SS_1 and YY. The operation is carried out a total of $HH_1 \times 4WW_1$ times. Finally, the matrix after the xor operation is reorganized into three matrices SD_{11}, SD_{22}, and SD_{33} of size $HH_1 \times 4WW_1$.

(8) DNA decoding operation. Decode the three channels separately and then merge them together to obtain the cipher image TT_1.

3.2. Decryption Algorithm

In this algorithm, the flowchart of the decryption–reconstruction algorithm is shown in Figure 7. First, encode the TT_1. Specifically, the inverse diffusion and inverse confusion operations are first carried out. Then, inverse DNA mutation is performed on it and it is allowed to perform DNA decoding operations to obtain a decrypted image. Finally, the basis function that best matches the residuals is chosen step by step to reconstruct a sparse signal that better fits the observed signal and the OMP algorithm can be used to reconstruct the decrypted image.

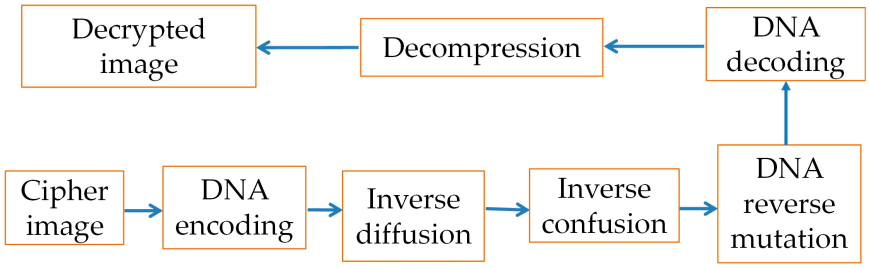


Figure 7. Decryption algorithm flowchart.

4. Experimental Results and Simulation Effects

In this chapter, extensive simulation experiments are conducted to assess both the efficiency and security of the presented encryption algorithm. This algorithm is realized in the Windows 10 system based on the MATLAB R2019b platform with the computer hardware environment of Intel CPU 1.60 GHz; RAM 8.00 GB; Core i5-10210U.

Due to copyright problems, this experiment uses multiple images for simulation tests and finally uses "1.1.tiff"(256 × 256 × 3), "1.2.tiff"(256 × 256× 3), "2.1.tiff"(512 × 512× 3), "2.2.tiff"(512 × 512 × 3), "3.1.tiff"(1024 × 1024 × 3), "3.2.tiff"(1024 × 1024 × 3) "4.1.tiff"(256 × 256), and "4.2.tiff"(512 × 512) to test the simulation of the algorithm, as shown in Figure 8a.

From the simulation results in Figure 8, it can be seen that the image is first compressed by this algorithm, which reduces the amount of data, reduces the redundant information, and improves the security of encryption. And its decryption process is reversible and can decrypt and recover the cipher image.

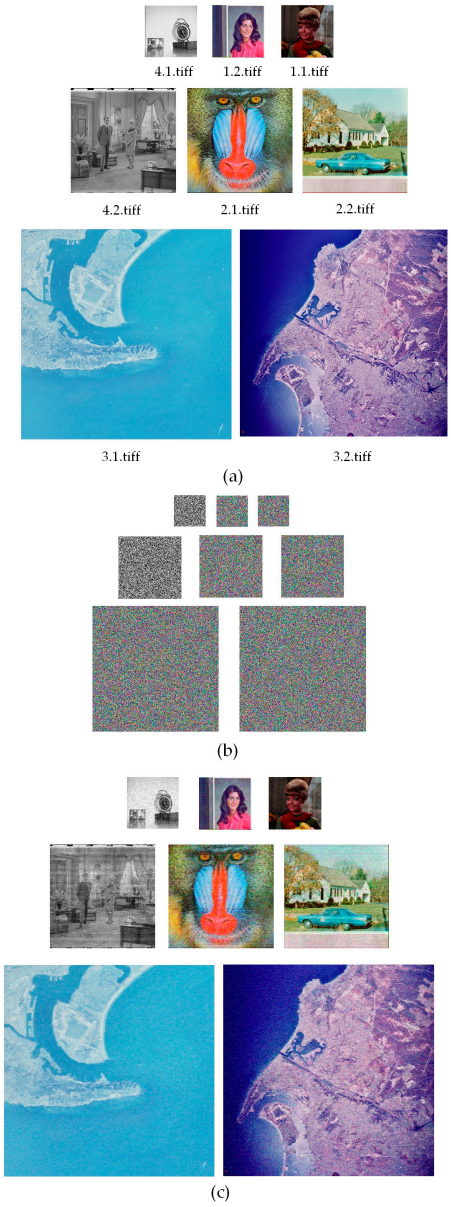


Figure 8. Simulation results for the compression–encryption algorithm: (**a**) original images, (**b**) cipher images, and (**c**) decrypted–reconstructed images.

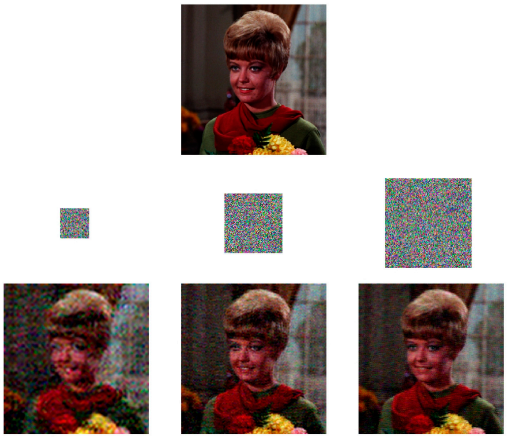
5. Security Analysis

5.1. Performance of Compression

The CRs are set to 0.2, 0.4, and 0.6, respectively, and three images of different sizes are compressed and reconstructed. The results of the tests for compression and reconstruction of the same image using different CRs can be seen in Figure 9. The quality of the reconstruction is represented as peak signal-to-noise ratio (PSNR) by a comparison between the reconstructed and original images given in Equation (8). The results of the reconstruction as presented in Table 2. From this, it is clear that even with low CRs an excellent quality for the reconstructed image can be achieved. A comparison between the algorithm and other algorithms for reconstruction quality is given in Table 3 [51–53], which shows the advantages of this algorithm, i.e., that the reconstruction quality using this algorithm is better compared with other algorithms. The comparison of the compression time of the CS algorithm with other algorithms is shown in Table 4 [45,46], which reflects the advantages of this algorithm in terms of compression efficiency.

$$\begin{cases} MSE = \dfrac{1}{CK}\sum\limits_{i=0}^{C-1}\sum\limits_{j=0}^{K-1}\|M(i,j) - R(i,j)\|^2 \\ PSNR = 10\log_{10}\left(\dfrac{MAX_I^2}{MSE}\right) \end{cases} \tag{8}$$

where M and R denote the original image and the reconstructed image.



(a) (b) (c)

Figure 9. Compression reconstruction effects under different CRs: (**a**) 0.2, (**b**), 0.4, and (**c**) 0.6.

Table 2. Comparison of PSNR with different CRs for different images.

Images	CRs	PSNR (dB)		
		R	G	B
1.1	0.4	26.8518	26.9826	27.6398
2.2	0.4	28.4870	28.2920	28.4591
3.1	0.4	27.7327	28.8674	28.9207
1.1	0.6	27.0836	27.9453	28.7639
2.2	0.6	29.7938	29.0476	29.9328
3.1	0.6	30.0084	30.8790	30.6279

Table 3. Comparison of PSNR with other algorithms.

Algorithm	Sizes	CRs	$PSNR_{aver}$
Reference [51]	512 × 512	0.5	23.3608
Proposed	512 × 512	0.5	28.9032
Reference [52]	256 × 256	0.5	28.0714
Proposed	256 × 256	0.5	28.6956
Reference [53]	256 × 256	0.75	29.5600
Proposed	256 × 256	0.75	30.7205

Table 4. Comparison of compression time with other algorithms.

Algorithm	Sizes	CRs	Time (s)
Reference [45]	256 × 256 × 111	0.5	186.91
Proposed	256 × 256 × 111	0.5	21.0424
Reference [46]	3840 × 2160	0.7236	2
Proposed	3840 × 2160	0.7236	1.855

5.2. Analysis of Security Key

5.2.1. Key Space

Key space is very important in the security assessment of encryption algorithms. The size of the key space directly affects the security of the encryption algorithm because a large key space means more possibilities, making it difficult to break the encryption algorithm. The key to the algorithm is divided into three parts: the chaos-related parameters, initial values, and hash values. After a series of tests, the key space of each part and the key space in total are given, as listed in Table 5. The encryption algorithm in general is considered to be resistant to brute force attacks when a larger than 2^{100} [54] key space is available. A comparison of key space in this algorithm with that of other algorithms presented in Table 6 [55–58], which directly shows that it has a sufficiently large key space, ensuring the security of the encryption algorithm.

Table 5. Key space.

Parameters	Key Space
b, g	10^{15}
a, c, d, y_0, h_c, h_d	10^{16}
x_0, h_a, h_b	10^{17}
Total key space	$10^{177} \approx 2^{587}$

Table 6. Comparison between key space and other algorithms.

Algorithms	Reference [55]	Reference [56]	Reference [57]	Reference [58]	Proposed
Key space	2^{256}	2^{256}	2^{197}	2^{154}	2^{587}

5.2.2. Key Sensitivity

The key sensitivity test is a method for evaluating the security of an encryption algorithm. In testing the key sensitivity of the encryption process, a perturbation of 10^{-15} is added to b to change the encryption process key. In the case of the encryption process, the number of pixels change rate (NPCR) is used as a measure to evaluate the key sensitivity, as shown in Equation (9). The NPCRs of different images are shown in Table 7. Even a slight perturbation in the key during the encryption process may cause the encryption results to differ from the original results by more than 99.6094%. A 10^{-16} perturbation is added in the decryption process for the three sets of parameters a, c, and d. It is clear from Figure 10 that the images after decryption are not restored correctly. As can be seen from the figures, even though the perturbations added are very small, decryption cannot be successfully made, which indicates that the key is very sensitive.

$$NPCR(S_1, S_2) = \frac{1}{CK}\sum_{i=1}^{C}\sum_{j=1}^{K}|S_1(i,j) - S_2(i,j)| \times 100\%, \quad (9)$$

where $S_1(i, j)$ and $S_2(i, j)$ are two cipher images that change only one pixel before and after the plaintext images. The cipher images are of size $C \times K$.

Table 7. Results of key sensitivity tests.

Images	CR	NPCR (%)		
		R	G	B
1.1	0.6	99.6099	99.6174	99.6732
1.2	0.6	99.6505	99.6084	99.6091
2.1	0.6	99.6103	99.6092	99.6100
2.2	0.6	99.6235	99.6106	99.6083

Table 7. *Cont.*

Images	CR	NPCR (%)		
		R	G	B
3.1	0.6	99.6253	99.6284	99.6090
3.2	0.6	99.6098	99.6134	99.6101
4.1	0.6		99.6098	
4.2	0.6		99.6102	

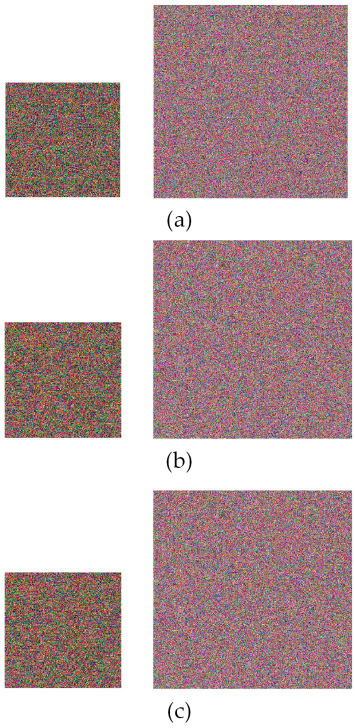


Figure 10. Key sensitivity of the decryption process: (**a**) parameter $a = a + 10^{-16}$, (**b**) parameter $c = c + 10^{-16}$, and (**c**) parameter $d = d + 10^{-16}$.

5.3. Attack Resistance Test

5.3.1. Differential Attack

Information obtained by attackers through changing information in plaintext images to analyze changes in cipher images causes differential attacks. To measure the ability of an algorithm to resist differential attacks, the NPCR and uniform average changing intensity (UACI, Equation (10)) are utilized. UACI is a measure of the degree of difference between the pixel values at the corresponding positions of two images. The IE algorithm can be successful against differential attacks, as proved by an NPCR of over 99.6094% and a UACI close to 33.4635%. The test results in Table 8 show that both values are close to theoretical expectations of NPCR and UACI, which indicates that a certain ability to resist differential attacks is provided by this algorithm. Table 9 [47,59] presents a comparison with other algorithms for both NPCR and UACI [60–63].

$$UACI(D_1, D_2) = \frac{1}{CK}\sum_{i=0}^{C}\sum_{j=0}^{K}\frac{|D_1(i,j) - D_2(i,j)|}{255 - 0} \times 100\% \quad (10)$$

Table 8. Differential attack test results.

Images	Sizes	Cipher Images (CR = 0.6)					
		NPCR (%)			UACI (%)		
		R	G	B	R	G	B
1.1	256 × 256 × 3	99.6094	99.6100	99.6187	33.4512	33.4969	33.4661
1.2	256 × 256 × 3	99.6151	99.6131	99.6098	33.5268	33.4695	33.5211
2.1	512 × 512 × 3	99.6167	99.6118	99.6135	33.4630	33.4661	33.5041
2.2	512 × 512 × 3	99.6094	99.6103	99.6102	33.4661	33.4947	33.4660
3.1	1024 × 1024 × 3	99.6092	99.6098	99.6117	33.4661	33.4651	33.4602
3.2	1024 × 1024 × 3	99.6099	99.6166	99.6099	33.4833	33.4601	33.4631
4.1	256 × 256		99.6147			33.4980	
4.2	512 × 512		99.6132			33.4699	

Table 9. Comparison of performance in resisting differential attacks with other algorithms.

Algorithms	Sizes	CR	NPCR (%)	UACI (%)
Reference [59]	512 × 512	0.75	99.6174	33.4570
Proposed	512 × 512	0.75	99.6174	33.4938
Reference [47]	256 × 256 × 3	0.6	99.6087	33.4815
Proposed	256 × 256 × 3	0.6	99.6127	33.6553
Theoretical value			99.6094	33.4635

5.3.2. Plaintext Attack

Plaintext attack is essential for encryption algorithms. In the plaintext attack tests, select both all black and white images for original images, as presented in Figure 11a. The results after compression–encryption and reconstruction are presented in Figure 11b,c. The results of decryption–reconstruction show visually no difference with plaintext images, indicating that it is resistant to plaintext attacks.

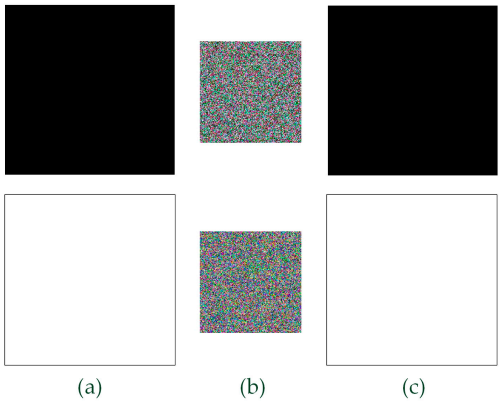


Figure 11. Result images of plaintext attacks: (**a**) plaintext images (**b**) cipher images (**c**) decrypted-reconstruction images.

5.4. Statistical Characteristics Analysis

5.4.1. Histogram

In this section, the distribution of image pixel points is analyzed by plotting histograms of images. It reflects the distribution of pixel-grey values in the image. The more uniform the distribution of the histogram of the cipher image, the better the encryption effect. For testing, two color images are chosen to observe the distribution of pixels in plaintext images and the decryption–reconstruction images presented in Figure 12 (R-channel in red, G-channel in green, B-channel in blue). Waving patterns are displayed in the histograms of original images, while the encrypted images (CR = 0.6) show relatively even patterns.

The histogram of the encrypted image maintains random distribution and uniformity and is significantly different from the histogram of the original image, indicating that the encryption algorithm is effective. This means that the encrypted images have a high degree of randomness. Information in the original images from the histogram is made extremely hard for attackers to access.

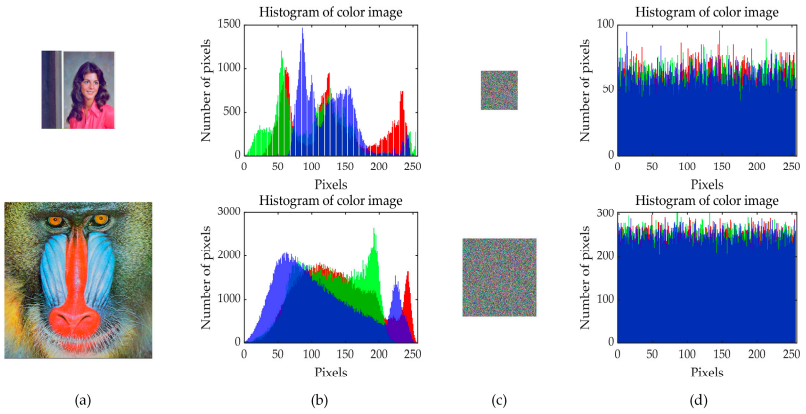


Figure 12. Histogram results: (**a**) original images, (**b**) histograms of original images, (**c**) cipher images, and (**d**) histograms of cipher images.

5.4.2. Correlation

Correlation analysis is the process of analyzing two or more elements of a variable that have a correlation in order to measure the closeness of the correlation between the variables. The two neighboring pixels in plaintext images are highly correlated in vertical (V), horizontal (H), and diagonal (D) directions, while in cipher images the correlation between neighboring pixels is low. The formula used to calculate the correlation of neighboring pixels of the image is shown in Equation (11).

$$\begin{cases} r_{xy} = \dfrac{U((x - U(x))(y - U(y)))}{\sqrt{V(x)}\sqrt{V(y)}} \\ U(x) = \dfrac{1}{T}\sum\limits_{i=1}^{N} x_i \\ V(x) = \dfrac{1}{T}\sum\limits_{i=1}^{N} (x_i - U(x))^2 \end{cases} \tag{11}$$

where the variable x is denoted by $U(x)$, $V(x)$ represents the variance of variable x, the grayscale values of x and y refer to the neighboring pixels, and the samples in total are represented by T.

The correlation coefficients for different plaintext images and compressed–encrypted images are listed in Table 10. Obviously, the correlation coefficients in the cipher images are nearly zero. The distributions of neighboring pixels for both plaintext and cipher images are illustrated in Figure 13. The cipher image has more scattered neighboring pixels in all three directions as compared to the plaintext image, as can be clearly seen in Figure 13. Thus, it is observed that related information with plaintext images can be well hidden by this encryption algorithm.

Table 10. Correlation coefficients for different images.

Images		Plaintext Images			Cipher Images (CR = 0.6)		
		H	**V**	**D**	**H**	**V**	**D**
1.1	R	0.9603	0.9724	0.9413	−0.0079	0.0007	−0.0160
	G	0.9659	0.9701	0.9492	−0.0065	0.0042	−0.0021
	B	0.9565	0.9570	0.9345	−0.0159	0.0031	0.0096

Table 10. *Cont.*

Images		Plaintext Images			Cipher Images (CR = 0.6)		
		H	V	D	H	V	D
1.2	R	0.9917	0.9809	0.9752	0.0046	−0.0209	−0.0235
	G	0.9867	0.9652	0.9532	0.0034	−0.0183	−0.0284
	B	0.9727	0.9555	0.9359	0.0095	−0.0228	−0.0138
2.1	R	0.8741	0.9295	0.8566	−0.0224	−0.0059	−0.0039
	G	0.7671	0.8646	0.7379	0.0020	−0.0042	0.0326
	B	0.8870	0.9037	0.8474	0.0106	0.0177	0.0185
2.2	R	0.9637	0.9534	0.9308	0.0286	−0.0050	0.0034
	G	0.9494	0.9457	0.9036	0.0065	−0.0026	0.0266
	B	0.9726	0.9740	0.9479	0.0077	0.0115	−0.0211
3.1	R	0.9316	0.9258	0.9044	−0.0203	−0.0230	−0.0151
	G	0.8781	0.8792	0.8491	−0.0074	0.0141	0.0218
	B	0.7981	0.8006	0.7407	0.0263	0.0002	0.0274
3.2	R	0.9244	0.9241	0.9020	−0.0116	−0.0225	0.0022
	G	0.9175	0.9194	0.8931	0.0092	0.0147	0.0399
	B	0.9058	0.9050	0.8860	0.0007	0.0068	0.0029
4.1		0.9746	0.9525	0.9358	−0.0015	−0.0057	−0.0089
4.2		0.8610	0.9317	0.8609	0.0060	−0.0148	−0.0002
All black					−0.0009	0.0174	0.0044
All white					−0.0061	0.0010	−0.0053

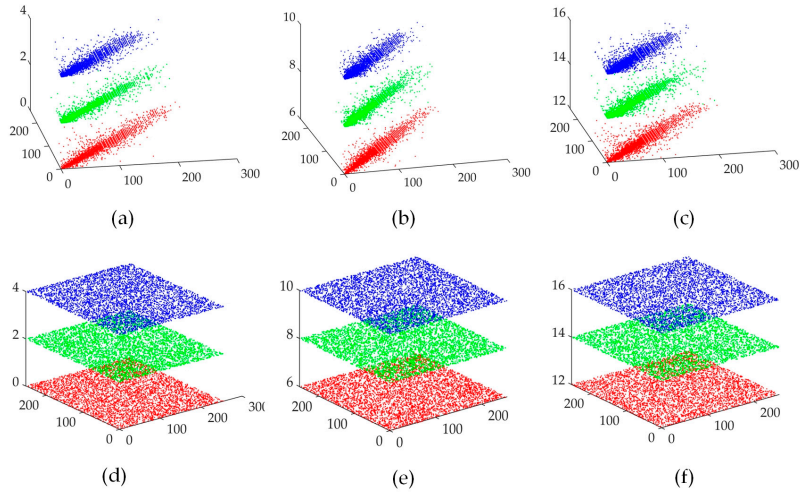


Figure 13. Correlation diagram of image "1.1" in the three directions of the original image: (**a**) H, (**b**) V, (**c**) D correlation diagram in three directions of cipher image, (**d**) H, (**e**) V, and (**f**) D.

5.4.3. Information Entropy

To measure the randomness of information, information entropy is a theory that has a value of 8 [64]. When a measured information entropy is close to the theoretical value, it indicates that the image pixels are well distributed. For the values of information entropy for different images encrypted by this algorithm refer to Table 11. It is concluded that the information entropy of the cipher image is very close to the theoretical value, illustrating that excellent effects of encryption are achieved.

Table 11. Information entropy for different images.

Images (Size)	Cipher Images (CR = 0.6)		
	R	G	B
1.1 (153.6 × 256 × 3)	7.9711	7.9812	7.9860
1.2 (153.6 × 256 × 3)	7.9922	7.9921	7.9923

Table 11. *Cont.*

Images (Size)	Cipher Images (CR = 0.6)		
	R	**G**	**B**
2.1 (307.2 × 512 × 3)	7.9979	7.9978	7.9980
2.2 (307.2 × 512 × 3)	7.9981	7.9981	7.9981
3.1 (614.4 × 1024 × 3)	7.9887	7.9840	7.9898
3.2 (614.4 × 1024 × 3)	7.9995	7.9995	7.9995
4.1 (153.6 × 256)		7.9915	
4.2 (153.6 × 256)		7.9981	
All black (153.6 × 256)		7.9911	
All white (153.6 × 256)		7.9922	

5.4.4. Homogeneity Analysis

Homogeneity analysis is used to measure the proximity of distributions on the Grey Level Co-occurrence Matrix (GLCM). The lower the homogeneity of the cipher image, the higher the security of the IE algorithm. The results of the homogeneity of different original images and their cipher images are shown in Table 12, which fully reflect the security of the algorithm.

Table 12. Homogeneity analysis results for different original images and their cipher images.

Image	Original Images			Cipher Images		
	R	**G**	**B**	**R**	**G**	**B**
1.1	0.9109	0.9246	0.9018	0.3883	0.3914	0.3890
1.2	0.9124	0.9060	0.9135	0.3925	0.3890	0.3901
2.1	0.7848	0.7598	0.7512	0.3906	0.3897	0.3894
2.2	0.8881	0.8803	0.8923	0.3909	0.3904	0.3896
3.1	0.8629	0.9378	0.9812	0.3892	0.3891	0.3892
3.2	0.7789	0.7841	0.8506	0.3894	0.3892	0.3902
4.1		0.9131			0.3888	
4.2		0.9048			0.3900	
All black		1			0.3919	
All white		1			0.3898	

5.5. Robustness

Noise attacks and shearing attacks are tested in this subsection to measure the robustness of this encryption algorithm.

5.5.1. Noise Attack

Noise attack is the process of interfering with or disrupting the transmission or identification of a signal by adding noise to the signal. In the cipher images, salt and pepper noise (SPN) of 0.01 or 0.05 is added, respectively, and decrypted results are presented in Figure 14. The results after adding SPN are successfully decrypted and are still visually recognizable. The PSNR of each image after adding noise is illustrated in Table 13. As can be seen from the figures, after suffering from different sizes of SPN after compression, the PSNR is still close to 30 dB, indicating that the algorithm can resist some level of noise attack.

Table 13. Decryption image quality with different intensities of noise attacks.

Image	PSNR (dB) SPN = 0.01			PSNR (dB) SPN = 0.05		
	R	**G**	**B**	**R**	**G**	**B**
1.1	29.0203	29.3532	29.3294	28.9485	29.1297	29.2689
1.2	29.5724	29.6824	29.5546	29.4373	29.2485	29.4983

Table 13. *Cont.*

Image	PSNR (dB) SPN = 0.01			PSNR (dB) SPN = 0.05		
	R	G	B	R	G	B
2.1	29.5873	29.4342	30.8453	29.0459	29.8942	29.9984
2.2	28.9384	29.8494	29.7484	29.3632	28.6482	28.4721
3.1	29.0232	28.8474	30.0018	28.0394	28.4824	29.0193
3.2	28.9384	29.4372	30.4382	28.3942	29.0001	29.2019
4.1		28.7533			27.8920	
4.2		28.7593			28.2001	

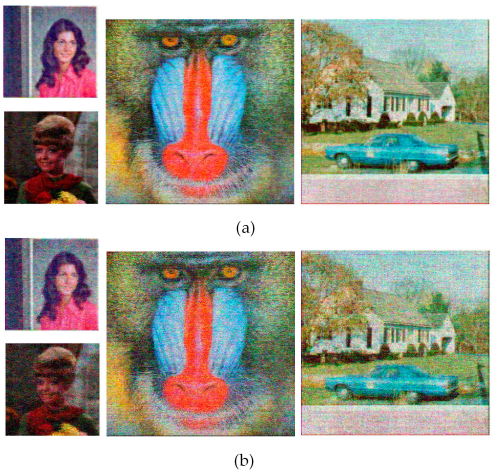


Figure 14. Restoration results after noise attacks for images '1.1', '1.2', '2.1', and '2.2': (**a**) 0.01 SPN and (**b**) 0.05 SPN.

5.5.2. Shearing Attack

During the transmission of image information, a part of this information may be maliciously sheared or even lost; therefore, the proposed algorithm should have the ability to resist shearing attacks. After cutting 12.5% of the cipher images, the results after being decrypted and reconstructed are shown in Figure 15. With a CR of 0.6, even though 12.5% of the images are cut off, it is still possible to recognize major information about original images after decryption–reconstruction. The test results show that a certain level of shearing attack can be resisted when using this algorithm.

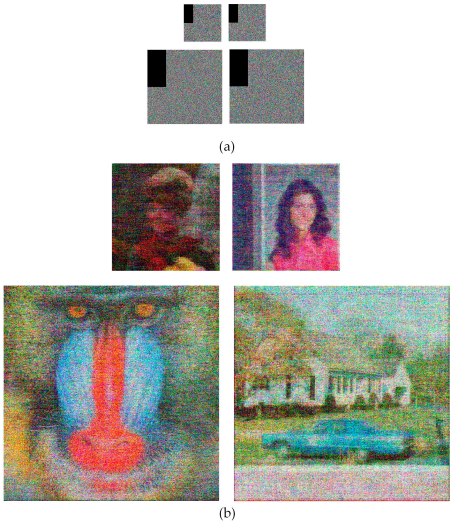


Figure 15. Results of shearing attack images: (**a**) size of shearing and (**b**) the decrypted–reconstructed images.

5.5.3. Speed Analysis

In addition to the security aspects, encryption speed is also an important factor in evaluating a good image encryption algorithm. The encryption speeds of the algorithm during encryption and decryption are shown in Table 14. It is obvious that the encryption speed of the algorithm in this paper reaches more than 349 kbit/s, which indicates that the algorithm has efficient encryption processing capability. The time comparison with different algorithms is shown in Table 15, from which the advantages in terms of either compression–encryption and decryption–reconstruction or the overall running efficiency of the algorithm can be clearly seen.

Table 14. Time and speed of image encryption and decryption (CR = 0.6).

Encryption	$\text{Time}_{\text{aver}}$ (s)	Speed (kbits/s)	Decryption	$\text{Time}_{\text{aver}}$ (s)	Speed (kbits/s)
CS	0.211	931.791	Reconstruction	15.492	12.691
Confusion	0.245	802.482	Inv-confusion	0.206	954.408
Diffusion	0.107	1837.458	Inv-diffusion	0.148	1328.432
Total	0.563	349.215	Total	15.846	12.407
Chaos iteration	0.051	3855.059			

Table 15. Comparison of time with different algorithms.

Algorithm	CR	Size	Compression–Encryption(s)	Decryption–Reconstruction(s)
Proposed	0.25	512 × 512	0.4884	4.0471
Reference [65]	0.25	512 × 512	1.3296	6.6809
Proposed	0.25	256 × 256	0.3447	0.7072
Reference [66]	0.25	256 × 256	0.3450	1.4950
Proposed	0.5	256 × 256	0.4858	2.8151
Reference [67]	0.5	256 × 256	0.9575	2.8240

6. Conclusions

A universal image compression–encryption algorithm based on DTM is proposed in this paper. Firstly, the image is compressed, then chaotic sequences iterated by TNN are used to perform DTM operations. Finally, the encrypted image can be obtained by utilizing chaotic sequences at the DNA level with confusion and diffusion algorithms. In the simulation tests, images of different sizes are chosen for encryption. The simulation results not only prove the effectiveness of the algorithm but also show that the algorithm can encrypt universal images of different sizes. As shown by performance tests, the algorithm has a large enough key space to resist brute force attacks. Moreover, this algorithm has the ability to withstand statistical analysis attacks and passes robustness tests. The security and reliability of the algorithm are proved. This compressed–encrypted image has an information entropy close to 8, its neighboring pixel correlation tends to 0, and both NPCR and UACI are basically in line with the theoretical values, while exhibiting a good reconstruction quality and also performing well in terms of encryption efficiency. The compression operation is performed by the algorithm before encryption, which reduces the workloads of encryption at the DNA level and improves the efficiency of the algorithm. The DTM is utilized in this algorithm to encrypt images by changing some gene sequences in the genome, thus protecting the privacy and security of the images, which brings a new idea to the field of image protection.

However, the algorithm also has certain a limitation, which is that the reconstruction time after decryption is long. In the future, this problem will be addressed to improve the efficiency of reconstruction without affecting the security of the algorithm. Further optimization of the design of the algorithm is required to improve the performance and effectiveness of privacy protection.

Author Contributions: Conceptualization, Y.C. and L.T.; Methodology, Y.C. and L.T.; Validation, Software L.T., Y.C., L.T., B.L. and X.X.; Formal analysis, Y.C. and L.T.; Investigation, X.X. and B.L.; Resources, Y.C., L.T., B.L. and X.X.; Data curation, L.T. and X.X.; Writing—original draft preparation, Y.C. and L.T.; Writing—review and editing, Y.C. and L.T.; Visualization, X.X. and B.L.; Supervision, Y.C.; Project administration, B.L. and X.X.; Funding acquisition, Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China (Grant No. 62061014), technological innovation projects in the field of artificial intelligence in Liaoning province (Grant Nos. 2023JH26/10300011), and basic scientific research projects in department of education of Liaoning Province (Grant No. JYTZD2023021).

Data Availability Statement: No new data were created or analyzed in this study.

Acknowledgments: The authors thank the referees for their detailed reading and comments that were both helpful and insightful.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu, C.; Xu, B. A night pavement crack detection method based on image-to-image translation. *Comput.-Aided Civ. Infrastruct. Eng.* **2022**, *37*, 1737–1753. [CrossRef]
2. Sun, F.; Lv, Z. A secure image encryption based on spatial surface chaotic system and AES algorithm. *Multimed. Tools Appl.* **2022**, *81*, 3959–3979. [CrossRef]
3. Sahoo, A.; Mohanty, P.; Sethi, P.C. Image encryption using RSA algorithm. In *Intelligent Systems: Proceedings of ICMIB 2021*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 641–652.
4. Zhang, X.; Wang, L.; Cui, G.; Niu, Y. Entropy-based block scrambling image encryption using DES structure and chaotic systems. *Int. J. Opt.* **2019**, *2019*, 3594534. [CrossRef]
5. Rehman, M.U.; Shafique, A.; Khan, K.H.; Hazzazi, M.M. Efficient and secure image encryption using key substitution process with discrete wavelet transform. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 101613. [CrossRef]
6. Peng, X.; Zeng, Y. Image encryption application in a system for compounding self-excited and hidden attractors. *Chaos Solitons Fractals* **2020**, *139*, 110044. [CrossRef]
7. Abdallah, A.A.; Farhan, A.K. A new image encryption algorithm based on multi chaotic system. *Iraqi J. Sci.* **2022**, *63*, 324–337. [CrossRef]
8. Abd, M.H.; Al-Suhail, G.A.; Tahir, F.R.; Ali Ali, A.M.; Abbood, H.A.; Dashtipour, K.; Jamal, S.S.; Ahmad, J. Synchronization of monostatic radar using a time-delayed chaos-based FM waveform. *Remote Sens.* **2022**, *14*, 1984. [CrossRef]
9. Lin, H.; Wang, C.; Sun, Y. A universal variable extension method for designing multiscroll/wing chaotic systems. *IEEE Trans. Ind. Electron.* **2023**, *71*, 7806–7818. [CrossRef]
10. Han, Z.; Al-Barakati, A.A.; Jahanshahi, H.; Mou, J. A novel circuit based on memristor-memcapacitor with extreme multistability. *Nonlinear Dyn.* **2024**, *112*, 4863–4877. [CrossRef]
11. Mou, J.; Han, Z.; Cao, Y.; Banerjee, S. Discrete Second-Order Memristor and Its Application to Chaotic Map. *IEEE Trans. Circuits Syst. II Express Briefs* **2024**, *71*, 2824–2828. [CrossRef]
12. Feng, W.; Wang, Q.; Liu, H.; Ren, Y.; Zhang, J.; Zhang, S.; Qian, K.; Wen, H. Exploiting newly designed fractional-order 3D Lorenz chaotic system and 2D discrete polynomial hyper-chaotic map for high-performance multi-image encryption. *Fractal Fract.* **2023**, *7*, 887. [CrossRef]
13. Wang, C.; Tang, D.; Lin, H.; Yu, F.; Sun, Y. High-dimensional memristive neural network and its application in commercial data encryption communication. *Expert Syst. Appl.* **2024**, *242*, 122513. [CrossRef]
14. Feng, W.; Zhao, X.; Zhang, J.; Qin, Z.; Zhang, J.; He, Y. Image encryption algorithm based on plane-level image filtering and discrete logarithmic transform. *Mathematics* **2022**, *10*, 2751. [CrossRef]
15. Ma, Y.; Mou, J.; Jahanshahi, H.; Alkhateeb, A.F.; Bi, X. Design and DSP implementation of a hyperchaotic map with infinite coexisting attractors and intermittent chaos based on a novel locally active memcapacitor. *Chaos Solitons Fractals* **2023**, *173*, 113708. [CrossRef]
16. Wang, X.; Cao, Y.; Li, H.; Li, B. A chaos-enhanced fractional-order chaotic system with self-reproduction based on a memcapacitor and meminductor. *Fractal Fract.* **2023**, *7*, 582. [CrossRef]
17. Ma, Y.; Mou, J.; Banerjee, S.; Miao, M. A quartic nonlinear flux-controlled memristor model and its application in chaotic system. *Appl. Comput. Math.* **2023**, *22*, 317–337.
18. Wang, L.; Zhou, N.; Sun, B.; Cao, Y.; Mou, J. Novel Self-Embedding Holographic Watermarking Image Encryption Protection Scheme. *Chin. Phys. B* **2024**, *33*, 050501. [CrossRef]
19. Sha, Y.; Mou, J.; Banerjee, S.; Zhang, Y. Exploiting flexible and secure cryptographic technique for multi-dimensional image based on graph data structure and three-input majority gate. *IEEE Trans. Ind. Inform.* **2023**, *20*, 3835–3846. [CrossRef]
20. Zhang, B.; Liu, L. Chaos-based image encryption: Review, application, and challenges. *Mathematics* **2023**, *11*, 2585. [CrossRef]

21. Tang, D.; Wang, C.; Lin, H.; Yu, F. Dynamics analysis and hardware implementation of multi-scroll hyperchaotic hidden attractors based on locally active memristive Hopfield neural network. *Nonlinear Dyn.* **2024**, *112*, 1511–1527. [CrossRef]
22. Deng, Q.; Wang, C.; Lin, H. Memristive Hopfield neural network dynamics with heterogeneous activation functions and its application. *Chaos Solitons Fractals* **2024**, *178*, 114387. [CrossRef]
23. Ren, L.; Qin, L.; Jahanshahi, H.; Mou, J. Infinitely Many Coexisting Attractors and Scrolls in a Fractional-Order Discrete Neuron Map. *Int. J. Bifurc. Chaos* **2023**, *33*, 2350197. [CrossRef]
24. Han, Z.; Sun, B.; Banerjee, S.; Mou, J. Biological neuron modeling based on bifunctional memristor and its application in secure communication. *Chaos Solitons Fractals* **2024**, *184*, 115020. [CrossRef]
25. Ma, T.; Mou, J.; Banerjee, S.; Cao, Y. Analysis of the functional behavior of fractional-order discrete neuron under electromagnetic radiation. *Chaos Solitons Fractals* **2023**, *176*, 114113. [CrossRef]
26. Cao, H.; Wang, Y.; Banerjee, S.; Cao, Y.; Mou, J. A discrete Chialvo–Rulkov neuron network coupled with a novel memristor model: Design, Dynamical analysis, DSP implementation and its application. *Chaos Solitons Fractals* **2024**, *179*, 114466. [CrossRef]
27. Mou, J.; Ma, T.; Banerjee, S.; Zhang, Y. A Novel Memcapacitive-Synapse Neuron: Bionic Modeling, Complex Dynamics Analysis and Circuit Implementation. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2024**, *71*, 1771–1780. [CrossRef]
28. Bao, B.; Wang, Z.; Hua, Z.; Chen, M.; Bao, H. Regime transition and multi-scroll hyperchaos in a discrete neuron model. *Nonlinear Dyn.* **2023**, *111*, 13499–13512. [CrossRef]
29. Zhang, Z.; Mou, J.; Banerjee, S.; Cao, Y. A chaotic hierarchical encryption/watermark embedding scheme for multi-medical images based on row–column confusion and closed-loop bi-directional diffusion. *Chin. Phys. B* **2024**, *33*, 020503. [CrossRef]
30. Alawida, M. A novel chaos-based permutation for image encryption. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 101595. [CrossRef]
31. Alexan, W.; Chen, Y.-L.; Por, L.Y.; Gabr, M. Hyperchaotic maps and the single neuron model: A novel framework for chaos-based image encryption. *Symmetry* **2023**, *15*, 1081. [CrossRef]
32. Akraam, M.; Rashid, T.; Zafar, S. A Chaos-Based Image Encryption Scheme Is Proposed Using Multiple Chaotic Maps. *Math. Probl. Eng.* **2023**, *2023*, 2003724. [CrossRef]
33. Liu, X.; Mou, J.; Zhang, Y.; Cao, Y. A new hyperchaotic map based on discrete memristor and meminductor: Dynamics analysis, encryption application, and DSP implementation. *IEEE Trans. Ind. Electron.* **2023**, *71*, 5094–5104. [CrossRef]
34. Maazouz, M.; Toubal, A.; Bengherbia, B.; Houhou, O.; Batel, N. FPGA implementation of a chaos-based image encryption algorithm. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 9926–9941. [CrossRef]
35. Fetteha, M.A.; Sayed, W.S.; Said, L.A. A lightweight image encryption scheme using dna coding and chaos. *Electronics* **2023**, *12*, 4895. [CrossRef]
36. Li, Z.; Peng, C.; Tan, W.; Li, L. A novel chaos-based image encryption scheme by using randomly DNA encode and plaintext related permutation. *Appl. Sci.* **2020**, *10*, 7469. [CrossRef]
37. Zefreh, E.Z. An image encryption scheme based on a hybrid model of DNA computing, chaotic systems and hash functions. *Multimed. Tools Appl.* **2020**, *79*, 24993–25022. [CrossRef]
38. Babaei, A.; Motameni, H.; Enayatifar, R. A new permutation-diffusion-based image encryption technique using cellular automata and DNA sequence. *Optik* **2020**, *203*, 164000. [CrossRef]
39. Fetteha, M.A.; Sayed, W.S.; Said, L.A.; Radwan, A.G. Chaos-based image encryption using dna manipulation and a modified arnold transform. In Proceedings of the International Conference on Model and Data Engineering, Cairo, Egypt, 21–24 November 2022; pp. 3–15.
40. Chen, X.; Mou, J.; Cao, Y.; Banerjee, S. Chaotic Multiple-Image Encryption Algorithm Based on Block Scrambling and Dynamic DNA Coding. *Int. J. Bifurc. Chaos* **2023**, *33*, 2350190. [CrossRef]
41. Fan, W.; Li, T.; Wu, J.; Wu, J. Chaotic Color Image Encryption Based on Eight-Base DNA-Level Permutation and Diffusion. *Entropy* **2023**, *25*, 1268. [CrossRef]
42. Wang, Q.; Zhang, X.; Zhao, X. Color image encryption algorithm based on bidirectional spiral transformation and DNA coding. *Phys. Scr.* **2023**, *98*, 025211. [CrossRef]
43. Kumari, P.; Mondal, B. Lightweight image encryption algorithm using NLFSR and CBC mode. *J. Supercomput.* **2023**, *79*, 19452–19472. [CrossRef]
44. Sha, Y.; Bo, S.; Yang, C.; Mou, J.; Jahanshahi, H. A chaotic image encryption scheme based on genetic central dogma and KMP method. *Int. J. Bifurc. Chaos* **2022**, *32*, 2250186. [CrossRef]
45. Gao, X.; Mou, J.; Banerjee, S.; Zhang, Y. Color-gray multi-image hybrid compression–encryption scheme based on BP neural network and knight tour. *IEEE Trans. Cybern.* **2023**, *53*, 5037–5047. [CrossRef] [PubMed]
46. Liu, X.; An, P.; Chen, Y.; Huang, X. An improved lossless image compression algorithm based on Huffman coding. *Multimed. Tools Appl.* **2022**, *81*, 4781–4795. [CrossRef]
47. Zhang, Z.; Cao, Y.; Jahanshahi, H.; Mou, J. Chaotic color multi-image compression-encryption/LSB data type steganography scheme for NFT transaction security. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 101839. [CrossRef]
48. Bao, P.; Xia, W.; Yang, K.; Chen, W.; Chen, M.; Xi, Y.; Niu, S.; Zhou, J.; Zhang, H.; Sun, H. Convolutional sparse coding for compressed sensing CT reconstruction. *IEEE Trans. Med. Imaging* **2019**, *38*, 2607–2619. [CrossRef] [PubMed]
49. Ahmed, N.; Natarajan, T.; Rao, K.R. Discrete cosine transform. *IEEE Trans. Comput.* **1974**, *100*, 90–93. [CrossRef]

50. Buiakova, O.; Baker, H.; Scott, J.; Farbman, A.; Kream, R.; Grillo, M.; Franzen, L.; Richman, M.; Davis, L.; Abbondanzo, S. Olfactory marker protein (OMP) gene deletion causes altered physiological activity of olfactory sensory neurons. *Proc. Natl. Acad. Sci. USA* **1996**, *93*, 9858–9863. [CrossRef] [PubMed]
51. Wang, X.; Su, Y. Image encryption based on compressed sensing and DNA encoding. *Signal Process. Image Commun.* **2021**, *95*, 116246. [CrossRef]
52. Xu, Q.; Sun, K.; Cao, C.; Zhu, C. A fast image encryption algorithm based on compressive sensing and hyperchaotic map. *Opt. Lasers Eng.* **2019**, *121*, 203–214. [CrossRef]
53. Chai, X.; Zheng, X.; Gan, Z.; Han, D.; Chen, Y. An image encryption algorithm based on chaotic system and compressive sensing. *Signal Process.* **2018**, *148*, 124–144. [CrossRef]
54. Gan, Z.; Chai, X.; Bi, J.; Chen, X. Content-adaptive image compression and encryption via optimized compressive sensing with double random phase encoding driven by chaos. *Complex Intell. Syst.* **2022**, *8*, 2291–2309. [CrossRef]
55. Hua, Z.; Zhou, Y.; Huang, H. Cosine-transform-based chaotic system for image encryption. *Inf. Sci.* **2019**, *480*, 403–419. [CrossRef]
56. Yang, F.; An, X. A new discrete chaotic map application in image encryption algorithm. *Phys. Scr.* **2022**, *97*, 035202. [CrossRef]
57. Xian, Y.; Wang, X. Fractal sorting matrix and its application on chaotic image encryption. *Inf. Sci.* **2021**, *547*, 1154–1169. [CrossRef]
58. Zhu, L.; Song, H.; Zhang, X.; Yan, M.; Zhang, T.; Wang, X.; Xu, J. A robust meaningful image encryption scheme based on block compressive sensing and SVD embedding. *Signal Process.* **2020**, *175*, 107629. [CrossRef]
59. Huang, X.; Dong, Y.; Ye, G.; Shi, Y. Meaningful image encryption algorithm based on compressive sensing and integer wavelet transform. *Front. Comput. Sci.* **2023**, *17*, 173804. [CrossRef]
60. Zhou, Z.; Xu, X.; Jiang, Z.; Sun, K. Multiple-Image Encryption Scheme Based on an N-Dimensional Chaotic Modular Model and Overlapping Block Permutation–Diffusion Using Newly Defined Operation. *Mathematics* **2023**, *11*, 3373. [CrossRef]
61. Wang, X.; Liu, L.; Song, M. Remote sensing image and multi-type image joint encryption based on NCCS. *Nonlinear Dyn.* **2023**, *111*, 14537–14563. [CrossRef]
62. Du, Y.; Long, G.; Jiang, D.; Chai, X.; Han, J. Optical image encryption algorithm based on a new four-dimensional memristive hyperchaotic system and compressed sensing. *Chin. Phys. B* **2023**, *32*, 114203. [CrossRef]
63. Zhang, X.; Liao, J. Multiple-image encryption algorithm based on 3D-LWT and dynamic stereo S-box. *Multimed. Tools Appl.* **2024**, *83*, 16337–16362. [CrossRef]
64. Wu, Y.; Zhou, Y.; Saveriades, G.; Agaian, S.; Noonan, J.P.; Natarajan, P. Local Shannon entropy measure with statistical tests for image randomness. *Inf. Sci.* **2013**, *222*, 323–342. [CrossRef]
65. Chai, X.; Fu, J.; Gan, Z.; Lu, Y.; Zhang, Y. An image encryption scheme based on multi-objective optimization and block compressed sensing. *Nonlinear Dyn.* **2022**, *108*, 2671–2704. [CrossRef]
66. Chai, X.; Wu, H.; Gan, Z.; Zhang, Y.; Chen, Y.; Nixon, K.W. An efficient visually meaningful image compression and encryption scheme based on compressive sensing and dynamic LSB embedding. *Opt. Lasers Eng.* **2020**, *124*, 105837. [CrossRef]
67. Luo, Y.; Liang, Y.; Zhang, S.; Liu, J.; Wang, F. An image encryption scheme based on block compressed sensing and Chen's system. *Nonlinear Dyn.* **2023**, *111*, 6791–6811. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.