A novel color image encryption algorithm based on genetic recombination and the four-dimensional memristive hyperchaotic system*

Xiu-Li Chai(柴秀丽)[1,2,†], Zhi-Hua Gan(甘志华)[3], Yang Lu(路杨)[4], Miao-Hui Zhang(张苗辉)[1], and Yi-Ran Chen(陈怡然)[2]

[1] *School of Computer and Information Engineering, Institute of Image Processing and Pattern Recognition, Henan University, Kaifeng* 475004, *China*
[2] *Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA* 15261, *USA*
[3] *School of Software, Henan University, Kaifeng* 475004, *China*
[4] *Research Department, Henan University, Kaifeng* 475004, *China*

(Received 3 April 2016; revised manuscript received 6 June 2016; published online 25 August 2016)

Recently, many image encryption algorithms based on chaos have been proposed. Most of the previous algorithms encrypt components *R*, *G*, and *B* of color images independently and neglect the high correlation between them. In the paper, a novel color image encryption algorithm is introduced. The 24 bit planes of components *R*, *G*, and *B* of the color plain image are obtained and recombined into 4 compound bit planes, and this can make the three components affect each other. A four-dimensional (4D) memristive hyperchaotic system generates the pseudorandom key streams and its initial values come from the SHA 256 hash value of the color plain image. The compound bit planes and key streams are confused according to the principles of genetic recombination, then confusion and diffusion as a union are applied to the bit planes, and the color cipher image is obtained. Experimental results and security analyses demonstrate that the proposed algorithm is secure and effective so that it may be adopted for secure communication.

Keywords: genetic recombination, memristive hyperchaotic system, color image encryption, confusion, diffusion

PACS: 05.45.Gg, 05.45.–a, 05.45.Vx **DOI:** 10.1088/1674-1056/25/10/100503

1. Introduction

With the rapid development of the Internet and computers, more and more multimedia information, such as color images, video, and mp3 are transmitted and stored through the Internet. Information security, especially image information security has become an important problem. Image encryption technology is an efficient method to protect digital images. However, some traditional encryption methods, such as data encryption standard (DES), international data encryption algorithm (IDES) and advanced encryption standard (AES) are not suitable for image encryption due to their intrinsic features.[1] Recently, many image encryption technologies based on DNA, compressive sensing, Gyrator transform, chaos, quantum computation and others have been introduced. The image encryption based on chaos has proven to be superior due to the special characteristics of chaotic systems, such as ergodicity, sensitive dependences on initial conditions and parameters, random-like behavior and mixing effect, and many chaos-based image encryption algorithms have been presented.[2–17] In 1998, Fridrich[4] presented that a chaos-based image encryption method should include the permutation and diffusion process. In the permutation phase, bit-level permutation of images used in our paper may change the positions and values of the plain image pixels simultaneously, thus it is superior to pixel-level confusion.[5–8] Zhu *et al.*[5] introduced a symmetric image encryption scheme, employed the Arnold cat map for bit-level permutation and the logistic map for diffusion. Zhang and Wang[6] put forward a novel image encryption algorithm by using a mixed linear-nonlinear coupled map lattice and by using the bit-level permutation strategy to make the lower bit planes and higher bit planes of plain images permute mutually without any extra storage space. However, some encryption schemes have been broken and proven to be unsecure,[9–12] for the fixed secure key regardless of the plain image that makes the methods vulnerable to the chosen plaintext attack, thus, more secure image encryption algorithms need to be introduced.

Chaotic systems used in the image encryption schemes can be divided into one-dimensional (1D) chaotic maps[13,14] and multi-dimensional (MD) chaotic systems.[15–17] The 1D chaotic systems have simple structures, and are easy to implement, but they have a limited range of chaotic behaviors and small key space, thus encryption methods using them have a

*Project supported by the National Natural Science Foundation of China (Grant Nos. 61203094 and 61305042), the Natural Science Foundation of the United States (Grant Nos. CNS-1253424 and ECCS-1202225), the Science and Technology Foundation of Henan Province, China (Grant No. 152102210048), the Foundation and Frontier Project of Henan Province, China (Grant No. 162300410196), the Natural Science Foundation of Educational Committee of Henan Province, China (Grant No. 14A413015), and the Research Foundation of Henan University, China (Grant No. xxjc20140006).

†Corresponding author. E-mail: chaixiuli@henu.edu.cn

© 2016 Chinese Physical Society and IOP Publishing Ltd

relatively low security level. The MD chaotic systems have complex structures and multiple parameters, therefore the encryption algorithms based on them have a large key space and higher security level. For instance, Zhou *et al.*[13] introduced a simple and effective chaotic system by using a combination of two 1D chaotic maps (seed maps) and it has larger chaotic ranges than its seed maps, and they also proposed a novel image encryption algorithm based on the chaotic system. Using two 1D logistic maps and a tent map, Wang *et al.*[14] put forward a new block image encryption scheme based on the dynamic random growth technique. Zhang and Wang[15] presented non-adjacent coupled map lattices with more outstanding features than the logistic map and coupled map lattices, and proposed a new bit-level image encryption algorithm based on it. Recently, Wu *et al.*[16] proposed a new lossless encryption algorithm for color images based on a six-dimensional (6D) hyperchaotic system and the two-dimensional (2D) discrete wavelet transform (DWT). Tong *et al.*[17] gave a joint color image encryption and compression scheme, and the algorithm employed the discrete cosine transformation dictionary to sparsely represent the color image and used an image encryption method based on a hyperchaotic system to achieve image compression and encryption simultaneously. In 1971, Chua proposed the concept of a memristor for the first time,[18] and the HP Laboratory declared the physical realization of the memristor in 2008.[19] Because of a unique switch mechanism, natural memory function, continuous input and output characteristics and the nanoscale size, the memristor has received a great deal of attention and shown wide potential applications in nonvolatile memory, artificial neural networks, intelligent information processing and others.[20–22] Many nonlinear memristive oscillators have been introduced, and recently a simple memristor-based chaotic circuit was studied by Muthuswamy and Chua.[23] The dynamical behaviors of memristive chaotic systems are closely dependent on the initial states and system parameters, all these features make the memristive chaotic systems, especially hyperchaotic systems more desirable for image encryption. In this paper, a 4D memristive hyperchaotic system is employed to generate pseudorandom sequences and the performance of the encryption algorithm will be improved by its good features.

Since Leiser *et al.*[24] introduced the possibility of applying the DNA binary strands to encryption, some image encryption algorithms based on DNA computation have been proposed due to the features of massive parallelism, huge storage and ultra-low power consumption.[25–28] A novel image encryption algorithm based on the DNA sequence and multi-chaotic maps was presented in Ref. [25], and Zhang *et al.*[26] analyzed its weakness and broke the method. Recently, Wang *et al.*[27] proposed a novel image encryption scheme by using DNA sequence operations and a CML spatiotemporal chaos system. Huang and Ye[28] introduced an image encryption algorithm based on hyperchaos and the DNA sequence, a 4D hyperchaotic system was employed to produce the pseudorandom sequence for diffusion, and a circular permutation was used for the plain image. These schemes convert the plain image and chaotic sequences into DNA matrices and encrypt the original image based on the DNA complementary rule, encoding and decoding rules, and addition, subtraction and XOR rules. Usually, these operations are a little complicated and time-consuming. Besides, their encryption algorithms are based on grayscale images. As for color images, the correlation between components R, G, and B is high, however, some encryption algorithms use the same methods to encrypt components of the color images by neglecting the correlations between components R, G, and B which make encryption algorithms more vulnerable to attack. Considering the larger data capacity and higher correlation among pixels, the encryption of color images needs better statistic and diffusion properties of encryption algorithms than that of grey images.

According to the above analyses, in this paper, a novel color image encryption scheme based on genetic recombination and a 4D memristive hyperchaotic system is proposed. The contributions in this paper are as follows. First, instead of complex traditional DNA encoding and decoding rules, the essential principle of genetic recombination is used to permutate the 4 compound bit planes of the plain image and key streams, and compound bit planes come from the recombination of the 24 bit planes of components R, G, and B, which can confuse some elements between 4 bit planes of the original image, make the key streams more random and upgrade the encryption effect. Second, key streams are generated from the 4D memristive hyperchaotic system, its initial values are produced by the SHA 256 hash function of the color plain image, therefore our algorithm has high sensitivity to the plain image and large key space, and it can effectively resist against both chosen plain image attack and known plain image attack. Third, in the process of encryption, permutation and diffusion are employed as a stage, and this can save a great deal of time.

The rest of the paper is organized as follows. In Section 2, the 4D memristive hyperchaotic system and the basic rules of genetic recombination are presented. The proposed cryptosystem is described in detail in Section 3. Experimental results are provided in Section 4. In Section 5, the performance and security analysis of the encryption algorithm are presented. In the last section, some conclusions are drawn from the present studies.

2. Preliminary work

2.1. 4D memristive hyperchaotic system

By adding a memristor and a crossproduct item into a 3D autonomous quadratic system, a 4D memristive hyperchaotic system can be obtained, which may generate a real four-wing

chaotic attractor;[29] it has richer and more complex dynamical behaviors than most of the known memristive systems,[30–32] and then it is more suitable for image encryption.

The system may be described by the following 4D ordinary differential equations:

$$\begin{cases} \dot{x} = ax + byz, \\ \dot{y} = cy + dxz - kyW(u), \\ \dot{z} = ez + fxy + gxu, \\ \dot{u} = -y, \end{cases} \tag{1}$$

where $a = 0.35$, $b = -10$, $c = -0.6$, $d = 0.3$, $e = -1.6$, $f = 2$, $W(u) = m + 3nu^2$, k, m, n, g are positive control parameters and x, y, z, u are the state variables. When $k = 0.2$, $m = 0.1$, $n = 0.01$, $g = 0.1$, the Lyapunov exponents computed by the Wolf method are 0.0905, 0.0147, -0.0001, and -1.9862, which means that the system is really a hyperchaotic system, and it can act as a four-wing hyperchaotic attractor shown in Fig. 1.

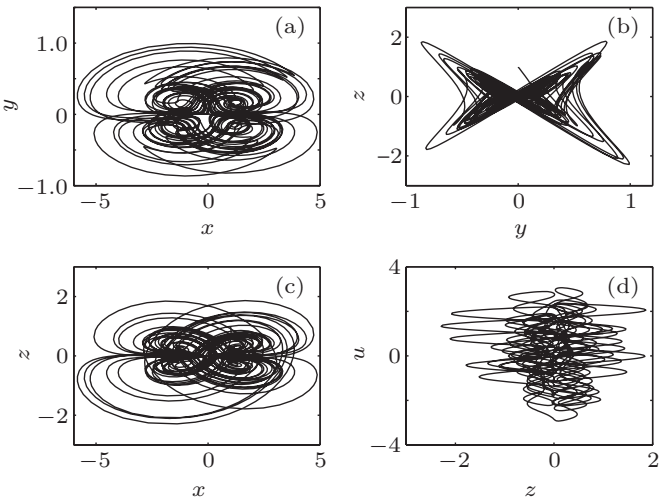


Fig. 1. Hyperchaotic attractors in (a) x–y plane, (b) y–z plane, (c) x–z plane, and (d) z–u plane.

Figure 2 illustrates the changes of the state variables x, y, z, u in 5000 iterations. These verify that the system has good hyperchaotic characteristics.

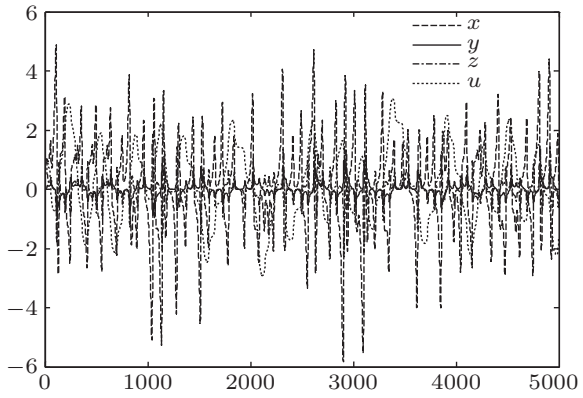


Fig. 2. Hyperchaotic tracks of the chaotic system.

2.2. Genetic recombination

Genetic recombination is the production of offspring with combinations of traits that differ from those found in either parent from the viewpoint of biological meaning. A strand of DNA or RNA is broken, joined to another part of a different DNA or RNA molecule, and then a new DNA or RNA molecule appears. In both meiotic and mitotic cells, genetic recombination between homologous chromosomes is a common mechanism used in DNA repair. Considering DNA strands, one of the DNA strands is broken by the natural or human aim, and then a new DNA strand is reconstructed with another one. Genetic recombination is made up of random recombination and exchange recombination. The random recombination refers to that the broken DNA strand will join some genes from other DNA strands of nonhomologous chromosomes, and the exchange recombination means that the broken DNA strands will exchange some genes with others from homologous chromosomes, and they can be illustrated in Fig. 3.

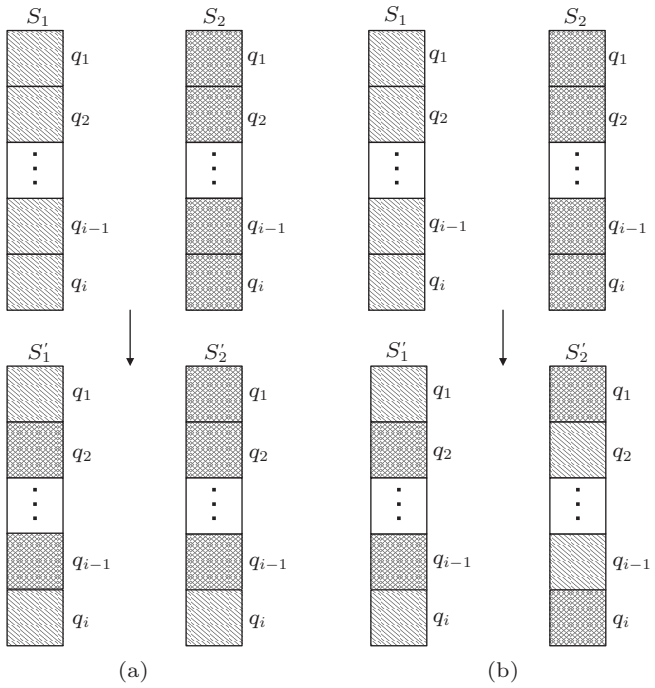


Fig. 3. Rules of random recombination and exchange recombination: (a) random recombination, and (b) exchange recombination.

In Fig. 3, the sequences S_1 and S_2 are divided into many parts by position indices q_i and then new sequences are reconstructed according to the rules of random recombination and exchange recombination. Genetic recombination of the sequences changes the position and values of elements of the sequences, and makes the sequences more random. If we apply genetic recombination to the plain image, we can change the positions and values of plain image pixels; if we apply it to the chaotic key streams, then more random key streams can be obtained. These can upgrade the performance of the image encryption algorithm.

3. Proposed cryptosystem

3.1. Generation of the initial values of the hyperchaotic system

Attacking analysis results of some encryption algorithms show that the methods cannot resist the chosen and known plain image attacks when they are not sensitive to changes in the plain images or secret key. In order to solve the problem, the proposed cryptosystem utilizes a 256-bit external secret key K, which is produced by the SHA 256 hash of the plain image. Even if there is only one bit difference between two plain images, their hash values will be completely different. The 256-bit secret key is divided into 8 bit blocks (k_i), thus K can be expressed as follows:

$$K = k_1, k_2, \ldots, k_{32}. \tag{2}$$

The 256-bit hash value serves as the one-time keys and is used to produce the initial values of the hyperchaotic system, and they may be derived as follows:

$$\begin{aligned} x_0 &= ((k_1 \oplus k_2 \oplus k_3 \oplus k_4 \oplus k_5) \\ &\quad + (k_6 \oplus k_7 \oplus k_8)) \bmod 2^8/255, \end{aligned} \tag{3}$$

$$\begin{aligned} y_0 &= ((k_9 \oplus k_{10} \oplus k_{11} \oplus k_{12} \oplus k_{13}) \\ &\quad + (k_{14} \oplus k_{15} \oplus k_{16})) \bmod 2^8/255, \end{aligned} \tag{4}$$

$$\begin{aligned} z_0 &= ((k_{17} \oplus k_{18} \oplus k_{19}) + (k_{20} \oplus k_{21}) \\ &\quad + (k_{22} \oplus k_{23} \oplus k_{24})) \bmod 2^8/255, \end{aligned} \tag{5}$$

$$\begin{aligned} u_0 &= ((k_{25} \oplus k_{26} \oplus k_{27}) + (k_{28} \oplus k_{29}) \\ &\quad + (k_{30} \oplus k_{31} \oplus k_{32})) \bmod 2^8/255. \end{aligned} \tag{6}$$

Here, x_0, y_0, z_0, and u_0 are the initial values of the hyperchaotic system, the mod is the modular operator and $x \oplus y$ denotes the XOR operation. Obviously, equations (2)–(6) show that the initial values of the hyperchaotic system are highly sensitive to the change of the plain image and the proposed algorithm with total complexity of 2^{128} [33] can resist any brute-force attack.

3.2. Preprocessing the color plain image

Assume that the color plain image is denoted as P, whose size is $M \times N$. For enhancing the performance of the color image encryption method, the plain image may be processed and the steps are shown as follows.

First, decompose the color image P into three components R, G, and B and their sizes are all $M \times N$. Next, decimal-to-binary conversion is applied to each pixel of components R, G, and B, 24 bit planes are obtained. Then, 6 bit planes are combined together, 4 compound bit planes appear, that is to say, the 1st, and 2nd planes of the R component, the 3rd, and 4th planes of the G component and the 5th, and 6th planes of the B component constitute compound bit plane 1; the 3rd, and 4th planes of the R component, the 5th, and 6th planes of the G component and the 7th, and 8th planes of the B component form compound bit plane 2; the 5th, and 6th planes of the R component, the 7th, and 8th planes of the G component and the 1st, and 2nd planes of the B component constitute compound bit plane 3; compound bit plane 4 is comprised of the 7th, and 8th planes of the R component, the 1st, and 2nd planes of the G component and the 3rd, and 4th planes of the B component. Then 4 images (CP_1, CP_2, CP_3, and CP_4) are obtained after the compound bit planes have been recombined. Finally, arrange the 4 images into corresponding vectors SP_1, SP_2, SP_3, and SP_4 each with a size of $1 \times MN$.

The preprocessing of the color plain image has two advantages. On the one hand, the bit planes of components R, G, and B are split and then recombined, 4 compound bit planes comprise two bit planes of components R, G, and B, and thus the following encryptions of vectors SP_1, SP_2, SP_3, and SP_4 can effectively remove the correlation among components R, G, and B and enhance the security level of the proposed cryptosystem. On the other hand, as is well known, the bits at different bit planes have different weights, in the process of the recombination of compound bit planes, the lower bit planes and the higher bit planes are exchanged, the position of the bit plane is changed and the weight of each bit is changed accordingly; this can downgrade the statistical information about the image and upgrade the diffusion performance.

3.3. Encryption process

The flow chart of our proposed encryption algorithm is illustrated in Fig. 4, where the detailed encryption steps are given.

Step 1 Calculate the initial values x_0, y_0, z_0, and u_0 of the memristive hyperchaotic system through Eqs. (2)–(6).

Step 2 Employ the preprocessing of the color plain image $P(M \times N)$, and obtain vectors SP_1, SP_2, SP_3, and SP_4.

Step 3 Generate the 4 integers q_1, q_2, q_3, and q_4 with vectors SP_1, SP_2, SP_3, SP_4 according to the following equation:

$$\begin{aligned} q_i &= \left\lfloor (\text{average}(SP_i) - \lfloor \text{average}(SP_i) \rfloor) \times 10^{14} \right\rfloor \\ &\quad \bmod (MN-1) + 1, \quad i = 1,2,3,4, \end{aligned} \tag{7}$$

where average(SP_i) denotes the average value of vector SP_i.

Step 4 Perform genetic recombination to vectors SP_1, SP_2, SP_3, SP_4 with integers q_1, q_2 through Eqs. (8)–(11) and obtain the recombined vectors SP_1', SP_2', SP_3', and SP_4' as follows:

$$\begin{aligned} SP_1' &= [SP_1(1), SP_1(q_1-1)] \cup [SP_4(q_1), SP_4(q_2)] \\ &\quad \cup [SP_2(q_2+1), SP_2(MN)], \end{aligned} \tag{8}$$

$$\begin{aligned} SP_2' &= [SP_2(1), SP_2(q_1-1)] \cup [SP_1(q_1), SP_1(q_2)] \\ &\quad \cup [SP_3(q_2+1), SP_3(MN)], \end{aligned} \tag{9}$$

$$\begin{aligned} SP_3' &= [SP_4(1), SP_4(q_1-1)] \cup [SP_3(q_1), SP_3(q_2)] \\ &\quad \cup [SP_1(q_2+1), SP_1(MN)], \end{aligned} \tag{10}$$

$$SP_4' = [SP_3(1), SP_3(q_1-1)] \cup [SP_2(q_1), SP_2(q_2)] \cup [SP_4(q_2+1), SP_4(MN)]. \quad (11)$$

Here, $SP_i(j)$ are the j-th elements of vectors SP_i ($i = 1,2,3,4$) and $SP_i(j) \in [0, 2^6]$ ($j = 1,2,\ldots,MN$); $[x(i), x(j)]$ denotes the vector of x from the i-th element to j-th element, $x \cup y$ denotes the union of vector x and vector y.

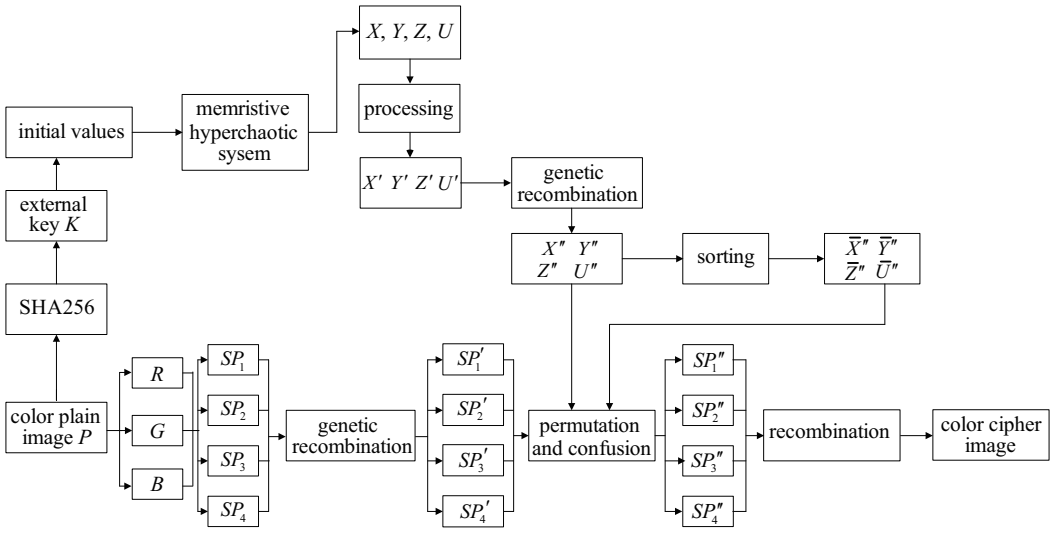


Fig. 4. Flow chart of the proposed color image encryption algorithm.

Step 5 Iterate the memristive hyperchaotic system $M \times N + l$ times by using the initial values x_0, y_0, z_0, and u_0 produced by Step 1, state variables x_i, y_i, z_i, and u_i can be obtained, and then the four integer sequences X, Y, Z, and U are generated

$$\begin{cases} X = \{x_1, x_2, \ldots, x_{MN}\}, \\ Y = \{y_1, y_2, \ldots, y_{MN}\}, \\ Z = \{z_1, z_2, \ldots, z_{MN}\}, \\ U = \{u_1, u_2, \ldots, u_{MN}\}, \end{cases} \quad (12)$$

where

$$\begin{cases} x_i = \lfloor (\mathrm{abs}(x_i) - \lfloor x_i \rfloor) \times 10^{14} \rfloor \bmod 2^{10}, \\ y_i = \lfloor (\mathrm{abs}(y_i) - \lfloor y_i \rfloor) \times 10^{14} \rfloor \bmod 2^{10}, \\ z_i = \lfloor (\mathrm{abs}(z_i) - \lfloor z_i \rfloor) \times 10^{14} \rfloor \bmod 2^{10}, \\ u_i = \lfloor (\mathrm{abs}(u_i) - \lfloor u_i \rfloor) \times 10^{14} \rfloor \bmod 2^{10}. \end{cases} \quad (13)$$

$\lfloor x \rfloor$ returns to the nearest integer for x, in order to avoid transit effects, we must discard the values obtained by the former l ($l \geq 500$) times.

Step 6 Convert the elements of sequences X, Y, Z, and U into new sequences X', Y', Z', and U' through Eqs. (14) and (15).

$$\begin{cases} X' = \{x_1', x_2', \ldots, x_{MN}'\}, \\ Y' = \{y_1', y_2', \ldots, y_{MN}'\}, \\ Z' = \{z_1', z_2', \ldots, z_{MN}'\}, \\ U' = \{u_1', u_2', \ldots, u_{MN}'\}, \end{cases} \quad (14)$$

where

$$\begin{cases} x_i' = (x_i + y_i) \bmod 2^6, \\ y_i' = (y_i + z_i) \bmod 2^6, \\ z_i' = (z_i + u_i) \bmod 2^6, \\ u_i' = (x_i + y_i + u_i) \bmod 2^6. \end{cases} \quad (15)$$

Step 7 Implement genetic recombination of sequences X', Y', Z', and U' with q_3 and q_4 through Eqs. (16)–(19) and obtain the recombined sequences X'', Y'', Z'', and U'' as follows:

$$X'' = [X'(1), X'(q_3-1)] \cup [Y'(q_3), Y'(q_4)] \cup [Z'(q_4+1), Z'(MN)], \quad (16)$$

$$Y'' = [U'(1), U'(q_3-1)] \cup [X'(q_3), X'(q_4)] \cup [Y'(q_4+1), Y'(MN)], \quad (17)$$

$$Z'' = [Z'(1), Z'(q_3-1)] \cup [Z'(q_3), Z'(q_4)] \cup [U'(q_4+1), U'(MN)], \quad (18)$$

$$U'' = [Y'(1), Y'(q_3-1)] \cup [U'(q_3), U'(q_4)] \cup [X'(q_4+1), X'(MN)]. \quad (19)$$

Step 8 Sort the sequences X'', Y'', Z'', and U'', obtain the sorted sequences SX'', SY'', SZ'', and SU''. Compare the sorted sequences with the original ones, the mapping sequences $\bar{X}''$, $\bar{Y}''$, $\bar{Z}''$, $\bar{U}''$ can be obtained. Taking SX'' for example, after comparison, if the 1st element of sequence SX'' is located at $\bar{x''}_1$ position in sequence X'', the first element of the mapping sequence $\bar{X}''$ is $\bar{x''}_1$; if the 2nd element of sequence SX'' is located at $\bar{x''}_2$ position in sequence X'', the second element of the mapping sequence $\bar{X}''$ is $\bar{x''}_2$, and so on, the mapping sequence $\bar{X}''$ can be obtained. Thus, we can obtain other mapping sequences $\bar{Y}''$, $\bar{Z}''$, $\bar{U}''$ according to the same method. They can be shown in Eq. (20).

$$\begin{cases} \bar{X}'' = \{\bar{x''}_1, \bar{x''}_2, \ldots, \bar{x''}_{MN}, \}, \\ \bar{Y}'' = \{\bar{y''}_1, \bar{y''}_2, \ldots, \bar{y''}_{MN}\}, \\ \bar{Z}'' = \{\bar{z''}_1, \bar{z''}_2, \ldots, \bar{z''}_{MN}\}, \\ \bar{U}'' = \{\bar{u''}_1, \bar{u''}_2, \ldots, \bar{u''}_{MN}\}. \end{cases} \quad (20)$$

Step 9 Apply permutation and diffusion to vectors SP'_1, SP'_2, SP'_3, and SP'_4 and obtain vectors SP''_1, SP''_2, SP''_3, and SP''_4. Specifically, modify and rearrange the elements of the vectors SP'_1, SP'_2, SP'_3, and SP'_4 by using recombined sequences X'', Y'', Z'', U'' and mapping sequences $\bar{X}''$, $\bar{Y}''$, $\bar{Z}''$, $\bar{U}''$ as follows:

$$SP''_1(\bar{x''}_i) = SP'_1(i) \oplus x''_i, \tag{21}$$

$$SP''_2(\bar{y''}_i) = SP'_2(i) \oplus y''_i, \tag{22}$$

$$SP''_3(\bar{z''}_i) = SP'_3(i) \oplus z''_i, \tag{23}$$

$$SP''_4(\bar{u''}_i) = SP'_4(i) \oplus u''_i, \tag{24}$$

where $SP'_1(i)$, $SP'_2(i)$, $SP'_3(i)$, and $SP'_4(i)$ denote the i-th elements of vectors SP'_1, SP'_2, SP'_3, and SP'_4 respectively; x''_i, y''_i, z''_i, u''_i are the i-th elements of sequences X'', Y'', Z'', and $\bar{U}''$, respectively, $i = 1, 2, \ldots, MN$.

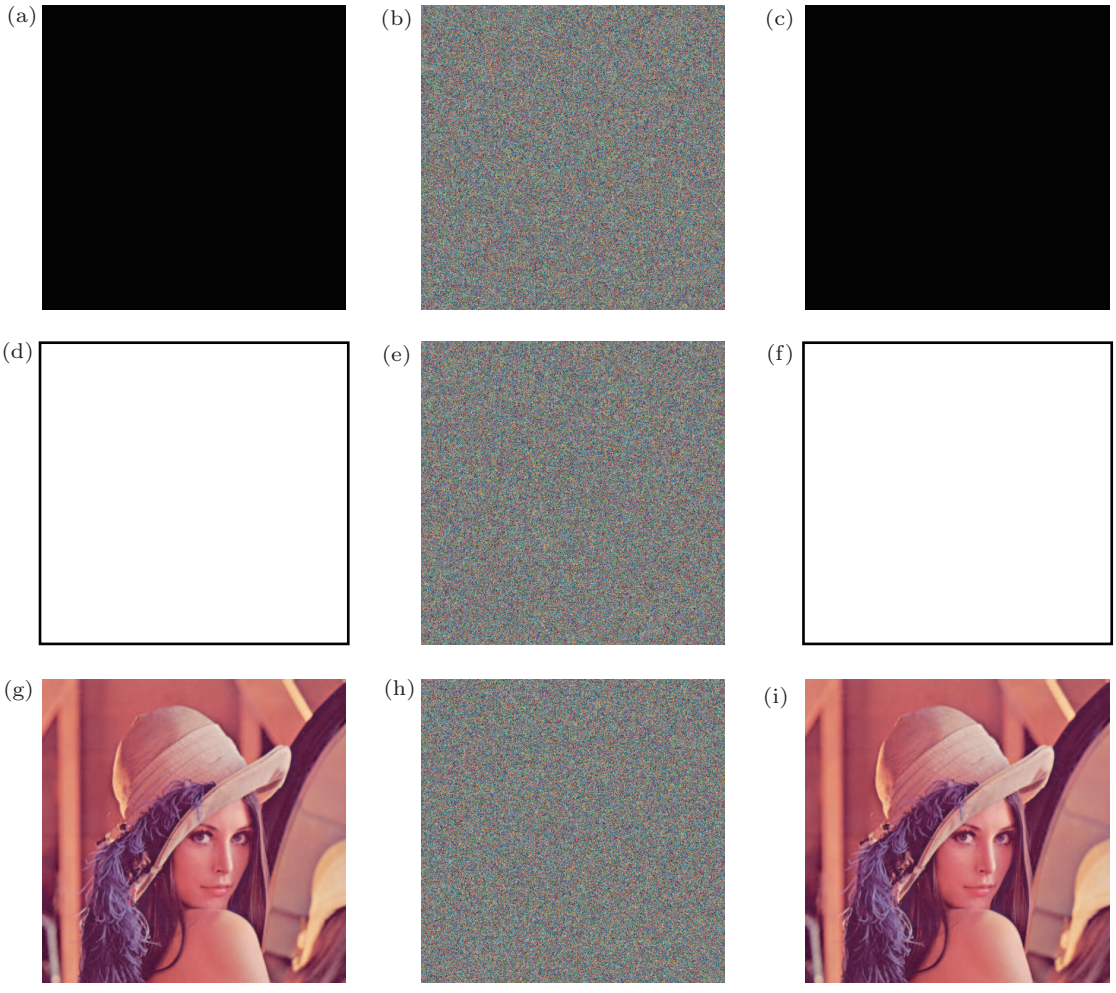
Step 10 Resize the vectors SP''_1, SP''_2, SP''_3, and SP''_4 to 4 image matrices, respectively, and their sizes are all $M \times N$. Next, decompose the image matrix into 6 bit planes, then 24 it planes will appear. Every 8 bit planes are recombined, 3 cipher images are obtained, and they are the cipher images of components R, G, and B. Finally, a color cipher image can be obtained through recombining the cipher images of components R, G, and B.

As described in the encryption process, the proposed scheme has three merits. Firstly, the initial values of the 4D memristive hyperchaotic system are produced by the SHA 256 hash value of the plain image, thus our method highly depends on the original image. Secondly, the essential principle of genetic recombination is employed to confuse the 4 compound bit planes of the plain image and key streams, and this not only realizes the combined confusion of the components R, G, and B, but also obtains the more random key streams. Thirdly, permutation and diffusion are manipulated in a stage, and this can improve the encryption speed.

4. Experimental results

In this section, simulation results are presented to testify the proposed algorithm. The experimental environment is as follows: CPU: Intel Core i5-2450 M, 2.5 GHz; Memory: 4 GB; Operating system: Windows 7; Coding tool: Matlab 2014a. "all black" (512×512), "all white" (512×512), "Lena" (512×512), "Girl" (256×256), and "Pens" (512×480) are used as plain images, the cipher images, and recovered images are shown in Fig. 5, respectively.

Figure 5 shows that the cipher images are something like noise and we cannot obtain any information from them. Besides, the proposed algorithm does not have the limitation to image size, and it can be applied to all kinds of images.



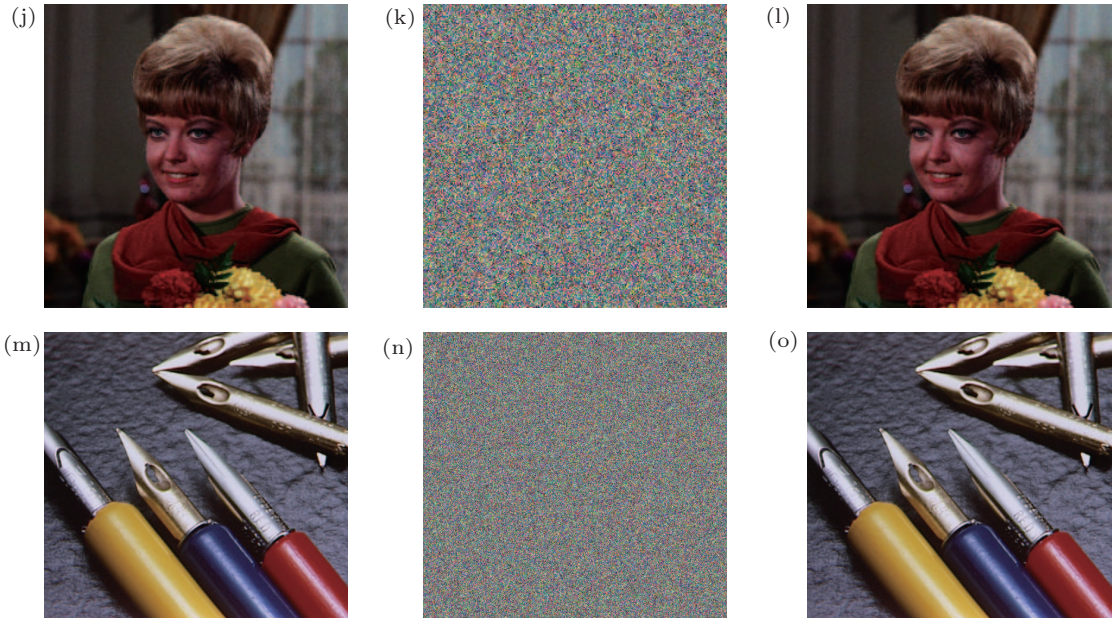


Fig. 5. (color online) Simulation results: (a) plain image of all black, (b) cipher image, (c) recovered image, (d) plain image of all white, (e) cipher image, (f) recovered image, (g) plain image of Lena, (h) cipher image, (i) recovered image, (j) plain image of girl, (k) cipher image, (l) recovered image, (m) plain image of pens, (n) cipher image, (o) recovered image.

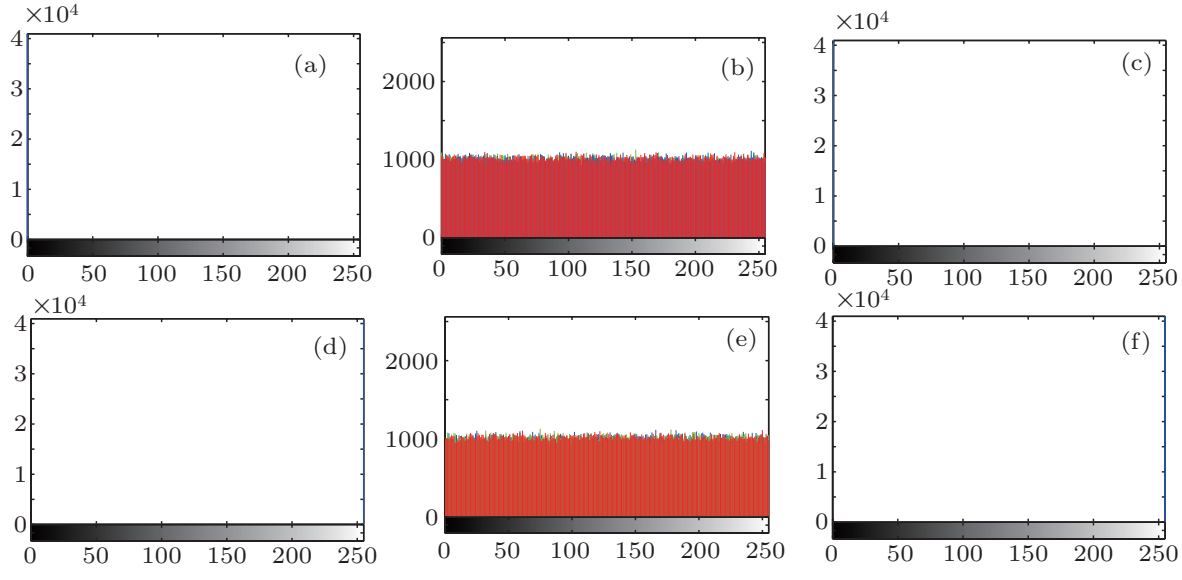
5. Performance and security analysis

5.1. Key space analysis

As a good image encryption algorithm, its key space should be large enough to make any brute force attack ineffective. However, a too large secret key may reduce the encryption speed. From the cryptography point of view, the size of the key space should not be smaller than 2^{100} to provide a high security level.[34] In the proposed algorithm, the secret keys include: (i) the 256-bit external key K produced by the SHA 256 hash function; (ii) parameter l which is related to the iteration process of the hyperchaotic system. Since the security of SHA 256 with complexity of the best attack is 2^{128},[33] the size of the key space is large enough to resist all kinds of brute-force attacks.

5.2. Histogram analysis

A histogram is used to illustrate the distribution of pixel values of an image. The ideal histogram of a cipher image is uniform, and the histograms of the plain image and the corresponding decrypted image are the same. Figure 6 illustrates the histograms of R, G, and B components for the plain image "all black" (512×512), "all white" (512×512), "Lena" (512×512), "girl" (256×256) , "pens" (512×480) and their cipher images and decrypted images. In Figs. 6(a), 6(d), 6(g), 6(j), and 6(m), the histograms of R, G, and B components are steep and not flat enough, while it is clear that in Fig. 6(b), 6(e), 6(h), 6(k), and 6(n), the histograms of R, G, and B components are flat, and the histograms of R, G, and B components for the plain images are the same as those for the decrypted images shown in Figs. 6(c), 6(f), 6(i), 6(l), and 6(o), respectively.



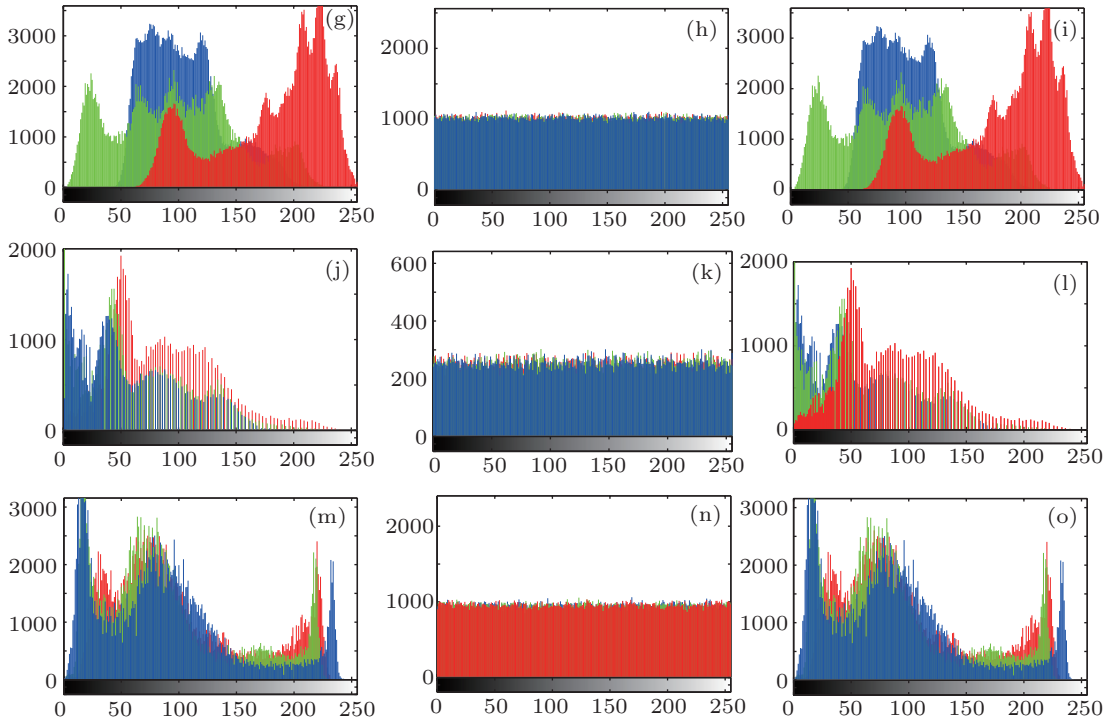


Fig. 6. (color online) Histograms of the plain images, cipher images, and decrypted images. (a) Histogram of all black. (b) Histogram of cipher image. (c) Histogram of decrypted image. (d) Histogram of all white. (e) Histogram of cipher image. (f) Histogram of decrypted image. (g) Histogram of Lena. (h) Histogram of cipher image. (i) Histogram of decrypted image. (j) Histogram of girl. (k) Histogram of cipher image. (l) Histogram of decrypted image. (m) Histogram of pens. (n) Histogram of cipher image. (o) Histogram of decrypted image.

Furthermore, variances of histograms are employed to quantitatively evaluate the uniformities of the ciphered images. The lower the variance, the higher the uniformity of the cipher image is. The variance of histogram can be computed from the following equation:[6]

$$\mathrm{var}(Z) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(z_i - z_j)^2. \tag{25}$$

Here, Z denotes the vector of the histogram value and $Z = \{z_1, z_2, \ldots, z_{256}\}$, z_i and z_j are the numbers of pixels whose gray values are equal to i and j, respectively. For Lena (512×512), the variances of the histograms of the plain image and cipher image are illustrated in Table 1. From the results, we can see that the variances of the cipher images are greatly reduced compared with the corresponding values of the plain images, those of our algorithm are less than variances of the cipher image of Lena in Ref. [5], and the proposed algorithm is efficient.

Table 1. Variances of the histograms of the plain image, and cipher image of Lena (512×512).

Image	Plain image of Lena (512×512)			Cipher image of Lena (512×512)			Ref. [5]
	R	*G*	*B*	*R*	*G*	*B*	
Variance	1.017×10^6	4.557×10^5	1.377×10^6	1.070×10^3	955.320	995.828	5554.82

5.3. Correlation analysis

Correlation reflects a linear relationship between two adjacent pixels for an image. The correlation of the plain images is generally high and it may leak information. A good encryption algorithm should remove the correlation between adjacent pixels as much as possible. The less the correlation, the more effectively the method performs. The correlation coefficients $r_{x,y}$ of two adjacent pixels can be calculated according to the following formula:

$$r_{x,y} = \frac{E((x-E(x))(y-E(y)))}{\sqrt{D(x)D(y)}}, \tag{26}$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{27}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2, \tag{28}$$

where $E(x)$ and $D(x)$ are the expectation and variance of variable x, respectively.

To analyze the correlation between the plain image and cipher image, we randomly choose 5000 pairs of two adjacent pixels from the plain images and their corresponding cipher images to calculate the correlation coefficients in horizontal, vertical and diagonal directions. The results are listed in Table 2. Table 2 shows that the correlation coefficients of the

plain images are all greater than 0.9, which means that there exist strong relationships between adjacent pixels in different directions; whereas, the correlation coefficients of the corresponding cipher images are all less than 0.02, which indicates the negligible correlation.

Table 2. Correlation coefficients between the plain image and cipher image.

Scan direction			Horizontal	Vertical	Diagonal
all black (512×512)	Cipher image	*R*	0.0007	0.0105	–0.0240
		G	–0.0002	0.0206	0.0108
		B	0.0091	0.0026	0.0208
all white (512×512)	Cipher image	*R*	0.0062	0.0206	0.0025
		G	–0.0060	–0.0070	0.0017
		B	0.0020	0.0053	–0.0142
Lena (512×512)	Plain image	*R*	0.9820	0.9721	0.9571
		G	0.9708	0.9538	0.9299
		B	0.9186	0.8826	0.8548
	Cipher image	*R*	0.0128	–0.0031	–0.0033
		G	–0.0170	0.0160	–0.0093
		B	0.0001	–0.0190	–0.0130
Girl (256×256)	Plain image	*R*	0.9632	0.9728	0.9464
		G	0.9664	0.9747	0.9439
		B	0.9532	0.9524	0.9346
	Cipher image	*R*	0.0092	0.0012	0.0162
		G	0.0196	–0.0199	–0.0063
		B	–0.0027	–0.0080	–0.0014
Pens (512×480)	Plain image	*R*	0.9768	0.9834	0.9759
		G	0.9729	0.9863	0.9727
		B	0.9681	0.9794	0.9654
	Cipher image	*R*	0.0091	0.0019	–0.0151
		G	0.0163	–0.0163	–0.0065
		B	0.0072	0.0147	–0.0077

Table 3. The same position correlations between *R*, *G*, and *B* components.

Scan direction		*R–G*	*R–B*	*G–B*
Plain image	Lena (512×512)	0.8498	0.5579	0.8610
	Girl (256×256)	0.7647	0.6910	0.9266
	Pens (512×480)	0.9901	0.9601	0.9743
Cipher image	All black (512×512)	0.0080	0.0212	0.0240
	All white (512×512)	0.0216	–0.0048	–0.0139
	Lena (512×512)	0.0022	0.0022	0.0212
	Girl (256×256)	–0.0000	–0.0031	–0.0166
	Pens (512×480)	–0.0072	0.0129	0.0073

There is high correlation between adjacent pixels of *R*, *G*, *B* components for the color images. Thus, a good encryption method should break the correlations between adjacent pixels of *R*, *G*, *B* components. Tables 3 and 4 show the results of the same position correlations and adjacent position correlations between *R*, *G*, *B* components of plain images and cipher images, respectively. The results demonstrate that the correlations of adjacent pixels in the cipher images are greatly reduced. The comparison of correlation coefficients using the Lena image presented in Table 5 gives the comparison results of correlation coefficients between our algorithm and other methods, and the test image is Lena. The results show that the confusion and diffusion performance of the proposed method are much better than those of other schemes.

Besides, as $D(x) = 0$ for all black and white images, their correlation coefficients of the plain images may not be calculated, and those of corresponding cipher images are tested and shown in Tables 2–5.

Table 4. Adjacent position correlations between *R*, *G*, and *B* components.

Scan direction		*R–G*	*R–B*	*G–B*
Plain image	Lena (512×512)	0.8260	0.5337	0.7929
	Girl (256×256)	0.7530	0.6658	0.8898
	Pens (512×480)	0.9793	0.9444	0.9574
Cipher image	All black (512×512)	–0.0132	–0.0028	–0.0030
	All white (512×512)	0.0051	0.0279	0.0085
	Lena (512×512)	0.0044	0.0067	0.0081
	Girl (256×256)	0.0343	–0.0057	–0.0017
	Pens (512×480)	–0.0061	–0.0106	–0.0074

Table 5. Comparison analyses of correlation coefficients using the Lena image.

Scan direction	Horizontal	Vertical	Diagonal
Plain image	0.9820	0.9741	0.9621
Proposed algorithm	0.0083	–0.0046	0.0050
Ref. [35]	0.0075	0.0129	0.0049
Ref. [36]	0.0117	0.0026	0.0011
Ref. [37]	0.0043	0.0054	0.0072
Ref. [38]	0.0108	0.0181	0.0061
Ref. [39]	0.0032	0.0042	0.0018

5.4. Key sensitivity

An effective cryptosystem should have high key sensitivity, and it can be observed from two aspects. On the one hand, in the encryption process, when the secret keys change slightly, a completely different cipher image is produced. On the other hand, in the decryption process, the plain image cannot be successfully decrypted if even a tiny difference exists between the encryption and decryption keys.

5.4.1. Key sensitivity of encryption process

Firstly, we test the key sensitivity of the encryption process. The generation of a secret key is mostly based on the plain image, so if there is a slight change between the two images, the hash values will be different, and therefore the initial values of the hyperchaotic system are different. The encryption results for image "Lena" (512×512) with only one pixel difference are shown in Fig. 7 and Table 6. The modified images are shown in Figs. 7(a), 7(d), and 7(g), the encryption results are shown in Figs. 7(b), 7(e), and 7(h), and the differences are illustrated in Figs. 7(c), 7(f), and 7(i). Table 6 shows that when the plain image of Lena has a pixel changed, a completely different key appears and the corresponding cipher image has about 99.6% of the pixels modified compared with Fig. 5(h).

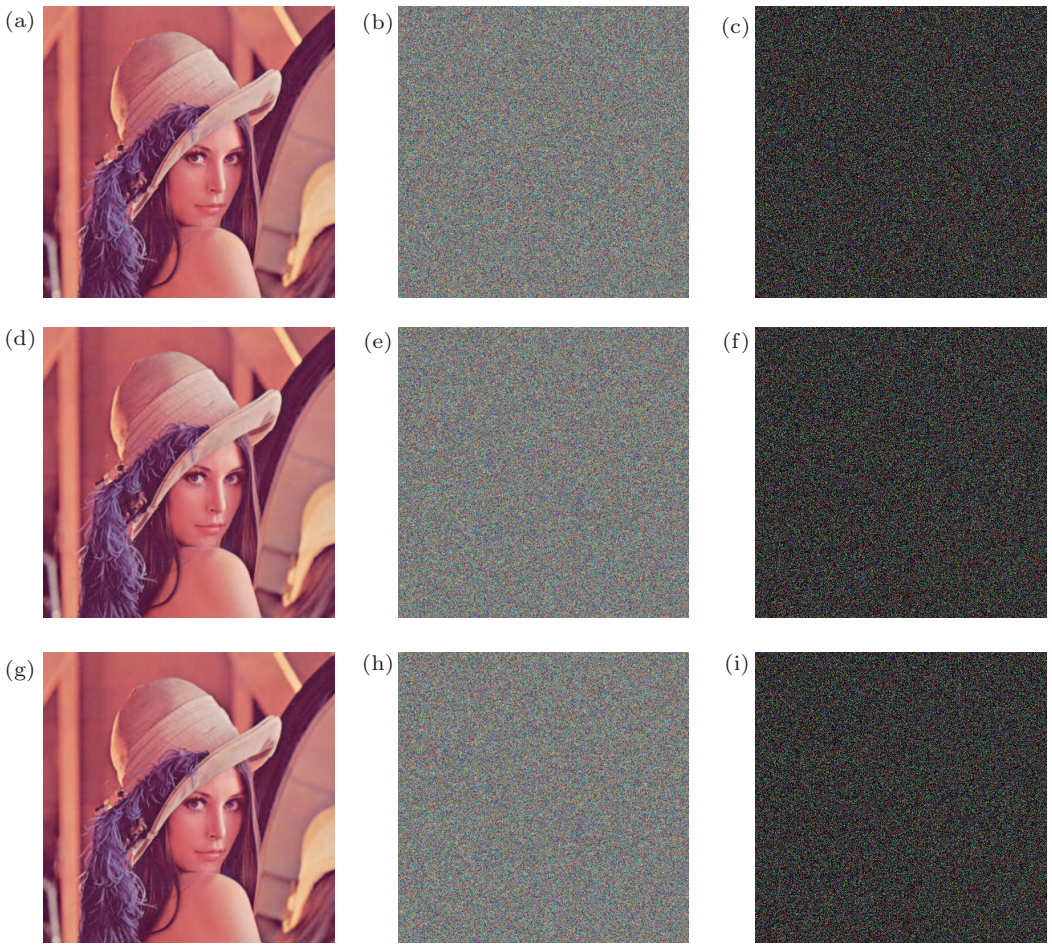


Fig. 7. (color online) Key sensitivity results of encryption process. (a) Lena with $P(1,1,1) = 200$. (b) Corresponding encryption result. (c) Difference between Fig. 7(b) and Fig. 5(h). (d) Lena with $P(20,35,2) = 120$. (e) Corresponding encryption result. (f) Difference between Fig. 7(e) and Fig. 5(h). (g) Lena with $P(120,225,3) = 255$. (h) Corresponding encryption result. (i) Difference between Fig. 7(h) and Fig. 5(h).

Table 6. Differences between cipher images generated with the plain image one pixel changed.

Pixel change	Key		Comparison with Fig. 5(h)
$P(1,1,1) = 200$	3923ca5bf9acbd9238f43dbfaac18a45	92f9290358206ea15dd3413a83fb986d	0.9960
$P(20,35,2) = 120$	d414a5c847b02454113912841b27f9e	b469d5234c3fc6e6d0ddc66eeef2eb56f	0.9961
$P(120,225,3) = 255$	d414a5c847b02454113912841b27f9e	b469d5234c3fc6e6d0ddc66eeef2eb56f	0.9960

5.4.2. Key sensitivity of decryption process

In addition, we analyze the key sensitivity of the decryption process. Figure 8 shows the decryption results for the cipher image in Fig. 5(h) of Lena with slightly modified keys K_1, K_2, and K_3. Differences between decipher images generated with slightly different keys and the plain image are 99.61%, 99.60%, and 99.59% shown in Table 7. The original image cannot be obtained successfully even with a slight change of the secret key. Thus our image encryption scheme is highly sensitive to the secure key.

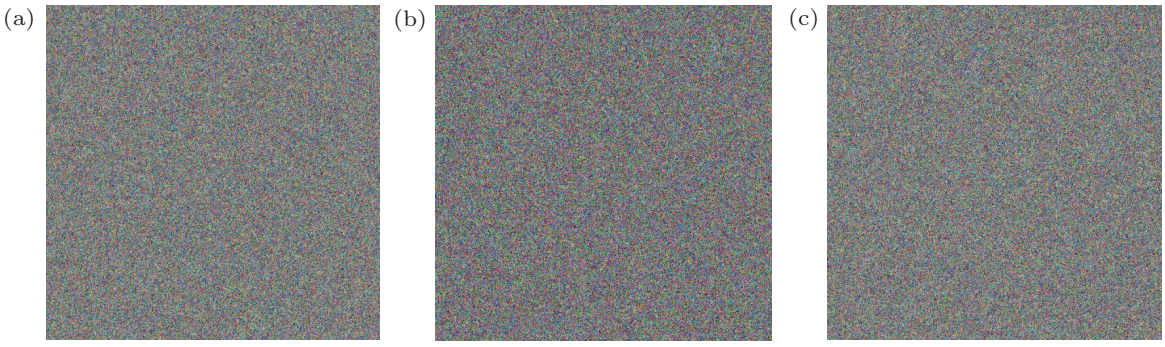


Fig. 8. (color online) Decryption results for the Lena image with different keys: (a) decrypted image with K_1, (b) decrypted image with K_2, (c) decrypted image with K_3.

Table 7. Differences between the decipher image generated with slightly different keys and the plain image.

Decryption image	Decryption key	Comparison with Fig. 5(g)
Fig. 5(i)	K_0= c056da23302d2fb0d946e7ffa11e0d946 18224193ff6e2f78ef8097bb8a3569b	0
Fig. 7(a)	K_1 =d056da23302d2fb0d946e7ffa11e0d946 18224193ff6e2f78ef8097bb8a3569b	0.9961
Fig. 7(b)	K_2 =c056da23302d2fb0d946e7ffa11e0d947 18224193ff6e2f78ef8097bb8a3569b	0.9960
Fig. 7(c)	K_3 =c056da23302d2fb0d946e7ffa11e0d946 18224193ff6e2f78ef8097bb8a3569a	0.9959

5.5. Attack analyses

5.5.1. Differential attack

A good image encryption algorithm must resist the differential attack, which requires that a slight change in the plain image should bring a completely different cipher image. The number of pixels changing rate (NPCR) and unified average changing intensity (UACI) are put forward to test the resisting differential attack performance of the encryption method. The closer to 1 the NPCR, the more effective the algorithm is. The closer to 0.3333 the UACI, the more effectively the cryptosystem resists differential attack. The values of NPCR and UACI can be calculated from the following formula:

$$\mathrm{NPCR} = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\%, \tag{29}$$

$$\mathrm{UACI} = \frac{1}{W \times H}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\%, \tag{30}$$

where $D(i,j)$ is defined as

$$D(i,j) = \begin{cases} 1, & C_1(i,j) \neq C_2(i,j), \\ 0, & \text{otherwise}, \end{cases} \tag{31}$$

with C_1 and C_2 denoting the cipher images before and after one pixel of the plain image has been changed, and W and H are the width and height of the image respectively.

To test the proposed algorithm, we encrypt the plain images which differ by only one pixel by using the same secret keys, and the results are shown in Table 8. Changing one pixel in the plain images may cause good performance by having NPCR > 0.99 and UACI > 0.33. Table 9 illustrates the comparison results between the average values of $\mathrm{NPCR}_{R,G,B}$ and $\mathrm{UACI}_{R,G,B}$ obtained by our algorithm and those by other methods. The results show that our encryption method is better than some encryption methods in Refs. [36] and [37] and the same as the methods in Refs. [35] and [39], which indicates that our scheme can effectively resist against differential attack.

Table 8. NPCR and UACI calculated by two cipher images whose plain images differ by one pixel and using the same secret keys.

Plain image	NPCR			Average $\mathrm{NPCR}_{R,G,B}$	UACI			Average $\mathrm{UACI}_{R,G,B}$
	R	G	B		R	G	B	
All black (512×512)	0.9961	0.9959	0.9950	0.9960	0.3325	0.3331	0.3324	0.3327
All white (512×512)	0.9959	0.9962	0.9961	0.9961	0.3318	0.3326	0.3324	0.3323
Lena (512×512)	0.9960	0.9960	0.9960	0.9960	0.3325	0.3328	0.3331	0.3328
Girl (256×256)	0.9960	0.9959	0.9958	0.9959	0.3344	0.3347	0.3339	0.3343
Pens (512×480)	0.9961	0.9960	0.9959	0.9960	0.3345	0.3337	0.3339	0.3340

Table 9. Comparison between average $\mathrm{NPCR}_{R,G,B}$ and $\mathrm{UACI}_{R,G,B}$ values.

Item	Ours	Ref. [36]	Ref. [37]	Ref. [35]	Ref. [39]
Average $\mathrm{NPCR}_{R,G,B}$	0.9960	0.00038	0.00004	0.9960	0.9963
Average $\mathrm{UACI}_{R,G,B}$	0.3332	0.00004	0.00332	0.3342	0.3342

5.5.2. Entropy attack

Information entropy is the most important measure of randomness. Information entropy $H(m)$ of an information source m can be calculated from the following formula:

$$H(m) = \sum_{i=0}^{2^n-1} p(m_i) \log \frac{1}{p(m_i)}, \tag{32}$$

where $p(m_i)$ denotes the probability of symbol m_i. For a random image with 256 grey levels, we may obtain a theoretical value 8 from Eq. (32). The closer to 8 the information entropy of the cipher image, the less possibly the cryptosystem divulges information. Table 10 indicates the entropies of R, G, B components of the plain image and cipher image, obtained by using our algorithms, and the comparison results with other methods are shown in Table 11. From Tables 10 and 11, it is known that the entropies calculated by our algorithm are closer to the ideal entropy value 8, so the algorithm proposed has a better property of information entropy.

Table 10. Entropy test for R, G, B components of color plain images and cipher images.

Plain image	R	G	B	Cipher image	R	G	B
All black (512×512)	0	0	0	All black (512×512)	7.9994	7.9993	7.9993
All white (512×512)	0	0	0	All white (512×512)	7.9994	7.9993	7.9992
Lena (512×512)	7.2531	7.5940	6.9684	Lena (512×512)	7.9993	7.9993	7.9993
Girl (256×256)	6.4200	6.4457	6.3807	Girl (256×256)	7.9969	7.9969	7.9968
Pens (512×480)	7.5336	7.4399	7.4893	Pens (512×480)	7.9992	7.9993	7.9992

Table 11. Information entropies of the Lena cipher image, obtained from our method and other methods.

	Ours	Ref. [40]	Ref. [41]	Ref. [42]	Ref. [43]
R component	7.9993	7.9993	7.9899	7.997	7.9899
G component	7.9993	7.9992	7.9894	7.997	7.9997
B component	7.9993	7.9993	7.9896	7.997	7.9979

5.5.3. Resisting known-plaintext and chosen-plaintext attack

Usually, the adversary employs all-black or all-white images as special plain images to attack the encryption algorithms, for the special images can make the permutation process invalid, and they can make the encryption method known and unsafe. Figures 5(b) and 5(e) show the cipher images of the all-black image and all-white image, and they are something like noise. The adversary cannot obtain useful information to break the encryption scheme. Therefore, our encryption algorithm can resist against the known-plaintext attacks and chosen-plaintext attacks effectively.

5.6. Color band analysis

Color bands of the color plain image and cipher image are employed to illustrate the distributions of their R, G, B components. Figure 9(a) shows the color band of the color Lena image and that of the color cipher image is illustrated in Fig. 9(b). Figure 9 shows that the color band of the plain image is highly correlated, and that of the cipher image is almost uniformly distributed, which means that our encryption algorithm has good performances of confusion and diffusion.

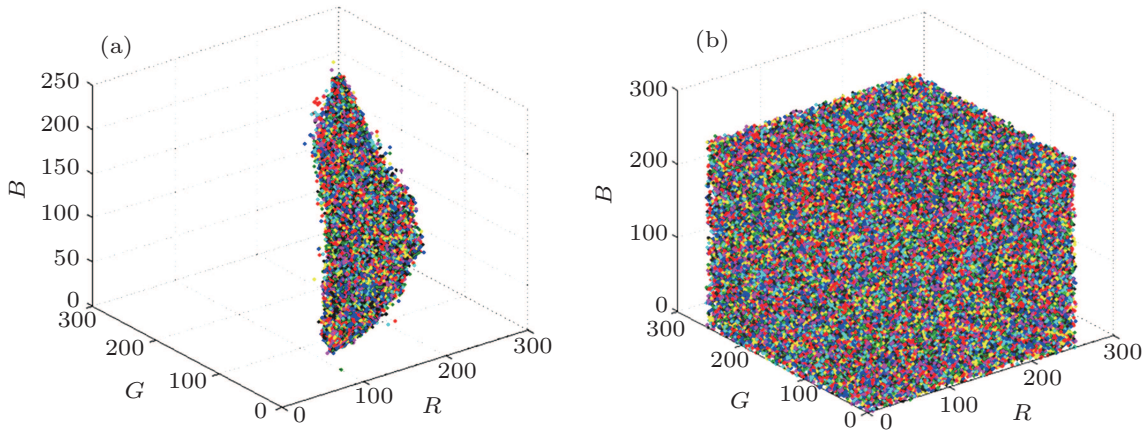


Fig. 9. (color online) Color band analyses of (a) color Lena image and (b) color cipher image.

5.7. Encryption speed

Regardless of the security considerations, encryption speed is also important, especially in real-time internet applications. In the proposed algorithm, some methods are employed to save the running time. Firstly, a 4D memristive hyperchaotic system is used and we obtain the chaotic sequences for the confusion and diffusion through iterating the hyperchaotic system once. Secondly, the essential principle of genetic recombination is used to confuse the plain images and chaotic sequences, and the traditional complex DNA encoding and decoding rules are removed. Besides, when we modify the pixel values of the plain images, the positions of the pixels are also changed, which means the confusion and diffusion are combined as a union, thus saving much time also.

Table 12 shows the performance comparison results with other algorithms, the round number of image scanning, permutation and diffusion which are used as indicators. As shown in Table 12, we can find that 1 round image-scanning and 1 round diffusion are required to obtain satisfactory security levels of NPCR $>$ 0.996 and UACI $>$ 0.333. In Ref. [46], 18-round image scanning had to be performed for a good performance, and it was not effective for real-time encryption. Thus, our schemes have less encryption rounds and higher running efficiency than algorithms in Refs. [44]–[50], and it can be used in real-time image encryption conditions.

Table 12. Comparison among performances obtained by different methods to achieve a satisfactory security level.

Algorithms	NPCR	UACI	The round number of		
			Image-scanning	Permutation	Diffusion
Ours	>0.996	>0.333	1	0	1
Ref. [44]	>0.996	>0.333	6	3	3
Ref. [45]	>0.996	>0.333	1	2	2
Ref. [46]	>0.996	>0.333	18	18	6
Ref. [47]	>0.996	>0.333	4	4	2
Ref. [48]	>0.996	>0.333	6	3	3
Ref. [49]	>0.996	>0.333	2	2	2
Ref. [50]	>0.996	>0.333	2	0	2

6. Conclusions

In this paper, a novel color image encryption algorithm based on genetic recombination and the memristive hyperchaotic system is introduced. A 4D memristive hyperchaotic system is employed to generate the key streams and the SHA 256 hash value of the color plain images is utilized to compute its initial values, so the proposed method is highly sensitive to the plain image and it can resist against the known and chosen plain image attack effectively. Furthermore, some encryption algorithms use the same methods to encrypt three components of the color images, by neglecting the correlations between components R, G, and B which make encryption algorithms more vulnerable to attacks; in order to remove it, the 24 bit planes of components R, G, and B are obtained and recombined to 4 compound bit planes for encryption. Then, we use the genetic recombination to confuse the compound bit planes and key streams. This can exchange some elements between 4 bit planes, but also make the key streams more random. Besides, confusion and diffusion are united together, and this can make our algorithm run faster.

The simulation experiments, including key space analysis, histogram analysis, correlation analysis, key sensitivity, differential attack, entropy attack, resisting known-plaintext and chosen-plaintext attack, color band analysis, and encryption speed, indicate that the proposed algorithm has good statistical and diffusion properties and can resist many kinds of attacks and perform well in speed. In addition, the proposed scheme is reliable to be applied to image encryption and transmission. Due to the use of a 4D memristive hyperchaotic system, the proposed scheme increases some computational cost compared with a low-dimensional chaos-based method. In the future, we plan to explore the algorithm for encrypting many colour images simultaneously, and implementing it in hardware, so that our algorithm can be used in actual communication systems.

References

[1] Zhang X Y, Zhang G J, Li Xuan, Ren Y Z and Wu J H 2016 *Chin. Phys. B* **25** 054201
[2] Xiao D, Cai H K and Zheng H Y 2015 *Chin. Phys. B* **24** 060505
[3] Wang X Y and Wang Q 2014 *Chin. Phys. B* **23** 030503
[4] Fridrich J 1998 *Int. J. Bifur. Chaos* **8** 1259
[5] Zhu Z L, Zhang W, Wong K W and Yu H 2011 *Inform. Sci.* **181** 1171
[6] Zhang Y Q and Wang X Y 2014 *Inform. Sci.* **273** 329
[7] Liu H J and Wang X Y 2011 *Opt. Commun.* **284** 3895
[8] Xu L, Li Z, Li J and Hua W 2016 *Opt. Laser Eng.* **78** 17
[9] Fatih Ozkaynak and Sirma Yavuz 2014 *Nonlinear Dyn.* **78** 1311
[10] Eslami Z and Bakhshandeh A 2013 *Opt. Commun.* **286** 51
[11] Akhavan A, Samsudin A and Akhshani A 2015 *Opt. Commun.* **350** 77
[12] Rabei Bechikh, Houcemeddine Hermassi, Ahmed A. Abd EI-Latif, Rhouma Rhouma and Safya Belghith 2015 *Signal Proces. Image Commun.* **39** 151
[13] Zhou Y C, Bao L and Philip Chen C L 2014 *Signal Proces.* **97** 172
[14] Wang X Y, Liu L T and Zhang Y Q 2015 *Opt. Laser Eng.* **66** 10
[15] Zhang Y Q and Wang X Y 2015 *Appl. Soft Comput.* **26** 10
[16] Wu X J, Wang D W, Kurths Jurgen and Kan H B 2016 *Inform. Sci.* **349–350** 137
[17] Tong X J, Zhang M, Wang Z and Ma J 2016 *Nonlinear Dyn.*
[18] Chua L O 1971 *IEEE Trans. Circuit Theory* **18** 507
[19] Tour J M and He T 2008 *Nature* **453** 42
[20] Duan S K, Zhang Y, Hu X, Wang L D and Li C D 2014 *Neural Comput. Appl.* **25** 1437
[21] Wang L D, Drakakis E, Duan S K, He P F and Liao X F 2012 *Int. J. Bifur. Chaos* **22** 1250205
[22] Adhikari S P, Yang C, Kim H and Chua L O 2012 *IEEE Trans. Neural Netw. Learn. Syst.* **23** 1426
[23] Muthuswamy B and Chua L O 2010 *Int. J. Bifur. Chaos* **20** 1567
[24] Leier A, Richter C and Banzhaf W 2000 *Biosystems* **57** 13
[25] Xue X L, Zhang Q and Wei X P 2010 *J. Comput. Theor. Nanosci.* **7** 397
[26] Zhang Y S, Wen W Y and Su M T 2014 *Optik* **125** 1562
[27] Wang X Y, Zhang Y Q and Bao X M 2014 *Opt. Laser Eng.* **73** 53
[28] Huang X L and Ye G D 2014 *Multimed Tools Appl.* **72** 57
[29] Ma J, Chen Z Q, Wang Z L and Zhang Q 2015 *Nonlinear Dyn.* **81** 1275
[30] Teng L, Iu Herbert H C, Wang X Y and Wang X K 2014 *Nonlinear Dyn.* **77** 231
[31] Cafagna D and Grassi G 2012 *Nonlinear Dyn.* **70** 1185
[32] Grassi G, Severance F L, Mashev E D and Bazuin B J and Miller D A 2008 *Int. J. Bifur. Chaos* **18** 2089
[33] European Network of Excellence in Cryptology II, ⟨http://www.ecrypt.eu.org⟩, 2010
[34] Norouzi B, Seyedzadeh S M, Mirzakuchaki S and Mosavi M R 2013 *Multimed Tools Appl.* **74** 781
[35] Mazloom S and Eftekhari-Moghadam A 2009 *Chaos, Solitons and Fractals* **42** 1745
[36] Liu S, Sun J and Xu Z 2009 *J. Comput.* **4** 1091
[37] Akhshani A, Akhavan A, Lim S C and Hassan Z 2012 *Commun. Nonlinear Sci. Numer. Simul.* **17** 4653
[38] Wang X Y, Teng L and Qin X 2012 *Signal Proces.* **92** 1101
[39] El-Latif A A A, Li L, Wang N, Han Q and Niu X 2013 *Signal Proces.* **93** 2986
[40] Wang Y, Zhang X, Zheng Z M and Qiu W J 2015 *Nonlinear Dyn.* **81** 151
[41] Wu X J, Bai C and Kan H 2014 *Commun. Nonlinear Sci. Numer. Simul.* **19** 1884
[42] Zhang Y S and Xiao D 2014 *Int. J. Electron. Commun.* **68** 361
[43] Faraoun K M 2014 *Opt. Laser Technol.* **64** 145
[44] Zhang Y S and Xiao D 2014 *Commun. Nonlinear Sci. Numer. Simul.* **19** 74
[45] Wang Y, Wong K W, Liao X F, Xiang T and Chen G R 2009 *Chaos, Solitons and Fractals* **41** 1773
[46] Lian S, Sun J and Wang Z 2005 *Chaos, Solitons and Fractals* **26** 117
[47] Wong K W, Kwok B S H and Law W S 2008 *Phys. Lett. A* **372** 2645
[48] Xiao D and Liao X W 2009 *Chaos, Solitons and Fractals* **40** 2191
[49] Wang Y, Wong K W, Liao X and Chen G 2011 *Appl. Soft Comput.* **11** 514
[50] Benyamin N, Mohammad S S and Sattar M 2014 *Multimed. Syst.* **20** 45