Asymmetric Image Encryption Approach with Plaintext-Related Diffusion

Jakub ORAVEC, Jan TURAN, Lubos OVSENIK, Tomas IVANIGA, David SOLUS, Michal MARTON

Dept. of Electronics and Multimedia Communications, Technical University of Kosice,
Bozeny Nemcovej 32, 040 01 Kosice, Slovakia

jakub.oravec@tuke.sk

Submitted June 7, 2017 / Accepted August 21, 2017

Abstract. *This paper deals with topic of image encryption based on chaotic maps. A solution which has advantage of robustness against chosen-plaintext attacks is proposed. Permutations of image pixels are carried out in a way that enables operations on grayscale images with arbitrary resolution. All calculations done with user key and also all diffusion processes employ the same chaotic map. This feature enables usage of look-up tables which reduce computational times. The paper includes several experiments which verify achieved results and also briefly describes advantages and drawbacks of proposed solution.*

Keywords

Arnold's cat map, confusion, diffusion, chaotic maps, image encryption, logistic map

1. Introduction

First application of chaos for purposes of encryption was proposed by Matthews in late 1980s [1]. Since then many chaotic encryption algorithms were described. One of the first papers which dealt specifically with image encryption was published by Fridrich in 1998 [2].

Fridrich's article provides scheme which was later used in many other solutions [3–5]. This scheme employs two operations with image pixels (they serve as plaintext). First one – *confusion* shuffles image pixels in a way that minimizes correlation between adjacent pixels. The other operation is *diffusion*, which tries to establish dependence between amplitudes of all image pixels. If two plaintext images differ in amplitude of only one pixel, good diffusion algorithm should result in two entirely different encrypted images.

Encryption algorithms designed for images have some advantages over conventional algorithms such as Advanced Encryption Standard (AES) [6], [7]. One of the most noticeable advantages is relatively small number and low difficulty of operations performed during encryption or decryption. This property is crucial for reaching fast computational times. However, it also causes one drawback of chaotic image encryption algorithms – it is quite easy to perform attacks, which try to retrieve plaintext image from its encrypted version. Brute-force attacks can be prevented by large key-space, possibility of statistical and some of differential attacks can be avoided by suitable diffusion algorithm. However, newer types of attacks could cause problems.

Fridrich's algorithm was broken in 2010 by an attack introduced by Solak et al. [8]. This attack changes amplitudes of plaintext image pixels and explores the dependencies between them and amplitudes of encrypted image pixels. Therefore, it could be classified as chosen-plaintext type of attack. Mentioned attack and its generalized versions are for purposes of this paper named simply as Solak's attack.

The rest of article is organized as follows: Sec. 2 contains brief survey of already published approaches. Sec. 3 describes all methods used by our proposal. Steps of encryption and decryption algorithms are mentioned in Sec. 4. Achieved results are discussed in Sec. 5. The paper ends with Sec. 6 which provides a review of advantages and drawbacks of proposed solution.

2. Related Work

Several ways for decreasing effects of Solak's attack or eliminating its possibility were proposed. The dependencies between amplitudes of plaintext pixels and encrypted pixels can be disturbed by modification of used key. Plaintext pixel amplitudes are used as input for hash functions by Zhang et al. in [9] and by Liu and Wang in [10]. Corresponding outputs are used as a set of parameters for next steps. Therefore, different images produce various encryption keys. Thus it is not possible to establish a list of pixel dependencies by testing of various images.

Another scheme consisting of two iterations of diffusion and one iteration of confusion was presented by Zhang in [11]. In this case, the relation between plaintext and key is created during confusion. Fu et al. [12] uses circular shift of key elements for producing different keys used by diffusion. Amount of shifting done in each iteration depends on values computed in previous iteration. Hence for first iteration, the shifting depends just on plaintext pixel amplitudes.

Proposal of Kanso and Ghebleh [13] adjusts number of chaotic map iterations according to plaintext image amplitudes. Results of the iterating are then used in the diffusion algorithm. Guanghui et al. uses multiple chaotic maps in [14], where the output of chaotic maps is divided into several parts. Each part is then employed for modification of key used in current iteration of encryption.

3. Used Methods

3.1 Logistic Map

Logistic map (LM) was introduced by May in 1976 [15]. May considered LM as a tool for modelling growths or decreases of wildlife population. LM can be also presented as one dimensional chaotic map, which maps $x(n) \in \langle 0, 1\rangle$ to $x(n+1)$ in the same range with respect to parameter $r \in (0, 4\rangle$ (1):

$$x(n+1) = r \cdot x(n) \cdot (1 - x(n)) \tag{1}$$

where n denotes iteration number.

Properties of LM depend on parameter r. Effects of various r can be viewed on bifurcation diagram (see Fig. 1).

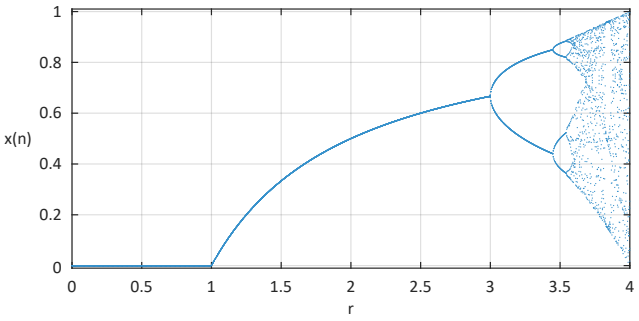

Fig. 1. Bifurcation diagram of logistic map.

As it can be seen, first bifurcation occurs when $r \sim 3$. The chaotic behavior of LM starts at $r \sim 3.56995$ which is also known as 'onset of chaos' [16]. However several islands of stability are present also after this point. One example of these islands is located at $r \sim 3.82843$.

3.2 Arnold's Cat Map

Arnold's cat map (ACM) was described in 1968 by Arnold and Avez [17] as an example of toral automorphism. Name of this map was chosen by picture of cat head which was used for experiments. ACM can be described as two dimensional chaotic map which preserves measure – the range of outputs stays the same as the range of inputs. Set of equations for discretized version ACM is given as (2):

$$\begin{bmatrix} x(n+1) \\ y(n+1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} x(n) \\ y(n) \end{bmatrix} \pmod N \tag{2}$$

where $x(n), y(n), x(n+1)$ and $y(n+1) \in \{0, 1, \ldots, N-1\}$, n denotes iteration number, N is the height and also width of image. Modulus of N in both equations restrains usage of ACM only to square images (resolution of $N \times N$ pixels).

As ACM maps each pair of coordinates $x(n)$, $y(n)$ to unique pair $x(n+1)$, $y(n+1)$, it is possible to construct inverse set of equations (3):

$$\begin{bmatrix} x(n-1) \\ y(n-1) \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x(n) \\ y(n) \end{bmatrix} \pmod N. \tag{3}$$

One known drawback of ACM is existence of *fixed points*. These are matrix elements which do not change their coordinates in consecutive iterations of ACM. An example of fixed point is shown on Fig. 2. The fixed point is indicated by gray color.

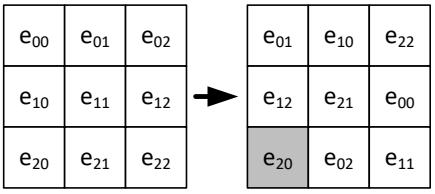

Fig. 2. Example of a fixed point.

In our solution, we tried to suppress occurence of fixed points by shifting elements prior to each iteration of ACM. The shift of element e with original coordinates l, k to new coordinates l', k' can be given as (4) for direct version of ACM and as (5) for inverse version of ACM:

$$(l', k') = (l+1, k+1) \pmod N, \tag{4}$$

$$(l', k') = (l-1, k-1) \pmod N \tag{5}$$

where l, k, l' and $k' \in \{0, 1, ..., N-1\}$, N is the height and also width of matrix or image.

The shifting ensures that matrix element with coordinates of a fixed point is changed before each iteration of ACM. Therefore this element would move to location which is not a fixed point and its coordinates would be changed in following iterations of ACM.

3.3 Ciphertext Chaining

Chaining of ciphertext helps to introduce dependencies between adjacent pixels of encrypted images. Hence it is important for establishing robustness against statistical and differential attacks. For spreading a change in amplitude of arbitrary pixel into amplitudes of all other pixels, it is possible to employ two iterations of a simple feedback (6):

$$f_i(n) = \begin{cases} f_0(0), & \text{if } n = 0, i = 1, \\ f_1(n) + f_1(\text{num}_P - 1) \pmod{2^L}, & \text{if } n = 0, i = 2, \\ f_{i-1}(n) + f_i(n-1) \pmod{2^L}, & \text{if } n \neq 0 \end{cases} \tag{6}$$

where $i \in \{1, 2\}$ denotes sequential number of iteration, f_i is vector after ith iteration of chaining, f_0 is a vector of image

pixel amplitudes prior to encryption, n denotes index of currently processed pixel, $n \in \{0, 1, ..., \text{num}_P - 1\}$, num_P is the total number of image pixels, L is the color depth of image.

Operation inverse to chaining done by (6) can be given as (7):

$$f_{i-1}(n) = \begin{cases} f_i(n) - f_i(n-1) \pmod{2^L}, & \text{if } n \neq 0, \\ f_2(n) - f_1(\text{num}_P - 1) \pmod{2^L}, & \text{if } n = 0, i = 2, \\ f_1(0), & \text{if } n = 0, i = 1. \end{cases} \tag{7}$$

3.4 Key Diffusion

The output of diffusion algorithm should be sensitive to even small differences between entered diffusion keys. Because our proposal uses diffusion key consisting of 8 bytes, these bytes need to be diffused prior to encryption or decryption. The principle of direct key diffusion is illustrated on Fig. 3.

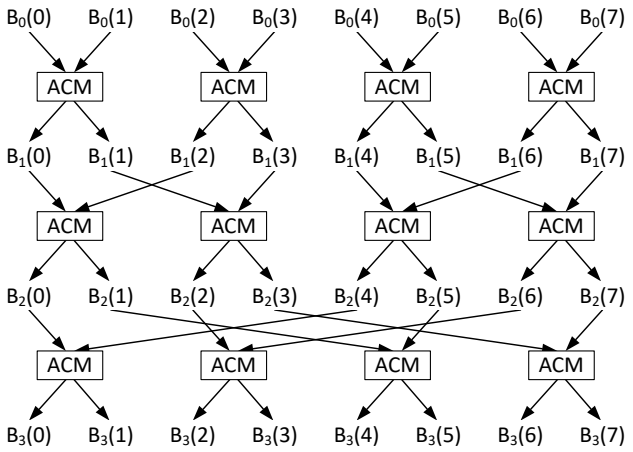

Fig. 3. Direct key diffusion diagram.

Key diffusion consists of three iterations which are used for creating dependencies between every possible pair of key bytes $B_0(b), b \in \{0, 1, \ldots, 7\}$. Inverse key diffusion used during decryption iterates in reverse order. In the case of direct key diffusion, block denoted as 'ACM' employs set of equations (2), inverse key diffusion uses set (3).

3.5 Key Expansion

In our solution, the dependencies between key and image pixels are introduced by using ACM, which takes one key byte and current pixel amplitude as inputs $x(n)$ and $y(n)$. Therefore one key byte is needed for each image pixel. Because keys of such length are not practical, we employ key expansion which enlarges the key to desired length.

Key expansion is related to one of properties of our algorithm – it uses different initial keys for encryption and for decryption. Because encryption uses expanded key from its start to its end, the decryption needs to start with end of expanded key. Thus it is necessary to provide last elements of expanded key to the decryption algorithm. Amount of these key elements is in presented case fixed to 8 bytes.

4. Proposed Solution

In this paper, we would like to describe a reasonably fast image encryption algorithm with results that are still sufficient by means of resistance against known types of attacks. The algorithm works with grayscale images with arbitrary resolution ($M \times N$ pixels). Confusion step uses logistic map for shuffling (permutation) of plaintext image pixels. Diffusion operates with results of Arnold's cat map.

4.1 Encryption

Encryption is done by following steps of Algorithms 1 and 2:

Algorithm 1: Confusion algorithm.

Input: grayscale plaintext image P, its height h and width w, 8 byte encryption keys key_x, key_y

Output: grayscale image after confusion C

1. Keys key_x and key_y are mapped to values $r'_x, r'_y \in \langle 0, 0.01)$.

2. Parameters of logistic maps (1) are set as $LM_{\text{rows}} : y(0) = 0.5, r_y = 3.99 + r'_y$ and $LM_{\text{cols}} : x(0) = 0.5, r_x = 3.99 + r'_x$.

3. Map LM_{rows} is iterated $h + 10$ times, map LM_{cols} is iterated $w + 10$ times.

4. Last h iterates of LM_{rows} are mapped to $sh_{\text{rows}} \in \{0, 1, \ldots, w - 1\}$ and last w iterates of LM_{cols} are mapped to $sh_{\text{cols}} \in \{0, 1, \ldots, h - 1\}$.

5. Each pixel with coordinates l, k in current image row l is shifted to new coordinates l, k': $(l, k') = (l, k + sh_{\text{rows}}(k) \pmod{w})$.

6. Each pixel with coordinates l, k' in current image column k' is shifted to new coordinates l', k': $(l', k') = (l + sh_{\text{cols}}(l) \pmod{h}, k')$.

As it can be seen, the logisitic maps are used for producing $h + 10$ and $w + 10$ iterates, respectively. First ten iterates are not used for pixel shifting but they help reaching sufficient chaotic properties of generated sequences.

The number of iterations of ACM is set as 11 for the same reason as for the LM – the map needs to produce results with chaotic behavior. Usage of $N = 256$ is caused by number of grayscale image pixel amplitudes and number of possible binary representations of one key byte. As these parameters of ACM are fixed, the computation speed could be improved through usage of look-up tables. These tables provide values of x', y' for all possible initial pairs of x, y.

Algorithm 2: Diffusion algorithm.

Input: grayscale image after confusion C, its height h and width w, 8 byte encryption key key_z

Output: grayscale encrypted image E, 8 byte decryption key key_d

1. Look-up tables for matrix of all possible inputs $x, y \in \{0, 1, \ldots, 255\}$ are created. These tables contain new coordinates of matrix elements after 11 iterations of Arnold's cat map (2).

2. Image after confusion C is reshaped to vector C_{vec} with 1 row and $h \cdot w$ columns.

3. Key key_z undergoes key diffusion (see Fig. 3). Then it is copied into first 8 elements of vector with extended key key'_z.

4. First iteration of diffusion. Each pixel from $C_{\text{vec}}(n)$ undergoes chaining by (6), resulting value overwrites its input. Then look-up tables are used for acquisition of pair $x'(n)$, $y'(n)$. Initial values are set as $x(n) = key'_z(n)$, $y(n) = C_{\text{vec}}(n)$.

5. Resulting $x'(n)$ are used as next bytes of extended key – they are copied into $key'_z(n+8)$. Computed $y'(n)$ are used as amplitudes of image pixels after first iteration of diffusion $D_{\text{vec}}(n)$.

6. Second iteration of diffusion. Each pixel from $D_{\text{vec}}(n)$ undergoes chaining by (6), resulting value overwrites its input. Then look-up tables are used for acquisition of pair $x'(n)$, $y'(n)$. Initial values are set as $x(n) = key'_z(n + h \cdot w)$, $y(n) = D_{\text{vec}}(n)$.

7. Resulting $x'(n)$ are used as next bytes of extended key – they are copied into $key'_z(n + h \cdot w + 8)$. Computed $y'(n)$ are used as amplitudes of encrypted image pixels $E_{\text{vec}}(n)$.

8. Vector E_{vec} is reshaped to matrix E with h rows and w columns.

9. Last 8 bytes of extended key key'_z are copied into vector key_d. Then this decryption key undergoes key diffusion (see Fig. 3).

4.2 Decryption

Decryption is in case of the proposed algorithm analogous to encryption. The only differences are present in the used key, equations and therefore also look-up tables. Because decryption algorithm needs to start with decryption of pixel amplitudes which were encrypted as last, it uses last 8 bytes of extended key which was produced during encryption. These bytes are then used as input for inverse key diffusion.

The key is then extended 'backwards' by look-up tables produced by inverse ACM (3). The tables are also used for computing pixel amplitudes after first round of diffusion. Then the effect of chaining is eliminated by (7). This process is repeated also for removing first iteraton of diffusion. Finally, decrypted image is achieved by performing inverse shifts of pixels in image rows and columns.

The safety of proposed solution is based on fact that potential attackers do not have access to decryption key. If key values would become compromised, the attackers should be able to use look-up tables and find pixel amplitudes which correspond to encrypted pixel amplitudes.

5. Experimental Results

Following experiments used three plaintext images. These images and their versions encrypted with key K_1 are shown on Fig. 4. Their resolution was 512×512 pixels in case of *lena*, 512×256 pixels for *black* and 256×256 pixels for image *f16*. The color depth of all images was 8 bits.

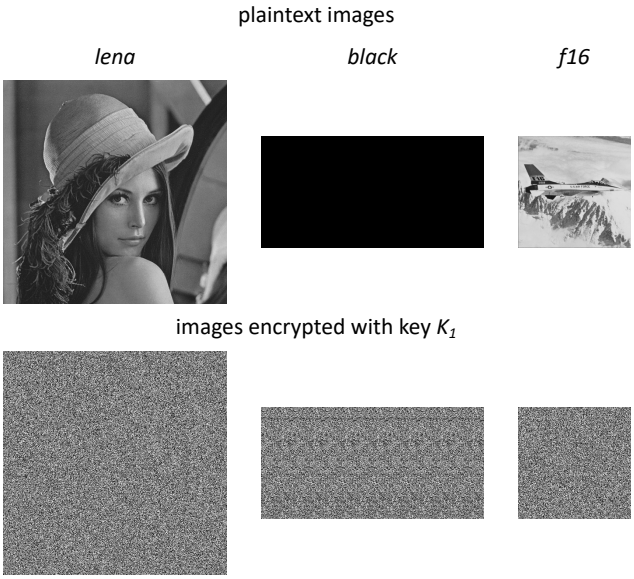

Fig. 4. Set of plaintext images and their encrypted versions.

Experiments used two kinds of keys: encryption keys K_1, K_2 which consisted of three parts key_x, key_y and key_z and decryption keys K'_1, K'_2 with their parts key_x, key_y and key_d. Differences between keys are indicated by bold characters, their values were set as:

- $K_1 = (key_x, key_y, key_z)$ = (0xA0B32465, 0xFD326667, 0x9745BC3470CD64EE),
- $K_2 = (key_x, key_y, key_z)$ = (0xA0B32465, 0xFD3266**68**, 0x9745BC3470CD64EE),
- $K'_1 = (key_x, key_y, key_d)$ = (0xA0B32465, 0xFD326667, 0x6E960C921BCC1FCD),
- $K'_2 = (key_x, key_y, key_d)$ = (0xA0B32465, 0xFD326668, 0x6E96**0B**921BCC1FCD).

5.1 Size of Key Space and Key Sensitivity

Key space is a set of all keys which could be used for encryption. In case of our algorithm the confusion keys key_x and key_y do not depend on diffusion key key_z and vice versa. This means that the key space includes all possible combinations of confusion and diffusion keys. As key_x and key_y are both represented by 4 bytes and key_z is given as 8 bytes, the size of keyspace can be computed as $\text{num}_k = 256^4 \cdot 256^4 \cdot 256^8 = 2^{32+32+64} = 2^{128}$.

If we would estimate the time necessary for decryption of image with resolution of 512×512 pixels as 100 ms, the brute-force attack would take approx. 1.079×10^{30} years. Thus this type of attack can be considered as not feasible.

Key sensitivity of our algorithms is shown on Fig. 5.

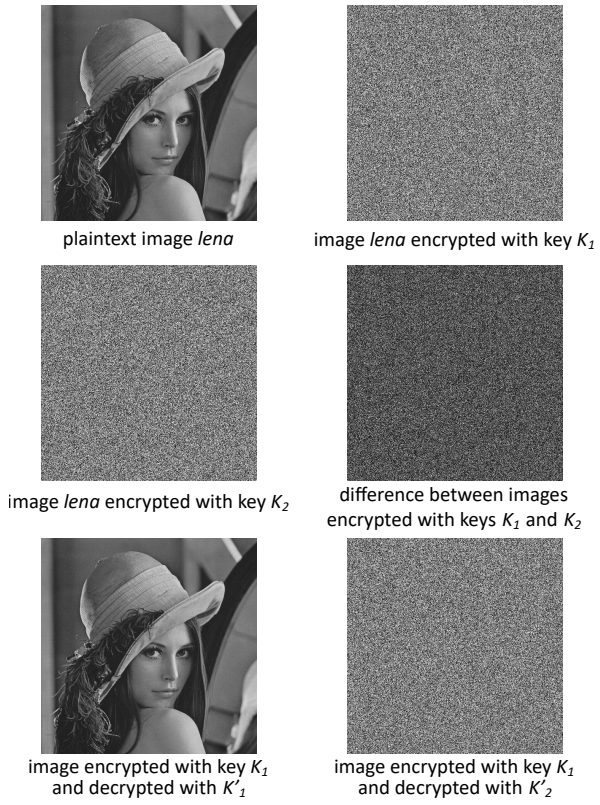

Fig. 5. Illustration of key sensitivity.

5.2 Statistical Attacks

These attacks compare properties of images before and after encryption. Ideally, an encrypted image should not provide any information about plaintext image. Level of robustness against statistical attacks could be evaluated by histograms, correlation diagrams and coefficients or by values of entropy.

Histograms of plaintext image *lena* and its version encrypted with key K_1 are shown on Fig. 6. The peaks present in first histogram are suppressed by encryption. This results

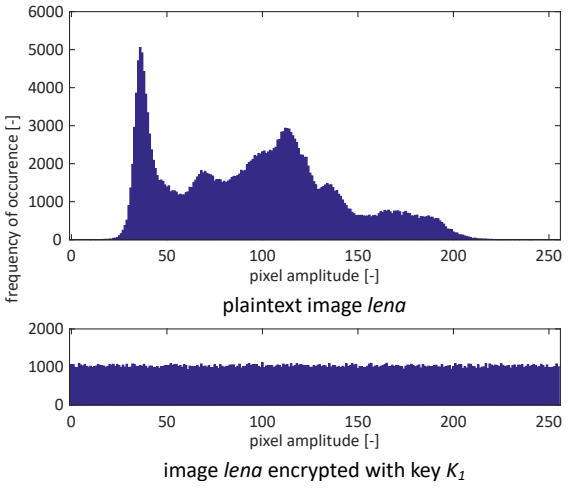

Fig. 6. Histograms before and after encryption.

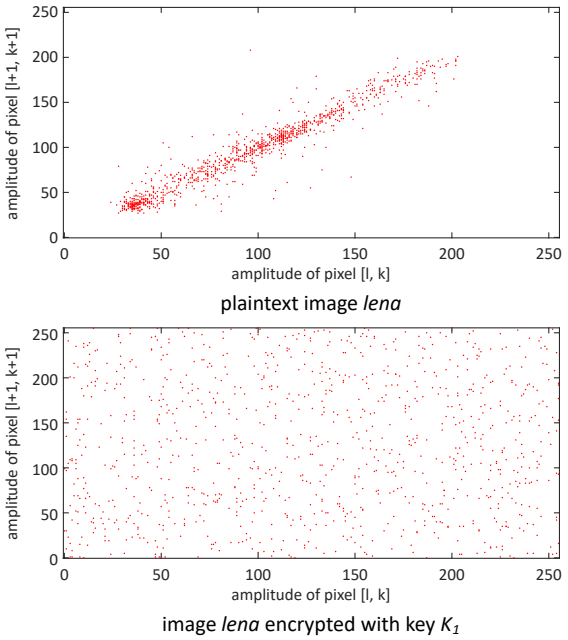

Fig. 7. Example of correlation diagrams.

in relatively uniform distribution of pixel amplitudes. Hence it can be concluded that statistical attacks are hardly possible.

Correlation diagrams display amplitudes of pairs of two adjacent image pixels on their axes. The adjacency is horizontal, vertical or diagonal. In an ideal case, the points should be located on the diagram with uniform distribution. The diagrams showing correlation of 1000 randomly chosen pairs of diagonally adjacent pixels for plaintext image *lena* and its version encrypted with key K_1 are displayed on Fig. 7.

Correlation coefficients ρ could be calculated by (8–10):

$$\rho = \frac{\text{cov}(P, E)}{\sqrt{\sigma_P^2 \cdot \sigma_E^2}}, \tag{8}$$

$$\text{cov}(P, E) = \sum_{l=0}^{h-1} \sum_{k=0}^{w-1} (P(l,k) - \bar{P}) \cdot (E(l,k) - \bar{E}), \tag{9}$$

$$\sigma^2_{Im} = \sum_{l=0}^{h-1}\sum_{k=0}^{w-1}(Im(l,k) - \bar{Im})^2 \tag{10}$$

where P and E denote plaintext and encrypted images, $\mathrm{cov}(P,E)$ is their covariance, σ^2_{Im} is dispersion of image Im, $\bar{Im}$ denotes its arithmetic mean, l and k are row and column indices, h is height and w is width of image in pixels.

Entropy can be viewed as a measure of randomness of an information flow. Maximal possible value of entropy is determined by amount of bits which represent one element of the flow. Hence for grayscale image the maximal entropy is set as 8 bits/pixel. Entropy H is calculated by using (11):

$$H = -\sum_{a=0}^{2^L-1} p(a)\cdot \log_2\left(p(a)\right) \text{ [bits/pixel]} \tag{11}$$

where L is color depth of image, $p(a)$ denotes probability of occurence of image pixel with amplitude a.

Calculated values of correlation coefficients ρ and entropy H are included in Tab. 1. Subscripts h, v and d denote horizontal, vertical or diagonal adjacency of pixels in 1000 randomly chosen pixel pairs. All presented values except for entropy are arithmetic means of 100 repeated measurements.

image	key	ρ_h	ρ_v	ρ_d	H [bits/pixel]
Plaintext images					
lena		0.9680	0.9761	0.9548	7.2344
black		inf *(division by zero)*			0.0000
f16		0.9555	0.9056	0.9005	6.7105
Encrypted images					
lena	K_1	0.0042	0.0022	−0.0045	7.9992
	K_2	0.0045	−0.0026	0.0052	7.9992
black	K_1	0.0076	−0.0065	0.0072	7.9570
	K_2	0.0088	−0.0068	0.0069	7.9570
f16	K_1	−0.0061	0.0054	0.0059	7.9972
	K_2	0.0048	0.0065	0.0058	7.9971

Tab. 1. Values of correlation coefficients and entropy.

5.3 Differential Attacks

Differential attacks investigate changes in encrypted images caused by modifications of corresponding plaintext images. Thus encryption algorithm should be sensitive even to small perturbations done in plaintext images.

Robustness against differential attacks can be evaluated by two measures. First one is called *Number of Pixel Change Rate* (NPCR). Its calculation requires two plaintext images P_1 and P_2, second one is a copy of first one with change of amplitude of one pixel. The size of this modification is minimal (one amplitude level). Then these images are encrypted as E_1 and E_2 and the value of NPCR is computed by (12):

$$\begin{aligned}\mathrm{NPCR} &= \frac{100}{h\cdot w}\sum_{l=0}^{h-1}\sum_{k=0}^{w-1}\mathrm{Diff}(l,k)\ [\%],\\ \mathrm{Diff}(l,k) &= \begin{cases}0, & \text{if } E_1(l,k) = E_2(l,k),\\ 1, & \text{if } E_1(l,k) \neq E_2(l,k)\end{cases}\end{aligned} \tag{12}$$

where l and k are row and column indices, h is height and w is width of image in pixels.

Second measure is known as *Unified Average Changing Intensity* (UACI). UACI also uses two encrypted images E_1 and E_2 which were created by the same way as for NPCR (13):

$$\mathrm{UACI} = \frac{100}{h\cdot w}\sum_{l=0}^{h-1}\sum_{k=0}^{w-1}\frac{|E_1(l,k) - E_2(l,k)|}{2^L - 1}\ [\%] \tag{13}$$

where l and k are row and column indices, h is height and w is width of image in pixels and L is its color depth.

The difference between NPCR and UACI is hidden in the way of evaluating difference of encrypted images. While NPCR reflects only the amount of pixels with different amplitude, values of UACI are affected also by the size of amplitude change. Computed values of NPCR and UACI are shown in Tab. 2. These values were acquired from set of 100 repeated measurements. The coordinates of pixel with modified amplitude were chosen randomly in each measurement.

image	key	minimal	arithm. mean	maximal
NPCR [%]				
lena	K_1	99.0063	99.1802	99.3469
	K_2	99.0250	99.1881	99.3443
black	K_1	99.0074	99.1727	99.3225
	K_2	99.0089	99.1730	99.3233
f16	K_1	99.0051	99.2060	99.4339
	K_2	99.0067	99.2185	99.4431
UACI [%]				
lena	K_1	33.0925	33.3483	33.5842
	K_2	33.1701	33.3818	33.5923
black	K_1	33.0690	33.2994	33.4963
	K_2	33.1369	33.3065	33.4601
f16	K_1	33.1132	33.3499	33.6126
	K_2	33.0741	33.3462	33.5756

Tab. 2. Calculated values of NPCR and UACI.

5.4 Relation Between Key and Plaintext

Previous paragraph contained an example of chosen-plaintext attack. Robustness against whole class of these attacks can be ensured by establishing a relation between used key and plaintext in form of pixel amplitudes.

Diffusion algorithm of our solution is based on usage of ACM, which takes parameters $x(n)$, $y(n)$ as its input. The output also consists of two parameters – $x(n+1)$ and $y(n+1)$. As the inputs are current byte of extended key $x(n)$ and value of processed pixel amplitude after chaining $y(n)$, the outputs of ACM also relate to them. First output, $x(n+1)$ is used for extending the key and second output, $y(n+1)$ represents amplitude of image pixel after diffusion.

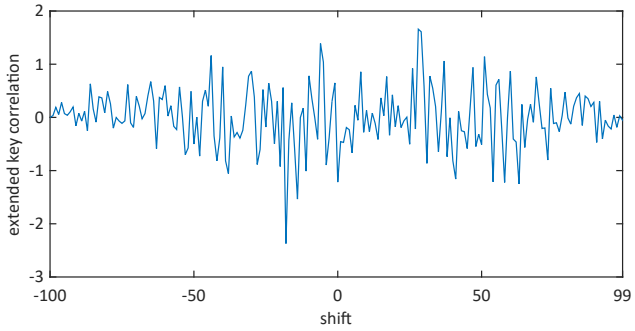

Fig. 8. Cross-correlation of parts of two extended keys.

These relations produce various extended keys for sets of images with minimal differences. Therefore it is not feasible to assume steps of encryption algorithm from testing of multiple plaintext images, because these images would result in different keys. Also the effects of key diffusion which is applied prior to encryption have to be taken into account.

Differences between various extended keys can be examined by their *cross-correlation*. Resulting function for first 100 elements of extended keys produced by second iteration of diffusion is illustrated on Fig. 8. Encryption with key K_1 was done on image *lena* (see Fig. 4), and its copy where amplitude of one pixel was changed by one level. Values of extended key elements were mapped to range $\langle -0.5, 0.5 \rangle$ before computation of cross-correlation.

5.5 Computational Difficulty

Speed tests of proposed algorithm were conducted in MATLAB R2015a on PC with 2.5 GHz CPU, 12 GB of RAM and Windows 10 operating system. The values presented in Tab. 3 were achieved by 100 repeated measurements.

Computational difficulty of algorithms can be compared by values of processing speed v_{proc}. This measure expresses amount of data which is encrypted during one second (14):

$$v_{\text{proc}} = \frac{h \cdot w \cdot L}{t \cdot 8 \cdot 10^6} \text{ [MB/s]} \tag{14}$$

where h is height and w is width of image, L is its color depth and t is time required for one encryption given in seconds. The values of v_{proc} presented in Tab. 3 are calculated from arithemetic means of measured durations.

5.6 Comparison with Other Approaches

The comparison of results is quite a hard task due to many differences between experiments in other papers. For instance, color versions of *lena* were used in [10], [13], while [9], [12] tested effects of their proposals on various other grayscale images. Also some of processing speed measurements were conducted on considerably slower machines [9], [11], [13]. Therefore only some parameters could be compared.

Results from Tab. 4 and Tab. 5 show that our solution achieves better values of processing speed v_{proc}, correlation coefficients ρ and UACI. However, its performance is not as good in case of entropy and NPCR. These drawbacks are possible topics for our future work.

image	**key**	**minimal**	**arithm. mean**	**maximal**	v_{proc} [MB/s]
Encryption times [ms]					
lena	K_1	81.5346	83.5082	91.7072	3.1391
	K_2	81.2470	83.3793	92.3348	3.1440
black	K_1	40.3856	42.0839	44.0148	3.0537
	K_2	40.3354	42.0500	44.1147	3.0158
f16	K_1	20.1917	21.4615	23.1964	3.1145
	K_2	20.1617	21.7306	23.5878	3.1171
Decryption times [ms]					
lena	K_1	104.3669	107.1934	111.1227	2.4455
	K_2	104.2005	107.3807	111.4821	2.4413
black	K_1	49.8841	51.6152	54.8236	2.5394
	K_2	50.0018	51.4642	54.8711	2.5469
f16	K_1	24.0211	25.0554	26.1471	2.6156
	K_2	24.3192	25.2162	36.3075	2.5990

Tab. 3. Measured durations of encryption and decryption.

approach		**proposed**	**Ref. [11]**	**Ref. [14]**
corr. coeff.	horizontal	*0.0042*	−0.0046	−0.0278
ρ – pixel	vertical	*0.0022*	−0.0511	−0.0065
adjacency	diagonal	*−0.0045*	−0.0065	−0.0074
entropy H [bits/pixel]		7.9992	*7.9993*	7.9895
NPCR [%]		99.3469	99.6108	*99.6600*
UACI [%]		*33.5923*	33.4679	33.5700

Tab. 4. Comparison of achieved numerical results.

approach	**proposed**	**Ref. [9]**	**Ref. [11]**	**Ref. [13]**
v_{proc} [MB/s]	*3.1440*	0.3703	0.3223	1.9300

Tab. 5. Comparison of processing speed.

6. Conclusion

This paper proposed an image encryption algorithm based on chaotic maps. Properties of Arnold's cat map were employed for creating relation between used key and plaintext in form of image pixel amplitudes. This correspondence seems crucial for establishing certain level of robustness against class of chosen-plaintext attacks which could create a list of dependencies between encrypted and plaintext image. Effects of presented algorithms were verified by series of experiments. Their numeric results were compared with values yielded by approaches which used the same plaintext image. Processing speeds were calculated for algorithms which provided sufficient information about used images.

Main advantage of our proposal is its simplicity which enables fast processing speed. Also the correlation coefficients of adjacent encrypted pixels are in case of presented image better than those achieved by other algorithms. However, these solutions provided higher values of NPCR. Future work can be done on the key diffusion algorithm, which currently restrains the length of entered diffusion key to 8 bytes.

Acknowledgments

This work was supported by following research grants: KEGA 023TUKE-4/2017, VEGA 1/0772/17 and ITMS 26220120055.

References

[1] MATTHEWS, R. On the derivation of a 'chaotic' encryption algorithm. *Cryptologia*, 1989, vol. 8, no. 1, p. 29–42. DOI: 10.1080/0161-118991863745

[2] FRIDRICH, J. Symmetric ciphers based on two–dimensional chaotic maps. *International Journal of Bifurcation and Chaos*, 1998, vol. 8, no. 6, p. 1259–1284. DOI: 10.1142/S021812749800098X

[3] CHEN, G., MAO, Y., CHUI, C. K. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons and Fractals*, 2004, vol. 21, no. 3, p. 749–761. DOI: 10.1016/j.chaos.2003.12.022

[4] MAO, Y., CHEN, G., LIAN, S. A novel fast image encryption scheme based on 3D chaotic baker maps. *International Journal of Bifurcation and Chaos*, 2004, vol. 14, no. 10, p. 3613–3624. DOI: 10.1142/S021812740401151X

[5] YE, R. A novel chaos-based image encryption scheme with an efficient permutation-diffusion mechanism. *Optics Communications*, 2011, vol. 284, no. 22, p. 5290–5298. DOI: 10.1016/j.optcom.2011.07.070

[6] FIPS *Publication 197: Specification for the Advanced Encryption Standard (AES)*. 2001, 47 pages. [Online] Cited 2017-04-27. Available at: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[7] HAJDUK, V., BRODA, M., KOVAC, et al. Image steganography with QR code and cryptography. In *Proceedings of the Radioelektronika 2016*. Kosice (Slovakia), 2016, p. 350–353. DOI: 10.1109/RADIOELEK.2016.7477370

[8] SOLAK, E., COKAL, C., YILDIZ, O. T., et al. Cryptanalysis of Fridrich's chaotic image encryption. *International Journal of Bifurcation and Chaos*, 2010, vol. 20, no. 5, p. 1405–1413. DOI: 10.1142/S0218127410026563

[9] ZHANG, X., ZHU, G., MA, S. Remote-sensing image encryption in hybrid domains. *Optics Communications*, 2012, vol. 285, no. 7, p. 1736–1743. DOI: 10.1016/j.optcom.2011.12.023

[10] LIU, H., WANG, X. Triple-image encryption scheme based on one-time key stream generated by chaos and plain images. *Journal of Systems and Software*, 2013, vol. 86, no. 3, p. 826–834. DOI: 10.1016/j.jss.2012.11.026

[11] ZHANG, Y. A chaotic system based image encryption algorithm using plaintext-related confusion. *Telkomnika*, 2014, vol. 12, no. 11, p. 7952–7962. DOI: 10.11591/telkomnika.v12i11.6480

[12] FU, C., HOU, S., ZHOU, W., et al. A chaos-based image encryption scheme with a plaintext related diffusion. In *Proceedings of the 9th International Conference on Information, Communications and Signal Processing ICICS 2013*. Tainan (Taiwan), 2013, p. 1–5. DOI: 10.1109/ICICS.2013.6782914

[13] KANSO, A., GHEBLEH, M. A novel image encryption algorithm based on a 3D chaotic map. *Communications in Nonlinear Science and Numerical Simulation*, 2012, vol. 17, no. 2, p. 2943–2959. DOI: 10.1016/j.cnsns.2011.11.030

[14] GUANGHUI, C., KAI, H., YIZHI, Z., et al. Chaotic image encryption based on running-key related to plaintext. *The Scientific World Journal*, 2014, vol. 15, no. 1, p. 1–9. DOI: 10.1155/2014/490179

[15] MAY, R. Simple mathematical models with very complicated dynamics. *Nature*, 1976, vol. 261, no. 5560, p. 459–467. DOI: 10.1038/261459a0

[16] GLEICK, J. *Chaos.* London: Vintage Books, 1998. ISBN: 978-0-749-38606-1

[17] ARNOLD, V. I., AVEZ, A. *Ergodic Problems of Classical Mechanics.* New Jersey: W. A. Benjamin, 1968.

About the Authors . . .

Jakub ORAVEC received his M.Sc. degree from Department of Electronics and Multimedia Communications (DEMC), Technical University of Košice in 2015. Now he continues as PhD. student. His research interests include image encryption, steganography and digital image processing.

Ján TURÁN received his M.Sc. and RNDr. degrees from Czech Technical University, Prague in 1974 and Charles University, Prague in 1980. He received his PhD. and DrSc. degrees from Technical University of Košice in 1983 and 1992. Since March 1979, he works at the Technical University of Košice, currently as a Full Professor. His research is focused on digital signal processing and fiber optics.

Ľuboš OVSENÍK received his M.Sc. and PhD. degrees from DEMC, Technical University of Košice in 1990 and 2002. He works at the Technical University of Košice, currently as an Associate Professor. His research interests are fiber optic communication systems and sensor networks.

Tomáš IVANIGA is currently a PhD. student at DEMC, Technical University of Košice. His research interests include fiber optic systems.

Dávid SOLUS is currently a PhD. student at DEMC, Technical University of Košice. His research is focused on usage of optical correlators in digital image processing.

Michal MÁRTON is currently a PhD. student at DEMC, Technical University of Košice. His research interests are free space optics and optical gyroscope systems.