Article

Remote-Sensing Image Encryption Algorithm Using the Advanced Encryption Standard

Xiaoqiang Zhang and Xuesong Wang *

School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221116, China; grayqiang@163.com
* Correspondence: wangxuesong@cumt.edu.cn; Tel.: +86-183-6125-5457

Received: 3 August 2018; Accepted: 29 August 2018; Published: 3 September 2018

Abstract: With the increasing use of multimedia in communications, the content security of remote-sensing images attracts much attention in both the academia and industry. The Advanced Encryption Standard (AES) is a famous symmetric cryptosystem. A symmetric remote-sensing image encryption algorithm using AES is presented. Firstly, to reduce the encryption times, the sender groups 16 pixel values together, and converts them into big integers; secondly, the sender encrypts big integers with AES and the chaotic system; finally, the encrypted image is obtained from encrypted big integers. Simulation data show that our algorithm exhibits both the high security and efficiency.

Keywords: image encryption; remote-sensing image; advanced encryption standard (AES); chaotic system

1. Introduction

Remote sensing is a non-contact and long-distance measuring technology, which acquires the information of an object or phenomenon. This technology plays a highly significant role in many fields, such as resource survey, environment protection, weather forecast and disaster evaluation. Most of remote-sensing images involve business secrets and even national security. Internet development and multimedia easy distribution make the content security of remote-sensing images become an important issue for scientists and engineers.

The chaotic system is a commonly applied means to encrypt images for its extreme sensitivity to initial values and parameters, ergodicity, etc. [1]. The experts of information security have designed lots of image encryption schemes with the chaotic systems recently [2–7]. Nevertheless, the security of the low-dimensional chaotic system is a little weak for its disadvantage of short code period. Researchers pay more attention to more complex chaotic systems, such as the high-dimensional chaotic systems and compound chaotic systems [8–13].

Advanced Encryption Standard (AES) is an excellent symmetric cryptosystem, which has the advantages of strong ability to resist attacks, so it is widely used in data encryption. At present, several experts of image encryption viewed that AES is fairly safe, but it isn’t suitable for encrypting image. The reason is that the image data is huge and has more redundancy than the text data [14]. However, Zhang et al. designed an image encryption algorithm with AES [15]. Authors group the pixel values of the plain image into data blocks with the size of 128 bits. They permute the first data block with the initial vector. After that, AES encrypts other data blocks with the cipher block chaining mode. Although authors stated that their algorithm can achieve high encryption speed, the efficiency can be further improved. Therefore, to protect the content of remote-sensing images, this paper presents a symmetric remote-sensing image encryption algorithm with AES and chaos. Experimental results and algorithm comparative analyses display that the proposed algorithm is desirable in terms of the security and efficiency.
The content structure of this paper is: Section 2 introduces AES, chaotic system, and secure hash algorithm in brief. A novel remote-sensing image encryption algorithm is designed in Section 3. Some similar algorithms are described in Section 4. Experiments are carried out in Section 5. Comparisons are drawn in Section 6. Section 7 makes the conclusions.

2. Theoretical Principle

2.1. AES

The National Institute of Standards and Technology (NIST) recruited a new data encryption standard for symmetric cryptosystem to be instead of the old version, i.e., Data Encryption Standard (DES) on 2 January 1997. Rijndael algorithm has been chosen by NIST and became the new advanced encryption standard, i.e., AES, on 2 October 2000 [16]. This algorithm is designed by Joan Daemen and Vincent Rijmen. AES is a block cipher, whose block size is 128 bits. AES core parameters are listed in Table 1. Each round is comprised of the four functions, i.e., SubBytes($·$), ShiftRows($·$), MixColumns($·$), and AddRoundKey($·$). Their detailed explanations can be found in Reference [16].

Table 1. Advanced Encryption Standard (AES) core parameters.

| Key Length | Block Size | Rounds |
|------------|------------|--------|
| 128        | 128        | 10     |
| 192        | 128        | 12     |
| 256        | 128        | 14     |

2.2. Chaotic System

The Piece-Wise Linear Chaotic Map (PWLCM) is defined by:

$$x_{i+1} = F_q(x_i) = \begin{cases} 
\frac{x_i}{q} & 0 \leq x_i < q \\
\frac{x_i - q}{q^2 - q} & q \leq x_i < 0.5 \\
F_q(1 - x_i) & 0.5 \leq x_i < 1 
\end{cases}$$

where $x_i \in [0, 1)$ and $q$ is the control parameter [17]. When $q \in (0, 0.5)$, this chaotic system can provide an excellent and random chaotic sequence, which is suitable for image encryption.

2.3. Secure Hash Algorithm

Secure Hash Algorithms (SHA) are a kind of hash functions, which is released by NIST. SHA is mainly applied to the integrity security services [18]. SHA-256 is a popular SHA, which outputs the message digest with the length of 256 bits.

The proposed algorithm uses SHA-256 to generate the initial value $x_0$ and variable $q$ of PWLCM. For the plain image, the proposed algorithm adopts SHA-256 to generate hash value $K$, which can be segmented into 8-bit parts, i.e.,:

$$K = k_1, k_2, \cdots, k_i, \cdots, k_{32}, \quad i = 3, 4, \cdots, 31$$

The variables $x_0$ and $q$ of PWLCM are computed by:

$$x_0 = \frac{k_1 \oplus k_2 \oplus \cdots \oplus k_i \cdots \oplus k_{16}}{255}, \quad i = 3, 4, \cdots, 15$$

where $\oplus$ denotes the exclusive OR (XOR) operation in the binary system.

$$q = \frac{k_{17} \oplus k_{18} \oplus \cdots \oplus k_i \cdots \oplus k_{32}}{510}, \quad i = 19, 20, \cdots, 31$$
3. Proposed Image Encryption Algorithm

For easy description, the sender and recipient are Alice and Bob, respectively. The following subsections describe the core technology of the proposed image encryption algorithm.

3.1. Sender’s Encryption Process

Figure 1 is the image encryption flowchart of the proposed algorithm. Alice performs the following steps to encrypt the plain image.

![Image Encryption Flowchart](image.png)

**Figure 1.** The encryption flowchart of the proposed algorithm.

**Step 1: Chaotic Sequence Generation**

Let the plain image be $I_{m \times n}$. The variables $x_0$ and $q$ of PWLCM can be generated with the method described in Section 2.3. Alice iterates Equation (1) $m \times n$ times with $x_0$ and $q$, and she obtains a chaotic sequence $X = \{x_1\}_{m \times n}$.

**Step 2: Chaotic Image Generation**

If 16 digits exist after the decimal point for all the $x_i \in X_{mn}$, Alice computes:

$$y_i = \text{mod} \left( x_i \times 10^{16}, 256 \right)$$

(5)

where $Y = \{y_i\}_{m \times n}$ and mod(·) is the modulo operation. After that the sender can convert the vector $Y$ into a matrix $C_{m \times n}$, named as the chaotic image.

**Step 3: Group Pixels**

The pixel value in binary form is an 8-bit number. Here, Alice uses AES-256 to encrypt the 128-bit data blocks. Therefore, Alice groups 16 pixel values together to form a big integer number with the 128-bit length. The first big integer is:

$$b_1 = \sum_{i=1}^{16} b_i 2^{(i-1)}$$

(6)

where $b_{1,1}, b_{1,2}, \ldots, b_{1,16}$ denote the pixels of $I$ in binary form. Alice can obtain $(m \times n)/16$ plain big integers $\{b_1, b_2, \ldots, b_{(m \times n)/16}\}$ for $I$ in total. Similarly, Alice can also obtain $(m \times n)/16$ chaotic big integers $\{r_1, r_2, \ldots, r_{(m \times n)/16}\}$ for $C$. 
Step 4: AES Encryption
Alice calculates:

\[ c_1 = \text{AES}_e(K_1, b_1) \oplus r_1 \]  
\[ c_i = b_i \oplus c_{i-1} \oplus r_i, \quad i = 2, 3, \ldots, (m \times n) / 16 \]  

where \( \{c_i\} \) are encrypted big integers, \( \text{AES}_e(\cdot) \) is the encryption function of AES, whose input parameters are the key \( K_1 \) and \( b_1 \).

Step 5: Recovering Pixels
Alice divides \( c_i, i = 1, 2, \ldots, (m \times n) / 16 \) into 16 parts, and converts them into pixel values \( \{p^1_1, p^1_2, \ldots, p^1_i, \ldots, p^1_{16}\}, j = 3, 4, \ldots, 15 \) as shown in Figure 2. According to the pixel positions, she rebuilds these pixel values into an encrypted image \( E \) with the size \( m \times n \).

\[ c_i \rightarrow \begin{array}{c} 8 \text{ bits} \\ 8 \text{ bits} \\ \cdots \\ 8 \text{ bits} \end{array} \]
\[ \downarrow \quad \downarrow \quad \cdots \quad \downarrow \quad \downarrow \]
\[ p^1_1, p^1_2, \ldots, p^1_{16} \]

**Figure 2.** Converting an encrypted big integer into pixel values.

### 3.2. Recipient’s Decryption Process

Figure 3 is the image decryption flowchart of the proposed algorithm. Bob decrypts the encrypted image \( E \) with the steps as follows.

**Figure 3.** The decryption flowchart of the proposed algorithm.

Step 1: Chaotic Sequence Generation
Bob iterates Equation (1) \( m \times n \) times with the variables \( x_0 \) and \( q \) of PWLCM, and obtains a chaotic sequence \( X = \{x_i\}_{m \times n} \).

Step 2: Chaotic Image Generation
Bob can get the \( Y = \{y_i\}_{m \times n} \) with Equation (5), and convert it into a matrix \( C_{m \times n} \).

Step 3: Group Pixels
Bob groups 16 pixel values together to form a big integer number with the 128-bit length. For the encrypted image $E$, Bob can obtain $(m \times n)/16$ encrypted big integers $\{c_i\}$. For the chaotic image $C$, Alice can also obtain $(m \times n)/16$ chaotic big integers $\{r_i\}$.

**Step 4: AES Decryption**

Bob computes:

$$d_1 = \text{AES}_d(K_1, c_1 \oplus r_1)$$

$$d_i = c_i \oplus c_{i-1} \oplus r_i, \quad i = 2, 3, \ldots, (m \times n)/16$$

where $\{d_i\}$ are decrypted big integers, and $\text{AES}_d(\cdot)$ is the decryption function of AES, whose input parameters are the key $K_1$ and $c_1 \oplus r_1$.

**Step 5: Recovering Pixels**

Bob divides $d_i$, $i = 1, 2, \ldots, (m \times n)/16$ into 16 parts, and converts them into pixel values. According to the pixel positions, he rebuilds these pixel values into a decrypted image $D$ with the size $m \times n$.

### 4. Existing Similar Algorithms

#### 4.1. Zhang’s Algorithm

Zhang et al. designed an image encryption scheme with AES (short for Zhang’s algorithm) [15]. Alice performs the following steps to encrypt the plain image $I_{m \times n}$.

- **Step 1**: divide $I_{m \times n}$ into $t$ pixel blocks with the size $4 \times 4$ to form 128 bits;
- **Step 2**: Tent map is used to generate the initial vector $IV$ of AES;
- **Step 3**: for the first block $I_1$ of $I_{m \times n}$, encrypt it by:

$$C_1 = \text{AES}_e(K, IV \oplus I_1)$$

where $K$ is the secret key, and $C_1$ is the first encrypted block;

- **Step 4**: for the $i$th block $I_i$ of $I_{m \times n}$, encrypt it by:

$$C_i = \text{AES}_e(K, C_{i-1} \oplus I_i), \quad i = 2, 3, \ldots, t$$

where $C_i$ is the $i$th encrypted block.

#### 4.2. Kalubandi’s Algorithm

Kalubandi et al. designed an image encryption scheme using AES and visual cryptography (short for Kalubandi’s algorithm) [19]. Alice performs the following steps to encrypt the plain image $I_{m \times n}$.

- **Step 1**: encode $I_{m \times n}$ with the Base64 standard;
- **Step 2**: initiate the AES 256-bit key;
- **Step 3**: encrypt the Base64 encoded text with AES;
- **Step 4**: convert the cipher text of AES into the encryption image.

#### 4.3. Hraoui’s Algorithm

Hraoui et al. designed an image encryption scheme with AES (short for Hraoui’s algorithm) [20]. Alice performs the following steps to encrypt the plain image.

- **Step 1**: segment $I_{m \times n}$ into $4 \times 4$ pixel blocks;
- **Step 2**: convert these blocks into 128-bit sequences;
- **Step 3**: encrypt these 128-bit sequences with AES;
- **Step 4**: convert the cipher text of AES into $4 \times 4$ encrypted pixel blocks;
Step 5: combine these encrypted pixel blocks into the encrypted image.

5. Experiments

The experimental purpose is to encrypt the plain image with the proposed algorithm and three similar algorithms, i.e., Zhang’s algorithm, Kalubandi’s algorithm and Hraoui’s algorithm. The plain image is Airfield as shown in Figure 4a, whose size is 512 \times 512. The computer configuration used in the experiments is shown as follows: Yoga 2 notebook PC of Lenovo Group, Beijing, China, Intel M-5Y71@1.20 GHz Processor, 8 GB RAM, Window 8 operating system and Matlab R2016a. \( K \) is the SHA-256 value of Airfield. Therefore, the variables \( x_0 \) and \( q \) of PWLCM can be calculated with the method described in Section 2.3. For the proposed algorithm, Zhang’s algorithm, Kalubandi’s algorithm and Hraoui’s algorithm, the AES keys are \( K_1, K_2, K_3, K_4 \) respectively. The above variable values are given in Table 2.

| Variables | Values |
|-----------|--------|
| \( K \)   | B40608C3183BC9E3D9933DE8B4DB1366FA970574CD6C12D8D9AD0E7DCF37C0FE |
| \( x_0 \) | 0.960784313725490 |
| \( q \)   | 0.150980392156863 |
| \( K_1 \) | 84AAC 12F541B666ECFC2A83C676908C8B381B1F1EF5B2B433F362493F52C |
| \( K_2 \) | 1D7EB543FF740AA2647E8900C0D512D7B7F141AE0F84B16941D6D7D587682F2 |
| \( K_3 \) | 453CFC8A74A14B81F093F9294B817FCE361DA5014041BD775BD879F94F1E |
| \( K_4 \) | CA9A7C77EA9A03627B40A335FEC353098 |

For the proposed algorithm, Figure 4b is its encrypted image. Bob can decrypt the encrypted image with the key to obtain the plain image. The decrypted image is the same with Figure 4a. For three similar algorithms, the encrypted images are shown in Figure 4c–e, respectively. These encrypted images are very chaotic. Therefore, all these four algorithms have excellent encryption effect.

![Figure 4. Cont.](image-url)
Figure 4. Airfield and its corresponding encryption images. (a) Airfield, (b) encryption image of the proposed algorithm, (c) encryption image of Zhang’s algorithm, (d) encryption image of Hraoui’s algorithm, and (e) encryption image of Kalubandi’s algorithm.

6. Algorithm Analyses

For an excellent image encryption algorithm, it can resist several commonly used attacks, such as the brute-force attack and differential attack. This paper analyzes the performance of the proposed algorithm in terms of the key space, histogram, correlation, differential attack, information entropy, and encryption efficiency.

6.1. Key Space Analysis

For a desirable encryption algorithm, its key space should be large enough to resist the brute-force attack. (1) For Zhang’s algorithm, the AES key is also the key of Zhang’s algorithm. This algorithm security main depends on the used key length, i.e., 128, 192, or 256 bits. Therefore, the size of key space is also \(2^{128} \approx 3.4 \times 10^{38}, 2^{192} \approx 6.3 \times 10^{57},\) or \(2^{256} \approx 1.16 \times 10^{77};\) (2) For Kalubandi’s algorithm, authors adopt AES-256 during the process of image encryption. Therefore, the key space is \(2^{256} \approx 1.16 \times 10^{77};\) (3) For Hraoui’s algorithm, authors adopt AES-128 during the process of image encryption. Therefore, the key space is \(2^{128} \approx 3.4 \times 10^{38};\) (4) for the proposed algorithm, the keys are the AES key, the variables \(x_0\) and \(q\) of PWLCM. In our experiment, we used AES-256 to encrypt the plain image. If 16 digits exists after the decimal point for the key, then the key space is \(10^{32} \times 2^{256} \approx 1.16 \times 10^{109}.\) Therefore, the proposed algorithm has the largest key space to withstand the brute-force attack.

6.2. Histogram Analysis

The histogram can reflect the statistical feature of pixel value distribution. For a desirable encryption scheme, the histogram of the encryption image is always uniform [21]. Figure 5a is the histogram of Airfield, and Figure 5b–e are the histograms of Figure 4b–e. The experimental results show that all the histograms of Figure 4b–e are distributed uniformly, which are totally different from Airfield’s. Therefore, all these four algorithms can effectively resist the statistical attack.
6.3. Correlation Analysis

To evaluate the performance of pixel correlation, we carry out some simulations. The correlation coefficient of each pair is calculated by:

$$r_{x,y} = \frac{E(x - E(x))E(y - E(y))}{\sqrt{D(x)D(y)}}$$  \hspace{1cm} (13)
\[ E(x) = \frac{1}{N} \sum_{i=1}^{N} x_i \]  
\[ D(x) = \frac{1}{N} \sum_{i=1}^{N} (x_i - E(x))^2 \]

where \( E(\cdot) \) is the expectation function and \( D(\cdot) \) is the variance function.

For the proposed algorithm, 5000 pairs of adjacent pixels are randomly chosen from Figure 4a–e. For Figure 4a,b, Figure 6 reflects their relevance for adjacent pixels in the horizontal, vertical, and diagonal directions, respectively.

**Figure 6.** Pixel relevance of Figure 4a,b: (a) horizontal direction in Figure 4a; (b) horizontal direction in Figure 4b; (c) vertical direction in Figure 4a; (d) vertical direction in Figure 4b; (e) diagonal direction in Figure 4a; and (f) diagonal direction in Figure 4b.

For the proposed algorithm, Zhang’s algorithm, Kalubandi’s algorithm, and Hraoui’s algorithm have their correlation coefficients of Figure 4a–e given in Table 3. Experimental data display that...
the values of the plain image are close to 1, but the values of all the encrypted images are close to 0. Therefore, all these four algorithms can destroy the pixel relevance completely.

### Table 3. Correlation coefficients.

| Directions     | Horizontal | Vertical | Diagonal |
|----------------|------------|----------|----------|
| Plain image    | 0.9402     | 0.9423   | 0.9032   |
| The proposed algorithm | −0.0047    | 0.0025   | 0.0014   |
| Zhang’s algorithm | −0.0039    | 0.0022   | −0.0020  |
| Kalubandi’s algorithm | −0.0012    | 0.0007   | 0.0034   |
| Hraoui’s algorithm | −0.0026    | −0.0009  | 0.0007   |

6.4. Differential Attack Analysis

The differential attack is used to check the plaintext sensitivity for an image encryption algorithm [22]. Therefore, if we make a slight change to the plain image, a desirable encryption algorithm can spread this influence over the whole encryption process. To evaluate the ability to resist the differential attack, the Number of Pixels Change Rate (NPCR) and Unified Average Changing Intensity (UACI) are defined by:

\[
NPCR = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f(i, j)}{m \times n} \times 100\% \tag{16}
\]

\[
UACI = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} |I'(i, j) - I''(i, j)|}{255 \times m \times n} \times 100\% \tag{17}
\]

where \(I'(i, j)\) is the encryption image of the plain image, \(I''(i, j)\) is the encryption image of the modified plain image, and \(f(i, j)\) is defined by:

\[
f(i, j) = \begin{cases} 0 & I'(i, j) = I''(i, j) \\ 1 & I'(i, j) \neq I''(i, j) \end{cases}
\tag{18}
\]

To evaluate the performance of the plaintext sensitivity, the simulation changes the pixel value \(I(1, 1)\) of the plain image into 200. For the proposed algorithm, Zhang’s algorithm, Kalubandi’s algorithm, and Hraoui’s algorithm, have their NPCR and UACI results of Figure 4a listed in Table 4. For the proposed algorithm and Zhang’s algorithm, NPCR and UACI results are large. Therefore, their plaintext sensitivity is very strong. However, for Kalubandi’s algorithm and Hraoui’s algorithm, NPCR and UACI results are 0. The reason is that both the proposed algorithm and Zhang’s algorithm use the mode of cipher block chaining to encrypt Figure 4a. Meanwhile, for the proposed algorithm, the variable \(x_0\) and \(q\) of PWLCM are generated by the SHA-256 value of Figure 4a. Even if two images are very similar, but only one bit is different. Their hash values of SHA-256 are completely different [21]. The results imply that both the proposed algorithm and Zhang’s algorithm are strong to withstand the differential attack.

### Table 4. Number of Pixels Change Rate (NPCR) and Unified Average Changing Intensity (UACI) results for Figure 4a.

| Algorithms          | NPCR  | UACI  |
|---------------------|-------|-------|
| The proposed algorithm | 99.62%| 33.38%|
| Zhang’s algorithm    | 99.36%| 33.14%|
| Kalubandi’s algorithm | 0     | 0     |
| Hraoui’s algorithm  | 0     | 0     |
6.5. Information Entropy Analysis

Information entropy can reflect the indeterminacy of image information. For an ideal encryption image, its information entropy is close to 8 [23]. For the gray image \( I \), we describe the information entropy as:

\[
H(m) = -\sum_{i=0}^{255} P(m_i) \log_2 P(m_i)
\]  

(19)

where \( m_i \) is the \( i \)th gray level, and \( P(m_i) \) is the emergence probability of \( m_i \).

For the plain image, its entropy value is 7.1206. For the encrypted images of the proposed algorithm, Zhang’s algorithm, Kalubandi’s algorithm and Hraoui’s algorithm, their entropy values are given in Table 5. Simulation data display that all these four algorithms have the ability to resist the statistical attack.

Table 5. Entropy values.

| Algorithms          | Entropy |
|---------------------|---------|
| The proposed algorithm | 7.9991 |
| Zhang’s algorithm    | 7.9994 |
| Kalubandi’s algorithm | 7.9994 |
| Hraoui’s algorithm   | 7.9991 |

6.6. Encryption Efficiency Analysis

The encryption efficiency is the significant performance for an encryption algorithm. The core encryption times can directly affect the encryption speed. For the proposed algorithm, Zhang’s algorithm, Kalubandi’s algorithm, and Hraoui’s algorithm have their core operation is AES encryption times. The AES encryption times for these algorithms are given in Table 6. The unit is times. A 128-bit data block is encrypted with the encryption function of AES, i.e., \( \text{AES}(\cdot) \), viewed as once AES encryption times. The data in Table 6 view that the proposed algorithm can obviously reduce AES encryption times. We analyze the AES encryption times of these algorithms in detail as follows.

Table 6. Comparison on the AES encryption times (unit: times).

| Algorithms          | AES Encryption Times |
|---------------------|----------------------|
| Zhang’s algorithm   | 16,384               |
| Kalubandi’s algorithm | 21,846               |
| Hraoui’s algorithm  | 16,384               |
| The proposed algorithm | 1                   |

(1) In Zhang’s algorithm, authors divide the pixel values into pixel blocks, and then encrypt them with AES. For the plain image with the size 512 × 512, it has 262,144 (i.e., 512 × 512) pixel values. These pixel values can be converted into 16,384 (i.e., 262,144/16) pixel blocks. Therefore, to protect the plain image, this algorithm should encrypt 16,384 pixel blocks with Equation (11) or (12), i.e., 16,384 AES encryption times;

(2) In Kalubandi’s algorithm, authors encode the plain image with the Base64 standard. And then they encrypt the Base64 encoded text with AES. For the plain image with the size 512 × 512, it has 262,144 pixel values. Each Base64 code represents with 6 bits of the pixel value, so three 8-bit pixel values can be represented by four 6-bit Base64 codes. If there are only two pixel values, they can be represented by three Base64 codes. If there is only one pixel value, it can be represented by two Base64 codes [24]. Therefore, for the plain image, 262,144 pixel values can be encoded with 349,526 (i.e., (62,143/3 × 4) + 2) Base64 codes, where 262,143 is multiples of 3,262,143 pixel values can be encoded with 349,524 Base64 codes, and the last pixel value can be encoded with 2 Base64 codes. The plain...
image can be encoded with 349,526 Base64 codes in total. These Base64 codes can be converted into 21,846 (i.e., \((349,520/16) + 1\)) 128-bit data blocks, where 349,520 is multiples of 16. The 349,520 Base64 codes can be converted with 21,846 128-bit data blocks, and the last 6 Base64 codes can be converted with one 128-bit data block. Therefore, to protect the plain image, this algorithm should encrypt 21,846 128-bit data blocks with AES\(_e(\cdot)\), i.e., 21,846 AES encryption times;

(3) In Hraoui’s algorithm, authors divide the image into \(4 \times 4\) pixel blocks, and then encrypt them with AES directly. The plain image with the size 512 \(\times\) 512 can be divided into 16,384 (i.e., \((512 \times 512)/16\)) pixel blocks. Therefore, to protect the plain image, this algorithm should encrypt 16,384 pixel blocks with AES\(_e(\cdot)\), i.e., 16,384 AES encryption times;

(4) In the proposed algorithm, Alice groups 16 pixel values together to form a big integer number with the 128-bit length. We consider that the AES operation is more complex than the XOR operation. The proposed algorithm only encrypts the first big integer with Equation (7), i.e., only once AES encryption times, and encrypt other big integers with the previous encrypted big integer and chaotic big integers, i.e., Equation (8).

7. Conclusions

To protect the remote-sensing image, this paper presents an image encryption algorithm with AES and chaos. Simulation data display that the proposed algorithm is very strong to resist several commonly used attacks. Comparing with three similar algorithms, the superiority of the proposed algorithm is obvious in terms of the security and encryption speed.

**Author Contributions:** Conceptualization, X.Z.; Methodology, X.Z.; Software, X.Z.; Validation, X.Z.; Formal Analysis, X.Z.; Investigation, X.Z.; Resources, X.W.; Data Curation, X.Z.; Writing-Original Draft Preparation, X.Z.; Writing-Review & Editing, X.Z.; Visualization, X.Z.; Supervision, X.W.; Project Administration, X.Z.; Funding Acquisition, X.Z.

**Funding:** This research was funded by the National Natural Science Foundation of China grant number [61501465].

**Acknowledgments:** The research work of this paper is supported by the National Natural Science Foundation of China (61501465). Authors would like to express their sincere thanks to the anonymous reviewers and editors for their constructive comments and suggestions.

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

**References**

1. Huang, X.; Ye, G. An image encryption algorithm based on hyper-chaos and DNA sequence. *Multimed. Tools Appl.* **2014**, *72*, 57–70. [CrossRef]
2. 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]
3. Lan, R.; He, J.; Wang, S.; Gu, T.; Luo, X. Integrated chaotic systems for image encryption. *Signal Process.* **2018**, *147*, 133–145. [CrossRef]
4. Parvaz, R.; Zarebnia, M. A combination chaotic system and application in color image encryption. *Opt. Laser Technol.* **2018**, *101*, 30–41. [CrossRef]
5. Teng, L.; Wang, X.; Meng, J. A chaotic color image encryption using integrated bit-level permutation. *Multimed. Tools Appl.* **2018**, *77*, 6883–6896. [CrossRef]
6. Liu, L.; Hao, S.; Lin, J.; Wang, Z.; Hu, X.; Miao, S. Image block encryption algorithm based on chaotic maps. *IET Signal Process.* **2018**, *12*, 22–30. [CrossRef]
7. Pak, C.; Huang, L. A new color image encryption using combination of the 1D chaotic map. *Signal Process.* **2017**, *138*, 129–137. [CrossRef]
8. Ran, Q.; Wang, L.; Ma, J.; Tan, L.; Yu, S. A quantum color image encryption scheme based on coupled hyper-chaotic Lorenz system with three impulse injections. *Quantum Inf. Process.* **2018**, *17*, 1–30. [CrossRef]
9. Gong, L.; Deng, C.; Pan, S.; Zhou, N. Image compression-encryption algorithms by combining hyper-chaotic system with discrete fractional random transform. *Opt. Laser Technol.* **2018**, *103*, 48–58. [CrossRef]
10. Zhang, J.; Hou, D.; Ren, H. Image encryption algorithm based on dynamic DNA coding and Chen’s hyperchaotic system. Math. Probl. Eng. 2016, 2016, 1–11. [CrossRef]
11. Zhu, H.; Zhang, X.; Yu, H.; Zhao, C.; Zhu, Z. An image encryption algorithm based on compound homogeneous hyper-chaotic system. Nonlinear Dyn. 2017, 89, 61–79. [CrossRef]
12. Zhang, X.; Nie, W.; Ma, Y.; Tian, Q. Cryptanalysis and improvement of an image encryption algorithm based on hyper-chaotic system and dynamic S-box. Multimed. Tools Appl. 2017, 76, 15641–15659. [CrossRef]
13. Kaur, M.; Kumar, V. Efficient image encryption method based on improved Lorenz chaotic system. Electron. Lett. 2018, 54, 562–563. [CrossRef]
14. Hua, Z.; Jin, F.; Xu, B.; Huang, H. 2D Logistic-Sine-coupling map for image encryption. Signal Process. 2018, 149, 148–161. [CrossRef]
15. Zhang, Y.; Li, X.; Hou, W. A fast image encryption scheme based on AES. In Proceedings of the 2nd International Conference on Image, Vision and Computing (ICIVC), Chengdu, China, 2–4 June 2017; pp. 624–628.
16. Daemen, J.; Rijmen, V. The Design of Rijndael: AES-The Advanced Encryption Standard; Springer: Heidelberg, Germany, 2013.
17. Wang, X.; Xu, D. A novel image encryption scheme based on Brownian motion and PWLCM chaotic system. Nonlinear Dyn. 2015, 75, 345–353. [CrossRef]
18. Guesmi, R.; Farah, M.A.B.; Kachouri, A.; Samet, M. A novel chaos-based image encryption using DNA sequence operation and Secure Hash Algorithm SHA-2. Nonlinear Dyn. 2015, 83, 1123–1136. [CrossRef]
19. Kalubandi, V.K.P.; Vaddi, H.; Ramineni, V.; Loganathan, A. A novel image encryption algorithm using AES and visual cryptography. In Proceedings of the 2nd International Conference on Next Generation Computing Technologies (NGCT), Dehradun, India, 14–16 October 2016; pp. 808–813.
20. Hraoui, S.; Gmira, F.; Jarar, A.O.; Satori, K.; Saaidi, A. Benchmarking AES and chaos based logistic map for image encryption. In Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications (AICCSA), Ifrane, Morocco, 27–30 May 2013; pp. 1–4.
21. Chai, X.; Chen, Y.; Broyde, L. A novel chaos-based image encryption algorithm using DNA sequence operations. Opt. Lasers Eng. 2017, 88, 197–213. [CrossRef]
22. Belazi, A.; El-Latif, A.A.A.; Belghith, S. A novel image encryption scheme based on substitution-permutation network and chaos. Signal Process. 2016, 128, 155–170. [CrossRef]
23. Tang, Z.; Zhang, X.; Lan, W. Efficient image encryption with block shuffling and chaotic map. Multimed. Tools Appl. 2015, 74, 5429–5448. [CrossRef]
24. Wikipedia, The Free Encyclopedia: Base64. Available online: https://en.wikipedia.org/wiki/Base64 (accessed on 21 August 2018).