2012

Image Encryption Using the Recursive Attributes of the eXclusive-OR Filter on Cellular Automata

Chatzichristofis, Savvas A.

Springer-Verlag

Image Encryption Using the Recursive Attributes of the eXclusive-OR Filter on Cellular Automata

Savvas A. Chatzichristofis[1], Oge Marques[2], Mathias Lux[3], and Yiannis Boutalis[1]

[1] Department of Electrical and Computer Engineering
Democritus University of Thrace, Xanthi, Greece
[2] Department of Computer & Electrical Engineering and Computer Science (CEECS)
Florida Atlantic University (FAU), Boca Raton, Florida, USA
[3] Institute for Information Technology
Klagenfurt University, Klagenfurt, Austria
{schatzic,ybout}@ee.duth.gr, omarques@fau.edu,
mlux@itec.uni-klu.ac.at

Abstract. A novel visual multimedia content encryption method based on cellular automata (CA) is presented in this paper. The proposed algorithm is based on an attribute of the eXclusive-OR (XOR) logic gate, according to which, its application to a square-sized CA has the ability to reconstruct the original content of a CA after a preset number of iterations. The resulted encrypted image is a lossless representation of the original/plaintext image, i.e. there is no loss of either resolution or contrast. Experimental results indicate that the encrypted image does not contain any statistical information able to reveal the original image.

1 Introduction

Nowadays it is often imperative to safeguard the transmission of visual multimedia information. The conventional encryption methods, e.g. the 3DES and AES, are incapable of encrypting data with patterns, like images. In order to overcome this problem, plaintext block chaining or plaintext feedback and output feedback techniques are usually applied. However, those methods only apply to problems of considerably small complexity. In recent literature though, there are numerous image encryption algorithms which may be classified according to their ability to lossless reconstruct the encrypted image or lead to loss of information after the decryption. Additionally, these methods can be classified based on the approach used to achieve the encryption, which may be divided into four categories: *SCAN*-based techniques (e.g. SCAN-based permutation of pixels) [1] [3] [15], *Chaos*-based techniques [19] [7] [11], *Structure*-based information [5], and finally algorithms where the encryption technique combine elements from the other techniques or use elements borrowed from different scientific disciplines. However, each approach is characterized by advantages and disadvantages in terms of security level and speed [3].

Several image encryption methods based on cellular automata (CA) are already reported in literature. In [14] a family of basic functions, generated from the evolving states of CA, is used to encrypt multimedia information. In [20], CA were used to produce the bit stream of the key in a Vernam cipher cryptography. An image encryption

G.C. Sirakoulis and S. Bandini (Eds.): ACRI 2012, LNCS 7495, pp. 340–350, 2012.

method based on the permutation of the pixels of an image and the replacement of the pixel values is proposed in [3], where the permutation is done by SCAN patterns. The pixel values are replaced using a progressive CA substitution with a sequence of CA data that is generated from the CA evolution rules. Other CA image security methods are reported in [8] and [10].

A new lossless visual multimedia content encryption method based on CA is proposed in this paper. The proposed algorithm is based on an attribute of the eXclusive-OR (XOR) logic gate, according to which, its application to a square-sized CA has the ability to reconstruct the the original content of the CA after a preset number of repetitions. This attribute is presented in details in Section 2.

The proposed method is a symmetric-key based one, which means that it requires the same key both for encrypting and decrypting an image. Detailed description of the key construction as well as its use in encrypting and decrypting images, is presented in Section 3. Section 4 describes the encryption characteristics and evaluates several security issues regarding the application of the proposed method. Finally, the conclusions are drawn in Section 5.

2 The Recursive Attribute of XOR Filters

A single channel image (i.e. a grayscale or a binary image) can be considered as a CA which is comprised of a linear two-dimensional (2D) table of identical cells. Every pixel of the image corresponds to a cell of the CA and each one of these cells can be found in k different states. The local state of the cell x, y during time step t is given by the formula:

$$s_t^{x,y} \in \sum = \{0, 1, \ldots, k-1\} \tag{1}$$

where $k = 256$ for grayscale images and $k = 2$ for binary ones.

The total state of the CA during time t, designated as s_t, is the configuration of the whole table:

$$s_t = (s_t^{0,0}, s_t^{0,1}, s_t^{0,2}, \ldots, s_t^{W-1,H-1}) \in \sum^{W \times H} \tag{2}$$

where W, H designate the width and height of the 2D CA respectively.

As already mentioned, this paper is based on an attribute of the XOR filter when applied to a 2D CA, and consequently to a single channel image. According to this attribute, the CA returns in its initial state after a predefined number of applications of the filter. The implementation of the XOR gate in the CA is quite widespread in the literature, as well as the recursiveness of the resulting outcomes [4] [6] [2] [9]. The filtering process is described below:

At every time step all the cells of the CA recalculate their state concurrently according to the following rule: The values of the cells in the Moore neighborhood with radius $r = 1$, participate in the logical operation (XOR) and their result is placed in the central cell of the neighborhood.

$$\begin{aligned} S_1^{x,y} &= S_0^{x-1,y-1} \oplus S_0^{x,y-1} \oplus S_0^{x+1,y-1} \\ &\oplus S_0^{x-1,y} \oplus S_0^{x,y} \oplus S_0^{x+1,y} \\ &\oplus S_0^{x-1,y+1} \oplus S_0^{x,y+1} \oplus S_0^{x+1,y+1} \end{aligned}$$

After applying the filter t times, the CA returns to its original state, i.e. $S_t^{x,y} = S_0^{x,y}$.

$$\begin{aligned} &S_1^{x,y} = S_0^{x-1,y-1} \oplus S_0^{x,y-1} \oplus S_0^{x+1,y-1} \oplus S_0^{x-1,y} \oplus \\ &S_0^{x,y} \oplus S_0^{x+1,y} \oplus S_0^{x-1,y+1} \oplus S_0^{x,y+1} \oplus S_0^{x+1,y+1} \\ \\ &S_2^{x,y} = S_1^{x-1,y-1} \oplus S_1^{x,y-1} \oplus S_1^{x+1,y-1} \oplus S_1^{x-1,y} \oplus \\ &S_1^{x,y} \oplus S_1^{x+1,y} \oplus S_1^{x-1,y+1} \oplus S_1^{x,y+1} \oplus S_1^{x+1,y+1} \\ &\ldots \\ &S_t^{x,y} = S_0^{x,y} \end{aligned}$$

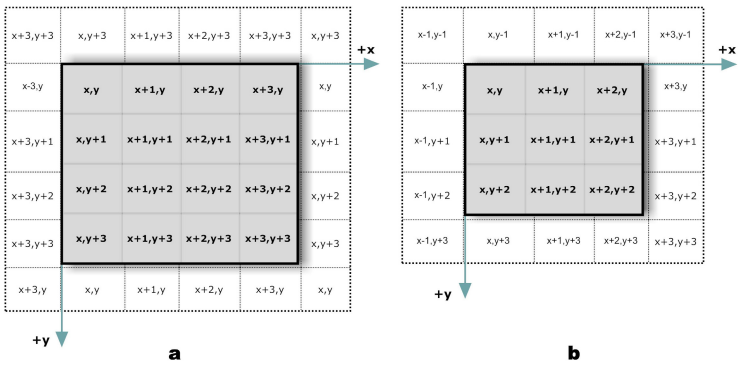


Fig. 1. Boundaries of the CA. (a) Periodical. (b) Pseudo-Cells

The reconstruction periodicity is based strictly on the size of the CA and on the **boundary conditions**. The proposed encryption method is based on the reconstruction periodicity which occurs under two different conditions:

Condition 1. *The application of an XOR filter on a CA of $N \times N$ dimensions and* periodical *boundaries has the ability to lossless reconstruct the CA after $t = N/2$ repetitions when $N = 2^p$ and $p \in Z$.*

Periodical boundaries of a CA are displayed in Figure 1(a). This may be visualized as taping the left and right edges of the rectangle to form a tube, then taping the top and bottom edges of the tube to form a torus (donut shape). The interested reader may find more information regarding this condition in [2] [9].

Condition 2. *The application of an XOR filter on a CA of $N' \times N'$ dimensions and* pseudo-cells *boundaries has the ability to lossless reconstruct the CA after $t = 2^p \times 2$ iterations when $N' = 2^p - 1$ and $p \in Z$. In other words, the application of an XOR filter on a CA can reconstruct the CA after $t = (N' + 1) \times 2$ steps.*

Pseudo-Cells boundaries of a CA are displayed in Figure 1(b). In this case, the data forming the boundaries belong to hypothesized neighboring cells. In other words, the $N' \times N'$ CA is regarded as a part of another $N' + 1 \times N' + 1$ CA, to the boundary conditions of which the filter is not applied. An application of a CLF (COORDINATE LOGIC FILTERS)-XOR filter to a 2D CA of 3×3 ($p = 2$) dimensions is presented in Figure 2.

125	145	187	156	158
255	214	57	125	147
187	196	164	165	166
125	126	128	127	129
187	189	80	82	84

a

125	145	187	156	158
255	156	225	201	147
187	72	214	14	166
125	14	251	143	129
187	189	80	82	84

b

125	145	187	156	158
255	240	146	124	147
187	47	94	48	166
125	251	85	221	129
187	189	80	82	84

c

125	145	187	156	158
255	0	233	12	147
187	132	44	188	166
125	79	141	151	129
187	189	80	82	84

d

125	145	187	156	158
255	82	71	249	147
187	186	164	219	166
125	250	254	251	129
187	189	80	82	84

e

125	145	187	156	158
255	24	159	77	147
187	54	214	112	166
125	138	133	11	129
187	189	80	82	84

f

125	145	187	156	158
255	116	236	248	147
187	81	94	78	166
125	127	43	89	129
187	189	80	82	84

g

125	145	187	156	158
255	132	151	136	147
187	250	44	194	166
125	203	243	19	129
187	189	80	82	84

h

125	145	187	156	158
255	214	57	125	147
187	196	164	165	166
125	126	128	127	129
187	189	80	82	84

i

Fig. 2. Application of CLF-XOR to a 2D CA size of 3×3

It is crucial to emphasize that under both conditions, during intermediate time steps, the CA content and consequently the single channel image is greatly distorted in an almost unpredictable manner.

In case of color images, each channel is processed individually. More specifically, each channel is considered as a 2D CA. Hence, for a typical RGB image, 3 CAs of identical dimensions are generated. The cell values are integer numbers in $[0, 255]$, which are subsequently converted into a binary form. The XOR filter is applied on every cell of every CA, which is actually an application of the filter to every pixel of each channel of the image. Essentially, this filter executes the XOR operation to the binary form values of the 9 cells that belong to the Moore neighborhood on which the filter is applied. The application of the filter to the nine(9) 8-bit values results in an 8-bit number, and the operation outcome is placed in the central cell of the neighborhood. This procedure is repeated for every pixel/cell of every channel.

The exact same procedure should be followed when the filter is applied more than once, with new values being generated for all the CA cells after each iteration. These new values constitute the basis for the next iteration and so on. It is worth noting once more that during the interim time steps, the image is greatly distorted, making it impossible to be recognized.

3 The Proposed Image Encryption Scheme

Let the "to be encrypted" image (plaintext image) be of $N \times N$ dimensions, with $N = 2^p, \quad p \in \mathcal{Z}$. If this is not the case, the image should be modified by adding extra random generated pixels as surrounding frame, in order to get the desired dimensions. Initially, each channel of the plaintext image is regarded as an independent CA, to each of which the XOR filter is applied $K[i]$ times, with $i \in [0, 2]$. Each value of i (i.e. $i = 0$, $i = 1$, and $i = 2$) corresponds to a channel of the plaintext image (i.e. red, green, and blue channel, respectively). The $K[i]$ values are generated via the random number generator proposed in [13] and belongs to $K[i] \in [0, N/2]$.

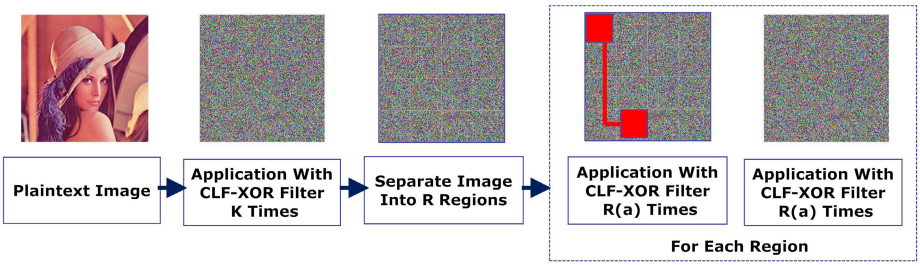


Fig. 3. Image Encryption Process

In the sequence, an integer pseudo-random number Q, where $Q \in [4, Y]$, $2^Y \leq N/2$, is generated via the same random number generator. These values define the number of sub-images that the $K[i]$ times filtered plaintext image will be divided to.

Every sub-image is of $N' \times N'$ dimensions, with: $N' = 2^Q - 1$.

Given that MOD $\left(\frac{N}{N'}\right) > 0$, necessarily, some sub-images will overlap. The manner via which the plaintext image is divided may be seen in Figure 4. Note that although $Q > 4$, so that the smallest sub-image possible being of dimensions 15×15, in the given example the values $N = 8$ and $Q = 2$ are selected for ease of understanding. Hence, the dimensions of each sub-image are $N' \times N' = 3 \times 3$. As may be seen in Figure 4, the top-right and bottom-left sub-images (designated as '3' and '5'), are borrowing pixels from the neighboring sub-images

The image is divided into R sub-images, each of which is regarded as an independent CA, with its boundaries being assumed as pseudo-cells. Actually, when applying the XOR filter to the sub-images' CA, the values of the cells of the neighboring CA participate to the boundary conditions. Consequently, in order to retrieve the content of each CA, no change should occur to the neighboring CA.

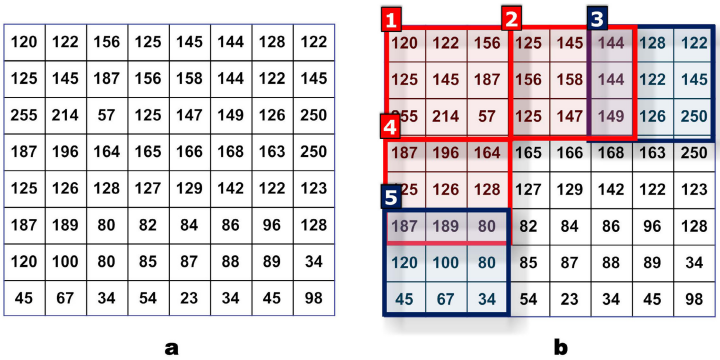
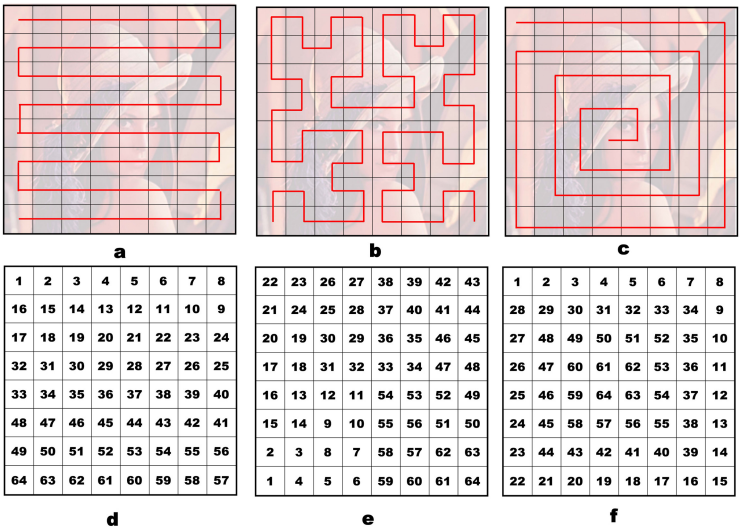


Fig. 4. Separating the plaintext Image into Sub Images

At this point, the random number generator generates R values in $[0, 2 \times (N' + 1)]$ which are stored in the array $RA[i], i \in [0, R-1]$, as well as R values in $[0, N/2]$ which are stored in array $RB[i], i \in [0, R-1]$.

Hence, in every sub-image i, the XOR filter will be applied $RA[i]$ times. A very important parameter of the proposed method is the ordering in which the sub-images will be filtered. Let for example a grayscale plaintext image C being divided into 4 sub-images $C[i], i \in [0, 3]$ of dimensions $N' \times N'$. Let also every sub-image to be filtered with the filter XOR $RA[i]$ times. Even if $RA[0] = RA[1] = RA[2] = RA[3]$, the result depends on the order in which the sub-images are filtered, because each filtering of every sub-image utilizes information of the current state of the neighboring sub-images. Furthermore, it is worth noting that in order to reconstruct every sub-image, it has to be filtered $(2 \times (N' - 1)) - RA[i]$ times, following the opposite order of the initial $RA[i]$ filter applications. For example, when an XOR filter is applied $RA[0]$ times to sub-image $C[0]$, then $RA[1]$ times to $C[1]$, then $RA[2]$ times to $C[2]$ and finally $RA[3]$ times to $C[3]$, in order to reconstruct the sub-images' content, the XOR filter must be applied $(2 \times (N' - 1)) - RA[3]$ times to $C[3]$, then $(2 \times (N' - 1)) - RA[2]$ times to $C[2]$, then $(2 \times (N' - 1)) - RA[1]$ times to $C[1]$, and finally $(2 \times (N' - 1)) - RA[0]$ times to $C[0]$.

The ordering in which the sub-images will be filtered is defined by a random integer generated by the pseudo-random number generator. This number defines which space ordering method [18] will be used. Some common examples of space ordering methods are (a) the raw order, (b) the row prime order, (c) the Mordon order, (d) the Peano Hilbert order, (e) the Cantor diagonal order, (f) the spiral order, (g) the Gray order, (h) the double gray order, (i) the U order, (j) the Z-Order, and so on [17]. Of course, any other ordering method could be utilized, given that it is known to both the sender and the receiver side. An illustration of (a) the Row Prime Order, (b) the Peano Hilbert Order, and (c) the Spiral Order, for 64 sub-images, are presented in Figure 5. Furthermore, Figures 5(d),(e), and (f), present the sub-images filtering ordering if the Raw Prime, the Peano Hilbert, or the Spiral ordering method is applied, respectively.



1	2	3	4	5	6	7	8
16	15	14	13	12	11	10	9
17	18	19	20	21	22	23	24
32	31	30	29	28	27	26	25
33	34	35	36	37	38	39	40
48	47	46	45	44	43	42	41
49	50	51	52	53	54	55	56
64	63	62	61	60	59	58	57

22	23	26	27	38	39	42	43
21	24	25	28	37	40	41	44
20	19	30	29	36	35	46	45
17	18	31	32	33	34	47	48
16	13	12	11	54	53	52	49
15	14	9	10	55	56	51	50
2	3	8	7	58	57	62	63
1	4	5	6	59	60	61	64

1	2	3	4	5	6	7	8
28	29	30	31	32	33	34	9
27	48	49	50	51	52	35	10
26	47	60	61	62	53	36	11
25	46	59	64	63	54	37	12
24	45	58	57	56	55	38	13
23	44	43	42	41	40	39	14
22	21	20	19	18	17	16	15

Fig. 5. Space Ordering Methods. (a) Row Prime Order, (b) Peano Hilbert Order and (c) Spiral Order

Finally, after the $RA[i]$ filter applications on the i sub-image, the whole image is filtered $RB[i]$ times. This way, the neighboring cells states are even more altered. The procedure is completed when all the sub-images are filtered.

The proposed method can be classified as a symmetric private key security method, meaning that the same key is required for both encryption and decryption; of course, both sender and receiver must know the key. The proposed key for encrypting a single channel image is divided into the following parts. The first part is an integer number $K, K \in [0, N/2]$ which describes the number of XOR filter applications to the plaintext image. The second part describes the size N' of every sub-image, whereas the third part is an integer F designating the utilized space ordering method.

The final part of the key includes R values $RA[i], i \in [0, R-1]$, describing the number of the XOR filter applications to every sub-image, as well as R values $RB[i], i \in [0, R-1]$ for the number of filter applications to the plaintext image, after every sub-image filtering. In case of color images, 3 such keys are required, one for every channel of the image. Note that to increase the encryption security, different values for every part of each key may be selected.

Given that the length of the key alone, may reveal crucial part of the information that an attacker would need to decrypt the image, the key may be modified as follows. Although the number of sub-images equals R, the key may include $\frac{N}{15} + 1$ values to the arrays RA and RB (the value 15 corresponds to the smallest sub-image size possible). From all the values in the arrays RA and RB, only the first R correspond to the actual times the XOR filter was applied to the sub-images and the plaintext image, whereas the rest of the values are random numbers generated. Thus, for same sized images, the key length is the same irrespectively of the size of the sub-images. The procedure of exchanging the key is not a feature of the proposed method and is omitted.

For decrypting the image, the receiver follows the exact opposite procedure. The first element retrieving from the key is the N' value so that the size of the sub-images is recognized, and subsequently identify which of the RA and RB values included in the key are useful. Then the XOR filter is applied $(N/2 - RB[i])$, $i \in [R, 0]$, times to the encrypted image, and then $(2 \times (N' + 1) - RA[i])$, $i \in [R, 0]$, times to the sub-image that is defined by the ordering method – which is known via the F value of the key. It is crucial to note that during decryption, the inverse ordering of the selected space ordering method is followed. Finally, it should be reminded that the proposed method reconstructs the plaintext image in a lossless manner.

4 Statistical Analysis

Initially, the resistance of the method against a brute force attack is evaluated. Brute force attack is a trial and error method in which every possible combination of characters against the encrypted data is tested in an attempt to retrieve the key. Suppose an adversary possesses a cipher image with $N \times N$ dimensions, with $N = 512$. To begin with, a random value $Q \in [4, 8]$, $2^8 \leq 256$ is chosen by the adversary, say $Q = 4$. Thus the adversary assumes that the image is divided into 4 sub-images of dimensions $512/(2^Q - 1) \times 512/(2^Q - 1)$. There are 1225 sub-images in total. Then, for every sub-image $16 \times 2 = 32$ combinations are tested. For the sake of simplicity, suppose that the plaintext image is NOT filtered by K times and that after filtering each sub-image, the entire image is NOT filter RB times. At the beginning, the adversary would apply the filter up to 32 times to every sub-image. In one of those trials the sub-image content would be reconstructed (given that Q is the correct one), and thus the 1/1225 of the image would have been revealed. If no sub-image would have been reconstructed, the adversary would choose a different Q. After revealing the first sub-image, the adversary has to unlock the 8 neighboring sub-images identifying the utilized space ordering method as well - with finite complexity. Then, following the inverse ordering, and after only up to 32 iterations per sub-image, the whole image is reconstructed.

By initially filtering the plaintext image K times, it is impossible for the adversary to realize that a sub-image is reconstructed, because its original state is the K times filtered state of the plaintext. Hence the only possible way to reconstruct the content is the following: select randomly a space ordering method and locate the last sub-image and its neighboring 8 sub-images. Adversary's goal would be to test all the possible RA combinations to this neighborhood, and then to each one of these combinations test the $N/2 = 256$ filtering iterations to the whole image. Once more, in this scenario it is supposed that after filtering a sub-image, the whole image is NOT RB times filtered. The complexity in this case is defined as: $256 \times 9^{32} = 8.8 \times 10^{32}$. Of course, the correct space ordering method should have been chosen at first place. Furthermore, note that there is no guaranty that a part of the plain text image is truly revealed even if the RA values of the neighborhood are identified, in case the rest of the sub-images are not reconstructed as well.

By applying the filter RA times to every sub-image, the complexity is intensively increased because it is impossible to isolate a neighborhood of sub-images, due to the fact that the population of the neighborhood changes. Furthermore, given that the whole

procedure should be followed for every channel of the image, it is strongly believed that a brute force attack would have no result at reconstructing the plaintext image.

The difference between the plaintext image and the encrypted one is analyzed in the following. To distinguish the difference, the function of peak signal to noise ratio (PSNR) is adopted. Although PSNR is commonly used as a measure of quality of reconstruction of non-lossless encryption methods, it is a very strong tool for describing image distortion as well. PSNR is usually expressed in logarithmic decibel scale. When using 8 bit images the greatest difference between images is calculated at 0 dB, whereas for matching images the value tends to infinity.

To assess the method, the experiment proposed in [2] is conducted for 3 artificially constructed images. Furthermore, the method is applied to 3 real-life images which are well known in the literature. Then, the PSNR between the plaintext and the encrypted images is calculated. The results per channel are presented in Table 1.

Table 1. PSNR Values (in dB) for the Encrypted and Random Generated Images

	Lena	Lady	Fruits
Random Image 1 (R/G/B)	6.87/6.98/6.54	7.08/6.41/6.91	7.24/7.30/6.94
Random Image 2 (R/G/B)	7.54/7.01/6.75	6.24/6.80/6.37	6.44/6.11/6.71
Random Image 3 (R/G/B)	6.42/6.10/6.26	6.96/6.37/6.92	7.42/7.36/6.81
Encrypted Lena (R/G/B)	7.78/7.97/7.90	7.21/7.45/7.12	8.11/8.43/8.52
Encrypted Lady (R/G/B)	8.89/8.68/8.65	8.69/8.33/8.12	8.06/8.70/8.27
Encrypted Fruits (R/G/B)	7.04/7.69/7.58	6.91/7.16/6.83	7.51/7.22/7.67

As may be seen, the PSNR results for the encrypted images and the random generated images are similar. This fact suggests that the encrypted artificially generated images are as distorted as the encrypted real-life images, and do not reveal information about the plaintext image content.

In addition, an important feature of the proposed method is its attribute to deteriorate elements of the encrypted image that would possibly reveal information relative with the nature of the image that was encrypted. The example of Figure 6 is provided for illustration of this fact, where the same key has been applied in order to encrypt a natural image (Lena) Figure 6(a) and a grayscale image depicting a document Figure 6(b). As may be observed, the brightness histograms of these images are quite different. Nevertheless, after the application of the method, the resulting histograms tend to coincide. The experiment was repeated twice for every image whereas some characteristics of the image histograms are presented in Table 2.

Evaluating the results in Table 2, as well as the histograms of the encrypted images, it is easily seen that in both cases the encrypted images tend to coincide. Consequently, it is impossible for an attacker to identify the nature of the image that was encrypted.

Finally, taking into account that the proposed method is based on CA, it is easily realized in hardware. In contrast to the serial computers, the implementation of the method is motivated by parallelism, an inherent feature of CA that contributes to further accelerating the method's operation. CA are perhaps the computational structures best suited for a fully parallel hardware realization [16] [12].

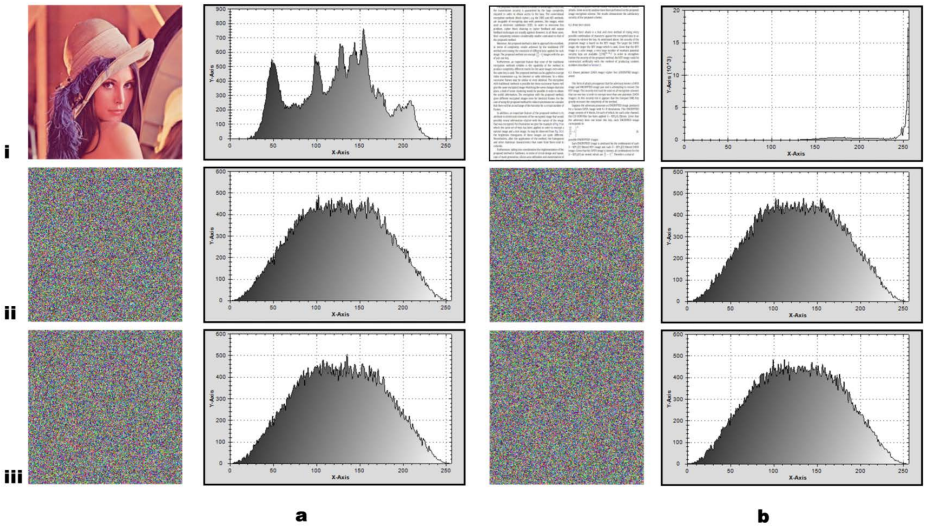


Fig. 6. Encrypting a (a) natural and a (b) grayscale image depicting a document using the same key

Table 2. Histograms Statistical Characteristics

	Mean	Median Area	Std. Dev.	Max. Prob
Lena	123.56	128	47.678	154
Encrypted Lena 1	127.02	127	49.444	102
Encrypted Lena 2	127.22	127	49.367	135
Document	228.16	250	35.414	255
Encrypted Document 1	127.15	127	49.241	134
Encrypted Document 2	127.21	127	49.347	112

5 Conclusions

This paper presented a visual multimedia content encryption method using the recursive attributes of the XOR filter on cellular automata. The decryption result is a lossless representation of the original encrypted image. Experimental results have shown that the resulted encrypted images do not contain statistical information able to reveal the source from which they originate (i.e. the plaintext image). Moreover, the proposed method appears to be able to withstand brute force attacks. Future work includes further analysis of the security of the method, by using tests such as the known plaintext - cipher text attack etc.

References

1. Bourbakis, N.G., Alexopoulos, C.: Picture data encryption using scan patterns. Pattern Recognition 25(6), 567–581 (1992)

2. Chatzichristofis, S.A., Mitzias, D.A., Sirakoulis, G.C., Boutalis, Y.S.: A novel cellular automata based technique for visual multimedia content encryption. Optics Communications 283(21), 4250–4260 (2010)
3. Chen, R.J., Lu, W.K., Lai, J.L.: Image encryption using progressive cellular automata substitution and scan. In: IEEE International Symposium on Circuits and Systems, ISCAS 2005, pp. 1690–1693 (2005)
4. Chen, R., Lai, J.: Image security system using recursive cellular automata substitution. Pattern Recognition 40(5), 1621–1631 (2007)
5. Chung, K., Chang, L.: Large encrypting binary images with higher security. Pattern Recognition Letters 19(5-6), 461–468 (1998)
6. Dasgupta, P., Chattopadhyay, S., Chaudhuri, P., Sengupta, I.: Cellular automata-based recursive pseudoexhaustive test pattern generator. IEEE Transactions on Computers 50(2), 177–185 (2001)
7. Gaoand, T., Chen, Z.: A new image encryption algorithm based on hyper-chaos. Physics Letters A 372(4) 21, 394–400 (2008)
8. Guan, P.: Cellular automaton public-key cryptosystem. Complex Systems 1, 51–56 (1987)
9. Jin, J., hong Wu, Z.: A secret image sharing based on neighborhood configurations of 2-d cellular automata. Optics & Laser Technology 44(3), 538–548 (2012)
10. Kari, J.: Cryptosystems based on reversible cellular automata. Personal communication (1992)
11. Koduru, S., Chandrasekaran, V.: Integrated confusion-diffusion mechanisms for chaos based image encryption, pp. 260–263 (2008)
12. Konstantinidis, K., Sirakoulis, G.C., Andreadis, I.: Design and implementation of a fuzzy-modified ant colony hardware structure for image retrieval. IEEE Transactions on Systems, Man, and Cybernetics, Part C 39(5), 520–533 (2009)
13. Kotoulas, L., Tsarouchis, D., Sirakoulis, G., Andreadis, I.: 1-d cellular automaton for pseudorandom number generation and its reconfigurable hardware implementation, p. 4 (2006)
14. Lafe, O.: Data compression and encryption using cellular automata transforms. Engineering Applications of Artificial Intelligence 10(6), 581–592 (1997)
15. Maniccam, S.S., Bourbakis, N.G.: Image and video encryption using scan patterns. Pattern Recognition 37(4), 725–737 (2004)
16. Nalpantidis, L., Sirakoulis, G.C., Gasteratos, A.: A Dense Stereo Correspondence Algorithm for Hardware Implementation with Enhanced Disparity Selection. In: Darzentas, J., Vouros, G.A., Vosinakis, S., Arnellos, A. (eds.) SETN 2008. LNCS (LNAI), vol. 5138, pp. 365–370. Springer, Heidelberg (2008)
17. Poullot, S., Buisson, O., Crucianu, M.: Z-grid-based probabilistic retrieval for scaling up content-based copy detection. In: CIVR, pp. 348–355 (2007)
18. Samet, H.: Foundations of Multidimensional and Metric Data Structur. Diane D. Cerra (2006)
19. Scharinger, J.: Fast encryption of image data using chaotic kolmogorov flows. Electronic Imaging 17(2), 318–325 (1998)
20. Seredynski, F., Bouvry, P., Zomaya, A.: Cellular automata computations and secret key cryptography. Parallel Computing 30(5-6), 753–766 (2004)