UDC 004.93.056.55 **doi: 10.32620/reks.2024.2.12**

Viktor AVRAMENKO, Mykyta BONDARENKO

Sumy State University, Sumy, Ukraine

IMAGE CRYPTOSYSTEM WITH IMAGE KEY USING INTEGRAL DISPROPORTION

*The **subject matter** of this article is the process of securing visual data through encryption. The **goal** of this study is to develop a new approach for image encryption and decryption using an image as a key, where security and performance are comparable or superior to alternatives. The **tasks** to be solved are as follows: developing a symmetric image cryptosystem for both greyscale and colored images in any format that can use an arbitrary image as a security key; adapting the capabilities of integral disproportionality for image encryption. The **encryption method** applies an integral disproportion formula for each pair of pixels' components for both the input image and the key image starting from the first one. The output is a ciphered object represented as an array of real numbers, with decryption being the inverse process. The **result** is a novel approach to image encryption, proposing a new cryptosystem in which an arbitrary image of varying size, format, or content serves as the encryption key. The proposed symmetric full-encryption cryptosystem is applicable to both greyscale and colored images in any format. The method employs a frequency-based approach that influences the values of pixels rather than their positions. In contrast to mainstream approaches in this area, the representation of a cipher is not an image but a binary array. The proposed algorithm is simple to implement. The proposed method appears to meet the required requirements in both security and performance. **Conclusions**. The scientific novelty of the results obtained is as follows: This research introduces a new image encryption method, which is unique in its application of integral disproportionality to alter pixel values, differing from traditional encryption methods. This provides a practical yet secure option to the existing encryption techniques*.

Keywords: *symmetric cryptosystem; image encryption; image decryption;, integral disproportionality.*

Introduction

A common approach to data protection involves the use of traditional symmetric and asymmetric cryptosystems, such as AES and RSA. Because these approaches operate directly on the byte representation of data, they are content-agnostic, meaning they do not require specific knowledge of the content type or format being encrypted. Whatever the data, AES and RSA can be applied, allowing encryption of various content types, including images.

However, the nature of data significantly impacts the reliability and performance of encryption. Visual data exhibits distinct characteristics, including strong correlations between adjacent pixels, high capacity, spatial redundancy, and visual patterns that can be exploited. Additionally, the larger data size of images, particularly in high-resolution formats, necessitates efficient encryption techniques to ensure both security and performance [1].

These characteristics underscore the need to create and refine specific cryptosystems tailored for working with images. Image encryption algorithms are designed to address the unique challenges posed by visual data. It is particularly crucial in domains where protecting images is of utmost importance, such as IoT [2 military map encryption [3], securing medical images [1], and document scanning [4, 5].

Among the different approaches, there is an idea to use another image as a key. Using an image as a cryptographic key presents a distinctive approach for enhancing security in image encryption. The inherent complexity and variability of images contribute to a vast and diverse key space, bolstering resistance against traditional cryptographic attacks. This method not only introduces an additional layer of security but also offers an intuitive and user-friendly aspect to the encryption process, as users can choose or upload visually meaningful images [4].

This work introduces a novel approach to image encryption, proposing a new cryptosystem in which an arbitrary image of varying size, format, or content serves as the encryption key. The proposed symmetric full-encryption cryptosystem is applicable to both greyscale and colored images in any format. The method employs a frequency-based approach that influences pixels' values rather than their positions. In contrast to mainstream approaches in this sphere, the representation of the cipher is not an image itself but a binary blob. The proposed algorithm is straightforward and simple to implement. The proposed method appears to meet the required requirements in both security and performance.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

The object of this research is the process of securing visual data. The subject of this research is a novel method for encrypting images using a key image. The purpose of this study is to utilize the peculiarities of image format to create a cryptosystem that can address both security and performance issues. The research task is as follows:

1. Explore the possibilities of encrypting images using another image as the key.

2. Create a new method that can match or surpass the security and efficiency of existing alternatives in this area.

3. Establish a symmetric image cryptosystem suitable for grayscale and colored images in any format, using integral disproportion to adjust pixel values.

4. Evaluate correctness, efficiency, and reliability by demonstrating the system's functionality with practical examples.

The novelty is the use of an approach based on integral disproportion, which was not used before. Despite the atypical approach, the results indicate its ability to maintain the integrity of encrypted data and demonstrate its reliability, decorrelation capability, and key sensitivity. It showcases a potential new direction for image encryption.

1. Current Research Analysis

In the study [6], the authors analyze the shortcomings of using traditional algorithms like AES for encrypting images. They highlight a significant security issue: the same encrypted data will result in the same cipher value. This issue is particularly prevalent in images with homogeneous zones, where identical blocks remain the same after encryption, leading to textured zones in the encrypted image and suboptimal entropy. To address these issues, they suggest modifying AES to incorporate a keystream generator tailored for image encryption, which helps overcome the textured zone problem. However, the analysis focuses only on grayscale images.

Some researchers have incorporated traditional algorithms into new methodologies. For example, [7] introduces an asymmetric cryptosystem in which the RSA public key is used to generate initial values for a quantum logistic map. These values enable the creation of pseudo-random keystream sequences using the chaotic map, and a three-dimensional quantum logistic map ensures a sufficiently large key space to ward off brute force attacks.

In general, the adoption of dynamic chaos-based image cryptosystems is gaining traction. Recent works include [8, 9] and [10]. Variations of this approach include combinations with other methods, such as the Combined Cellular Automata algorithm described in [11].

However, these methods have not seen widespread adoption due to drawbacks like low encryption algorithm performance, complex procedures, and lengthy decryption times, as noted in [12]. In addition, as mentioned in [13], hybrid parallel chaotic maps for image encryption often suffer from a limited key space.

An exhaustive analysis and classification of contemporary methods is conducted in [14]. This study describes several operations available for image encryption, including scrambling, diffusion, shuffling, rotation, substitution, confusion, and transposition. Diffusion and permutation are particularly popular because of their effectiveness and simplicity. Image encryption algorithms are typically categorized into full encryption and partial encryption algorithms, which are classified according to their domain orientation, whether spatial, frequency, or a hybrid of both. In addition, this research provides a deep analysis of different modern methods using specific security metrics.

Besides the mentioned approaches, there is an interesting concept of using another image to generate a key or employing the image itself as a key of using another image to generate a key or employing the image itself as a key. One of the first introductions of this concept was in a paper [15], which proposed two new algorithms that use images to derive either a bit plane or an edge map as the key. These algorithms decompose the original image into binary bit planes, which are then encrypted using XOR operations with the key image. The order of all bit planes is inverted, and the encrypted image is obtained by applying a scrambling algorithm.

A similar principle with added complexity is demonstrated in [16], where flexibility in various components of the encryption process is introduced. This method involves selecting a bitplane decomposition method to generate a bitplane of a source image, which then acts as the security key bitplane to encrypt bitplanes of the original image using XOR operations. The algorithm applies a scrambling algorithm for bit-level permutation and combines all processed bitplanes to produce the encrypted image. Thus, the security key consists of not only the key image but also the chosen decomposition algorithm, selected bitplane, and scrambling method. Although it may increase security, such an approach risk complicating key management.

In general, XOR operation usage on bitplanes is the cornerstone principle in all reviewed methods that employ images as keys. This class of methods is noted for their sufficiently large security key spaces arising from the numerous possible images that can be used to generate the key image, thereby enhancing resistance to brute force attacks.

Numerous publications in this direction have been published by Mohammed Ali Hussain et al. In [1], two algorithms are proposed. The first, which is limited to greyscale images, uses XOR for direct pixel-by-pixel image encryption. The second encrypt images through XOR with a randomly selected bitplane from the key image, which is further complicated by the process of shuffling bitplanes in a specific sequence.

The following works [4, 17, 18, 5] generally describe three variations: key bitplane encryption (KBE), key SCAN encryption (KSE), and key RGB displacement (KRDE).

KSE modifies the original and key images in the spatial domain using algorithms described by the SCAN language.

In KRDE, the original image and the key image are split into their basic RGB components, followed by XOR operations and scrambling to generate the cipher image.

Different security metrics for such methods are provided in [4, 5, 19]. However, compared with the algorithms reviewed in [14], the given values of entropy, mean value, pixel correlation, and UACI are suboptimal. Although their NPCR values are comparable, this indicates an area for potential security improvement in cryptosystems with image keys.

It should be noted, all the discussed methods result in an encrypted image. However, the so-called KRDE method in [5] produces a binary blob. This method involves performing XOR operations on each RGB component of the original and key image, followed by various calculations to generate a 3D array as the encrypted output. Unfortunately, this method lacks a detailed security analysis. Nonetheless, it can be assumed that abandoning the maintenance of the image properties in the encrypted object might result in security advantages.

A markedly different approach to encryption is explored in [12], which examines and formalizes the concept of integral cryptography. It notes that while most current cryptographic resilience approaches rely on the complexity of solving factorization, discrete logarithms (for asymmetric cryptosystems), or combinatorial complexity (for symmetric cryptosystems), there are other mathematical problems whose complexity could underpin the cryptographic strength of encryption algorithms. An additional advantage of this proposed model is the absence of effective cryptoanalysis algorithms, due to the limited prevalence of integral cryptography in contemporary cybersecurity systems.

Although this study does not directly address the security of images, it demonstrates the potential for developing new approaches to constructing cryptosystems.

One such approach is presented in [20], which uses the property of integral disproportionality, as described in [21]. The proposed symmetric cryptosystem encrypt text using real-variable functions as the key. Utilization of the disproportionality function makes the message invariant to the amplitude of the signal transmitted over the communication channel. Another example is [22], where a symmetric key cryptosystem uses the sum of real-type functions with random amplitude. However, these methods do not focus on image encryption.

The class of cryptosystems that use images as keys presents certain advantages, but is currently limited to algorithms that primarily employ XOR operations on bitplanes. Considering their drawbacks, it would be beneficial to explore the construction of image cryptosystems based on alternative mathematical principles, such as integral disproportion. Furthermore, it may be advantageous to not restrict ourselves to the principle where the encrypted output remains an image. Venturing beyond these conventional methods could pave the way for more robust and versatile cryptographic solutions in image encryption.

2. Materials and methods of research

2.1. Integral disproportion

Disproportion functions serve to examine proportional relationships within numerical functions and are categorized into several types: the disproportion over the n-th order derivative, the disproportion over the n-th order value, and sequential and relative disproportions.

The expression for the disproportion of the n-th order derivative of a function y(x) with respect to x is given as follows:

$$@\,d_x^{(n)}y = \frac{y}{x^n} - \frac{1}{n!}\cdot\frac{d^n y}{dx^n}, \qquad (1)$$

where @ is a symbol for calculating the disproportion;

d is a derivative;

n is a derivative's order;

y is a function;

x is an argument.

The disproportion (1) evaluates to zero if the function y(x) takes the form $y = kx^n$ and is invariant with respect to the proportionality coefficient k.

For the first order derivative (n = 1), the disproportion is expressed as:

$$@\,d_x^{(1)}y = \frac{y}{x} - \frac{dy}{dx}, \qquad (2)$$

where d is the derivative;

y and x are functions.

If the functions y(x) and f(x) are defined parametrically by x, the disproportion (2) takes the form:

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

$$@\,d^{(1)}_{f(x)}y(x) = \frac{y(x)}{f(x)} - \frac{dy/dx}{df/dx}, \quad (3)$$

where x is a parameter;

y(x) and f(x) are functions defined parametrically.

If there is a proportional dependence y(x) = f(x) between the two functions y(x) and f(x), the disproportion (3) is zero over the entire domain, irrespective of the proportionality coefficient's magnitude.

If the function is discrete or constant at some intervals, the direct application of the above disproportion (3) is not feasible. The pixel values should take only integers and don't have first derivatives. Thus, without using disproportion (3), employing the first-order Integral Disproportion is preferable. The Integral Disproportion of the function y(x) with respect to f(x) is as follows:

$$@\,I^{(1)}_{f(x)}y(x) = \frac{\int_{x-h}^{x} y(x)dx}{\int_{x-h}^{x} f(x)dx} - \frac{y(x)}{f(x)}, \quad (4)$$

where x is a parameter;

y(x) and f(x) are functions considered as one-dimensional arrays;

h is a step of changing a parameter;

If the approximate values of the integrals can be calculated using the trapezoid formula, then following the uniform step h for both y(x) and f(x), the expression (4) simplifies to:

$$@\,I^{(1)}_{f_i}y_i = \frac{y_{i-1} + y_i}{f_{i-1} + f_i} - \frac{y_i}{f_i}, \quad (5)$$

where y and f are arrays;

i is an index, i = 1…N;

y_0 is the first element of the array;

N is the number of array elements, determined by step h and the selected interval;

The trapezoidal rule is chosen because it offers an easy and practical formula for computing integral disproportions, while still being reliable from a practical perspective.

Expression (6) defines the inverse transformation through integral disproportion (5):

$$y_i = \frac{f_i \cdot (y_{i-1} - I_i \cdot (f_{i-1} + f_i))}{f_{i-1}}, \quad (6)$$

where y and f are arrays;

I is an integral disproportion;

i is an index.

Equations (5) and (6) are the core for the encryption and decryption processes in the proposed algorithm.

The properties of the mathematical framework of the integral disproportion of the first order (5) and its inverse transformation (6) allow us to use it for building cryptosystem algorithms. These equations are used for the encryption and decryption processes, respectively.

The algorithms are predicated on the use of a so-called “key-image” to encrypt the original image. The methodology revolves around the core mechanism of encrypting an image by processing the RGBA values of the corresponding pixels by the pixels of the key image.

2.2. Encryption

To recall, equation (5) considers an integral disproportion of function `f` with respect to function `y`, each represented by a one-dimensional array. In the case of image processing, we have a one-dimensional array of pixels for both the original image and the key image.

For simplicity, let's consider a case with a gray-scale image first. In such schemes, the color pallet is represented by only 1 byte, where the value `0` is black and `255` is white. This means that each pixel is represented by one value, and the entire image can be represented as a one-dimensional array of integers. This allows us to apply the integral disproportion for image encryption. Considering `f` is a key-image, and `y` is an image, equation (5) takes the form:

$$C_i = \frac{I_{i-1} + I_i}{E_{i-1} + E_i} - \frac{I_i}{E_i}, \quad (7)$$

where C is a cipher array;

I is an image array;

E is the key-image array;

i is an index.

The result of the integral disproportion calculation is also a one-dimensional array, but of real numbers. This array is a ciphered image (generally referred to as ciphertext), which can be transmitted through the channel as a binary blob.

As can be seen in formula (7), the i-th pixel is encrypted using the value of the previous pixel i-1, so it is necessary to have an initial anchor value to start calculations. Thus, encryption (7) starts from the second pixel (i = 1), while the first pixel is not encrypted and is kept as is. This so-called open pixel is transmitted openly. Although this might seem like a potential weak spot, a singular pixel is unlikely to divulge any significant information about the content of the entire image.

For colored images, other schemes are used, where the most popular one is RGBA. Each pixel in the image is characterized by four components - Red (R), Green

(G), Blue (B), and Alpha (A). The Alpha component represents the pixel's opacity.

Thus, the colored image can be represented as 4 one-dimensional arrays of integers.

Because each component is still a 1-byte integer value, we can apply the same method (7) to them:

$$R_{C_i} = \frac{R_{I_{i-1}} + R_{I_i}}{R_{E_{i-1}} + R_{E_i}} - \frac{R_{I_i}}{R_{E_i}}, \quad (8)$$

$$G_{C_i} = \frac{G_{I_{i-1}} + G_{I_i}}{G_{E_{i-1}} + G_{E_i}} - \frac{G_{I_i}}{G_{E_i}}, \quad (9)$$

$$B_{C_i} = \frac{B_{I_{i-1}} + B_{I_i}}{B_{E_{i-1}} + B_{E_i}} - \frac{B_{I_i}}{B_{E_i}}, \quad (10)$$

$$A_{C_i} = \frac{A_{I_{i-1}} + A_{I_i}}{A_{E_{i-1}} + A_{E_i}} - \frac{A_{I_i}}{A_{E_i}}, \quad (11)$$

where R, G, B, A are corresponding components;

C is a ciphered image array of the corresponding components;

I is an image array of the corresponding components;

E is a key-image array of the corresponding components;

i is an index.

Here, we have four one-dimensional arrays of real numbers, each of which represents the corresponding ciphered components of the image. Altogether, they are the ciphertext of the colored image, which is then transmitted to the communication channel. From the technical side, they can be transmitted separately or simultaneously.

It should be noted that the encrypted sequence contains information about pixel values, but not their coordinates. The receiver side should be able to correctly place pixels after decryption and ensure the original proportions of the image. Thus, the width of the input image is also transmitted alongside the ciphertext.

In general, the encryption core algorithm is as follows:

1. Read the input image pixel-by-pixel and extract the RGBA values.

2. Read the key image pixel-by-pixel and extract RGBA values.

3. Take the first pixel of the input. Keep it as is. It is not encrypted and transmitted openly.

4. Starting from the second one, each pixel's component is processed using the integral disproportion (formulas 8-11). The result of such calculation is a real number representing the encrypted RGBA component of the pixel. Thus, the encrypted pixel is a sequence of 4 real numbers. The encrypted image is a sequence of encrypted pixels.

5. The resulting sequence is transmitted over the communication channel along with the width of the original image.

2.3. Ciphertext

In contrast with other image cryptosystems, the ciphertext is not an image itself, but arrays, a so-called binary blob.

In the case of a grayscale image, it is a one one-dimensional array of real type, where the size of the array equals to the number of pixels in the image. Each value represents the ciphered pixel value of the grey-scale image.

In the case of the RGBA image, it is four one-dimensional arrays of real type. Their sizes are the same and also equal to the number of pixels in the image. Each value represents the ciphered pixel value of the corresponding RGBA component.

They may be transmitted sequentially or separately. (Table 1) and (Table 2) show how the sequence may be treated after encryption.

In the first case (Table 1), the resulting sequence is transmitted pixel by pixel, i.e., ciphered RGBA values for the first pixel and ciphered RGBA values for the second pixel. This case is straightforward and simple to implement.

In the second case (Table 2), we transmit four arrays, i.e., all ciphered R values, then all G values, B and A. It allows for splitting the ciphertext and transmitting it separately in case of necessity, e.g., via a time shift or using different communication channels.

Table 1

Sequentially transmitted sequence

Input	Ciphertext	Transmitted Sequence
$R_0\ R_1\ R_i$ ...	$R_0\ R_1\ R_i$ ...	$R_0\ G_0\ B_0\ A_0\ R_1\ G_1\ B_1\ A_1\ R_i\ G_i\ B_i\ A_i$ ...
$G_0\ G_1\ G_i$ ...	$G_0\ G_1\ G_i$ ...	
$B_0\ B_1\ B_i$ ...	$B_0\ B_1\ B_i$ ...	
$A_0\ A_1\ A_i$ ...	$A_0\ A_1\ A_i$ ...	

Table 2

Separately transmitted sequence

Input	Ciphertext	Transmitted Sequence
$R_0\ R_1\ R_i$ ...	$R_0\ R_1\ R_i$ ...	$R_0\ R_1\ R_i$ ... $G_0\ G_1\ G_i$ ... $B_{\_0}\ B_1\ B_i$ ... $A_0\ A_1\ A_i$ ...
$G_0\ G_1\ G_i$ ...	$G_0\ G_1\ G_i$ ...	
$B_0\ B_1\ B_i$ ...	$B_0\ B_1\ B_i$ ...	
$A_0\ A_1\ A_i$ ...	$A_0\ A_1\ A_i$ ...	

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

2.4. Decryption

The core of the decryption process relies on the inverse transformation from equation (6).

Again, let's consider a grayscale image first. In this case, each pixel of the image is described by one byte, and the ciphertext is an array of real numbers obtained by calculating the integral disproportion of the key image with respect to the input image in formula (7). Thus, the inverse transformation (6) allows the restoration of values of input image pixels by key-image and ciphertext:

$$D_i = \frac{E_i \cdot (D_{i-1} - C_i (E_{i-1} + E_i))}{E_{i-1}}, \quad (12)$$

where D is a deciphered array;

C is a ciphered image array of the corresponding components;

E is a key-image array of the corresponding components;

i is an index.

Similar to encryption (7), the decryption calculation (12) is started from the second pixel (i=1), since the first one is not encrypted. The ciphertext for the RGBA colored image consists of four arrays of real numbers calculated in the same way, one for each component. Applying formula (12) to each of them leads to the following decryption equations:

$$R_{D_i} = \frac{R_{E_i} \cdot (R_{D_{i-1}} - R_{C_i} \cdot (R_{E_{i-1}} + R_{E_i}))}{R_{E_{i-1}}}, \quad (13)$$

$$G_{D_i} = \frac{G_{E_i} \cdot (G_{D_{i-1}} - G_{C_i} \cdot (G_{E_{i-1}} + G_{E_i}))}{G_{E_{i-1}}}, \quad (14)$$

$$B_{D_i} = \frac{B_{E_i} \cdot (B_{D_{i-1}} - B_{C_i} \cdot (B_{E_{i-1}} + B_{E_i}))}{B_{E_{i-1}}}, \quad (15)$$

$$A_{D_i} = \frac{A_{E_i} \cdot (A_{D_{i-1}} - A_{C_i} \cdot (A_{E_{i-1}} + A_{E_i}))}{A_{E_{i-1}}}, \quad (16)$$

where R, G, B, and A are the corresponding components;

D is a deciphered image array of the corresponding components;

C is a ciphered image array of the corresponding components;

E is a key-image array of the corresponding components;

i is an index.

The result of such calculation is four arrays, where each value represents the decoded RGBA value of the original input image. Note that since the resulting array of equations (12) – (16) is still of real number type, we have to round each value to the closest integer to obtain the correct RGBA value:

$$x := \text{round}(x), \quad (17)$$

where x is an element of the array of a real number;

round() is a rounding function for the closest integer.

The last step is to collect these values to restore the image. Because the width parameter was also transmitted, the position of each pixel was calculated as follows:

$$x = i \% w, \quad (18)$$

where x is the x coordinate of the restored pixel;

i is an index of current pixel;

w is a width of the image.

$$y = \left\lfloor \frac{i}{w} \right\rfloor, \quad (19)$$

where y is the y coordinate of the restored pixel;

i is an index of the current pixel;

w is the width of the image.

Decryption core algorithm is as follows:

1. Read the key-image pixel-by-pixel.
2. Receive the encrypted sequence and the width parameter.
3. The first received pixel is not encrypted, keep it as is.
4. Process each real value of the ciphertext with inverse transformation (Equations (13) – (16)) for decryption: The result of this calculation is a real number that represents the encrypted RGBA component of the pixel.
5. Round this number to the closest integer (Equation (17)). The result is an encrypted RGBA component of the pixel.
6. Calculate the coordinates of each decrypted pixel using equations (18) and (19).
7. Collect pixels to restore the image.

2.5. Technical nuances and corner cases

The key-image can be any arbitrary image, with no specific requirements for size or format. Being pivotal for encryption and decryption, it is essential that both the transmitter and receiver have an identical copy of this key image. The key image should be picked or transmitted using a secure channel.

Due to the algorithm conditions, the number of pixels in the key image should coincide with those in the input image. Thus, an image of a different size (width and height) but with the same pixel number can be used out-of-box.

However, there is a requirement to use the arbitrary image as a key image, so the key image should be preliminarily resized. The proposed built-in solution is just to cut off pixels if the key-image is bigger than the input image or repeatedly expand the pixels of the key-image from the beginning if it's smaller. This solution is simple and straightforward, though the specific resizing technique may be configurable and be a part of the security key. For example, visual stretching is another option, but the specific stretching algorithm should be adjusted.

Using an incorrect key image (even slightly different) for decryption would result in a completely distorted image, akin to random noise. This underscores the importance of accurate key management.

Because RGBA are integers from 0 to 255, there can be division by 0 during encryption in equation (7). This is a corner case that should be handled. The proposed solution is to temporarily increment each RGBA before encryption and decrement them after decryption to obtain the original value. In other words, the pixel range is virtually offset from (0...255) to (1...256).

Since the algorithm works directly with pixels' values, it is applicable to any image format (JPG, PNG, BMP, etc.) and may be used for different color schemas, like grayscale, RGB, RGBA, HSV, and LUN. On the other hand, this means that the algorithm does not process the format-specific metadata, which leads to the fact that the file of the decrypted image is smaller than the original one, despite the image being the same.

2.6. Computational Complexity and Storage Requirements

The computational complexity of this algorithm is linear, O(N), where N is the number of pixels in the image. While the encrypted image requires four times more space than the original image, it is worth noting that a 32-bit float precision is used to represent the encrypted data instead of a 64-bit float. This choice not only reduces the storage requirements but also ensures, after rigorous testing, that the visual fidelity of the decrypted image is uncompromised, rendering it identical to the original on a pixel-by-pixel basis. To prevent division by zero during encryption, each RGBA value (originally ranging from 0 to 255) is incremented by 1. Since the value of 256 does not fit into u8 (one-byte unsigned integer type), a temporary u16 type is used to avoid overflow during this operation. After decryption, the values are decremented by 1 to retrieve the original image values.

The total space complexity for the algorithm is O(N), where N is the size of the input image in pixels. The auxiliary space complexity is O (1) because the algorithm requires only a fixed set of temporary variables beyond the input data. These variables include the aforementioned temporary u16 container, the first pixel's value, the image width, etc., none of which are dependent on the input size.

In addition, the algorithm may be easily parallelized by independently processing different RGBA values.

3. Experiments and Results

To illustrate the work of the algorithm, a step-by-step demonstration of the minimal example is provided. Let's encrypt the "input. PNG" (Fig. 1) using "key.PNG" (Fig. 2) as a key. Both images are simple pictures of size 2x2 (4 pixels and 16 integer values), with RGBA color scheme in PNG format. The values of pixels for the input image are described in (Table 3) and key-image values in (Table 4). Here, they are randomly generated, including random opacity, for demonstration. The PNG format was selected to support opacity.

Fig. 1. "input.PNG"

Fig. 2. "key.PNG"

Fig. 3. "output.PNG"

Table 3

Pixels of input image

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	204	62	215	218
G	107	151	132	211
B	1	76	124	196
A	254	210	234	218

Table 4

Pixels of key-image

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	71	5	151	0
G	155	175	131	0
B	34	165	10	0
A	207	249	240	251

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

After the encryption process described in the previous section, the ciphertext is generated. Though it isn't an image, it still consists of 16 values, but of real type. They are provided in (Table 5). As mentioned before, the open pixel (Column 1) is not encrypted, and the values are just cast to the real type. The remaining values look like random real numbers without exposing correlations with the original image. Note that the range of ciphertext values is not limited by the original value (from 0 to 255). After the decryption, the result is "output. PNG" (Fig. 3), whose values are described in (Table 6). As can be seen, (Table 3) and (Table 6) are equal, so decryption works correctly.

Table 5

Ciphertext

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	204.0	-9.229857	0.344770	-216.156860
G	107.0	0.051857	-0.082251	-209.406020
B	1.0	0.006383	-10.222393	-170.166670
A	254.0	0.078772	-0.066753	0.051845

Table 6

Decrypted image

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	204	62	215	218
G	107	151	132	211
B	1	76	124	196
A	254	210	234	218

To underscore the algorithm's handling of correlation, an edge case is examined where "red.PNG," a 2x2 image composed entirely of identical pixels, is encrypted. Each pixel in this image possesses RGBA values of (255, 0, 0, 255), rendering it uniformly red and opaque. Employing "key.PNG" (Fig. 2 and Table 4) as the key once again, the encryption process yields the ciphertext detailed in (Table 7). Remarkably, although the plaintext pixels are homogeneous, the resulting ciphertext values for the corresponding pixels diverge significantly, demonstrating the algorithm's effective decorrelation capability.

Table 7

Red ciphertext

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	255.0	-40.712470	1.556296	-252.653600
G	0.0	0.0056176167	-0.0010822513	-0.9849624
B	0.0	0.005951952	-0.07960966	-0.8333333
A	255.0	-0.012142301	-0.01947081	0.022666454

To emphasize the importance of using the correct key for decryption, consider a scenario in which a single pixel's value in the key image is altered. For instance, if we modify the first pixel of "key.PNG" (Fig. 2) from its original value of (5, 175, 165, 249) to (0, 0, 0, 0), effectively making it completely black and opaque, the impact on decryption becomes evident. Decrypting the ciphertext from (Table 5) with this modified key image results in the distorted output presented in (Table 8) and visualized in (Fig. 4). The altered image reveals that, aside from the first pixel, the rest of the data is incorrect and bears no resemblance to the original image, underscoring the algorithm's sensitivity to the precise key.

Table 8

Incorrect decryption

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	204	9	255	174
G	107	0	32	222
B	1	0	79	52
A	254	0	18	179

Fig. 4. Incorrect decryption

The algorithm's robustness was thoroughly tested on a diverse set of images, varying in both size and format. This was exemplified using both standard and custom images. A "Cat.JPG" (Fig. 5), which is an image with dimensions of 408x36 in JPG format, shows the author's cat lying on a desk. A "baboon.png" (Fig. 6), an example of standard image from the USC SIPI Image Database, has dimensions of 512x512 in PNG format. Encrypting the cat image (Fig. 5) using the baboon image (Fig. 6) as the key resulted in a decrypted image (Fig. 7) that perfectly mirrored the original image, thus validating the accuracy of the decryption process.

Note that the decrypted image (Fig. 7) is identical to the original image (Fig. 5); here, all 124848 pixels have the same RGBA values. It was tested programmatically in this case and many others.

The testing methodology relies on pixel-by-pixel comparison, where each RGBA component of a pixel in the original image is compared with the corresponding RGBA component of a pixel in the decrypted image. Thus, the images are identical and there is no loss in image quality, which demonstrates the reliability of the algorithm.

Fig. 5. “Cat.JPG” as input image

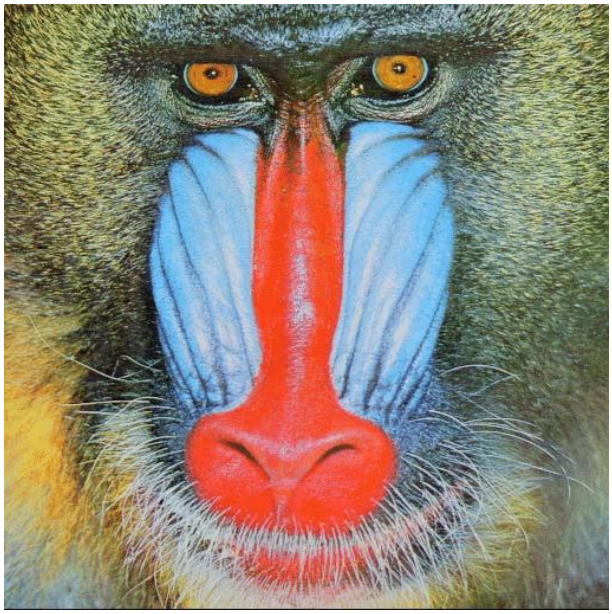

Fig. 6. “Baboon.PNG” as the key-image

Fig. 7. Decrypted “Cat.JPG”

To address potential concerns regarding the randomness of data in minimal examples, a more complex test was conducted. "Cat.JPG" (Fig. 5) was encrypted using "baboon.PNG" (Fig. 6) as the key-image. Subsequently, the second pixel of the key-image was altered to (0, 0, 0, 0), and the ciphertext was decrypted with this modified key. As can be seen in the result (Fig. 8), the decrypted image is corrupted. It confirms in real examples that even a slightly changed key (incorrect 1 pixel) leads to unsuccessful decryption, demonstrating the security of the proposed algorithm.

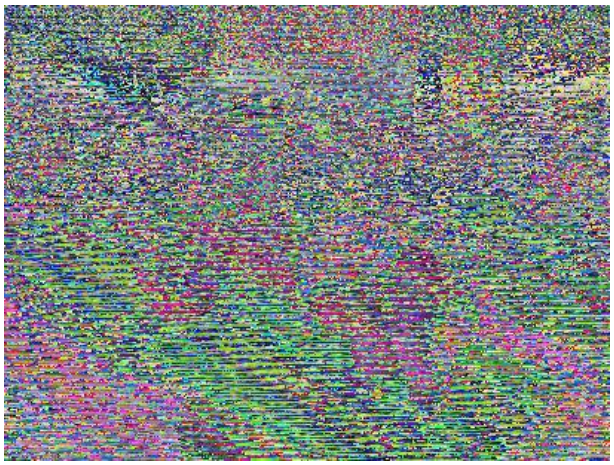

Fig. 8. Decrypted “Cat.JPG”
with a corrupted key-image

The (Fig. 8) was compared with the original image using the same pixel-by-pixel method. It shows that 124847 of 124848 pixels are different. The only untouched pixel is first, which is not encrypted and transmitted openly by design. Moreover, 373441 RGB components of 374544 are different. This means, except for the first one, there are no pixels, where at least one of the RGB values coincides with the original one. However, the A values of images are untouched because both input and key images are opaque and the A value is always 255.

Additionally, we can measure the quality metrics of decrypted images. Mean Normalized Square Error (MNSE) is a metric used to quantify the similarity between two images. The MNSE is calculated by comparing the pixel values of the original image I and the decrypted image D:

$$M = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{I_i - D_i}{I_i}\right)^2, \qquad (20)$$

where N is the total number of pixels;

I – is a pixel’s value of original image;

D – is a pixel’s value of decrypted image;

i – is an index number of the current pixel.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

The MNSE value of Fig.5 and Fig.7 is 0.00, which proves that they are identical and that there is no loss of quality during decryption.

It should be noted that each pixel component in the image corresponds to the real number in the ciphertext. This means that each byte of the original image transforms into 4 bytes in the ciphertext object. Followed by a few extra bytes (first pixel and width), it significantly exceeds the original image size. Therefore, the ciphertext size is at least four times larger, while the specific ratio depends on the format and compression. In our example, the size of the image (Fig. 5) is 240.2 kB, the size of the ciphertext object is 1.9 mB, and the decrypted image (Fig. 7) is 254.6 kB.

In addition to the provided examples, the algorithm was also tested on other images. These included screenshots of web pages (Fig. 9), scans of author's personal documents (Fig. 10), and an image of a hacker cat generated by ChatGPT's AI (Fig. 11). Their MNSE values are also 0.00. Scaled-down versions of these images are provided for space considerations.

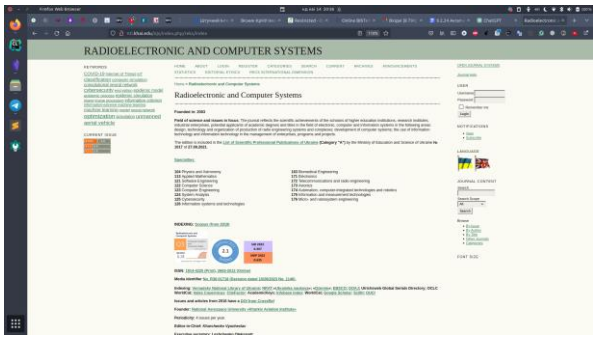

Fig. 9. Tested screenshot

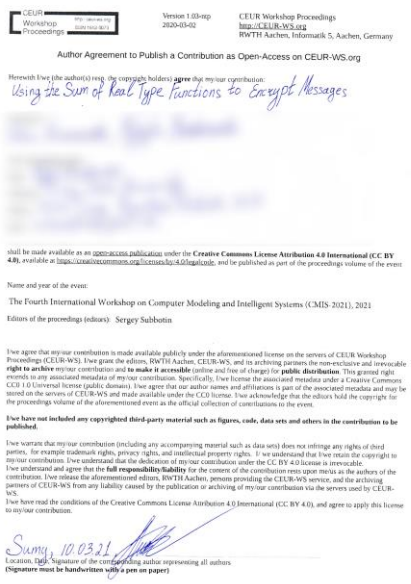

Fig. 10. Tested document scan

Fig. 11. Tested AI-generated image

4. Discussion

The experiments, particularly the encryption and decryption of standard images, demonstrate the robustness of the proposed image encryption algorithm. The use of an arbitrary image as the encryption key, a method not commonly explored in conventional encryption techniques, has proven effective. Notably, the algorithm's ability to handle images with uniform color distribution and maintain the integrity of encrypted data underscores its potential in diverse applications.

The algorithm’s success in handling images of various sizes and formats suggests its adaptability in real-world scenarios, ranging from secure image transmission to data protection in fields like healthcare and digital media. Furthermore, the algorithm’s sensitivity to key alterations, demonstrated through key-image modification tests, emphasizes its potential in scenarios where high security is paramount.

This research, while advancing the field of image encryption, identifies distinct areas for improvement. Some of them are related to the entire class of methods. For example, a universal challenge in symmetric cryptosystems is the secure sharing of the encryption key. Although this is not a direct limitation of our algorithm, it remains a critical factor for its practical deployment.

Similar to all reviewed methods of image encryption, our cryptosystem is focused only on the picture’s content. This means that the metadata that is specific to the format is not encrypted and not transmitted at all. Therefore, the size of the decrypted image is smaller than the original one, and they have different binary representations. From practical use, it means, for instance, different hashing results, and this fact should be considered.

In contrast to other methods, the proposed cryptosystem has unique limitations that offer opportunities

for improvement. The most notable disadvantage is the increase in the byte size of the ciphertext, which is approximately four times larger than that of the input image. This can be suboptimal for transmitting large files, suggesting the need for enhanced data compression or encoding techniques to reduce the ciphertext size without compromising the encryption integrity.

The asymptotic time and space complexity of the algorithm are comparable with known alternatives and sufficient for use on the client side, such as in web and mobile applications. However, the increase in ciphertext size may be bandwidth intensive. Also, it may not be suitable for embedded systems with limited resources.

Another possible problem is the transmission of the open pixel, which leaves it exposed in the ciphertext, theoretically posing a potential vulnerability.

The most important part of further research is a deep analysis of security and durability. The main task is to adapt the traditional security metrics of image cryptosystems, such as entropy, pixel correlation, NPCR, and UACI, to make them suitable for scenarios where the ciphertext is not an image.

The other part of further enhancement is related to tuning the optimal parameters of the algorithm. First, identifying the most secure types of images to use as keys requires extensive research and understanding which characteristics or criteria make certain images more suitable as key-images, thereby enhancing the overall encryption security. The second is selecting the right resizing algorithm, which can provide the optimal balance between security, performance, and usability.

Another important part of further work includes analysis, optimization, and practical recommendations for the implementation of the provided algorithm. Among others, optimization regarding memory hierarchy should be considered. It would also be beneficial to explore the implementation nuances for specific platforms, such as WebAssembly.

Conclusions

A novel approach to image encryption and decryption has been developed, which is characterized by the use of an arbitrary image as the encryption key. This symmetric full-encryption cryptosystem is versatile, accommodating both grayscale and colored images across various formats. Diverging from mainstream methods in this field, the cipher generated by our system is not a traditional image but a binary array, marking a significant departure from typical practices.

At the core of the proposed method is a unique mathematical foundation previously unexplored in the realm of image encryption. It employs the integral disproportion of the first order, which is applied to each pair of pixel components in both the image and the key image, beginning with the first pixel. This approach expands the capabilities of cryptographic algorithms in which an image serves as the security key. The examples provided demonstrate the method's effectiveness and reliability.

However, the method does have its limitations, the most notable being a fourfold increase in the size of the ciphertext. A meticulous investigation into the security and robustness of the proposed method is essential, using diverse criteria to assess its performance. This forms the primary objective of future research. In addition, determining the optimal selection of key-images and resizing algorithms represents another crucial area for exploration.

Contributions of authors: conceptualization, methodology, formulation of tasks, development of method – **Viktor Avramenko**; development of model, analysis software, verification, analysis, writing and editing, – **Mykyta Bondarenko**.

Conflict of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

This research was conducted without financial support.

Data Availability

The manuscript has no associated data.

Use of Artificial Intelligence

The authors confirm that they did not use artificial intelligence methods while creating the presented work.

All the authors have read and agreed to the published version of this manuscript.

References

1. Madhu, S., & Hussain, M. A. Securing Medical Images by Image Encryption using Key Image. *International Journal of Computer Applications*, 2014, vol. 104, iss. 3, pp. 30-34. DOI: 10.5120/18184-9079.

2. Makarichev, V., Lukin, V., Illiashenko, O., & Kharchenko, V. Digital Image Representation by Atomic Functions: The Compression and Protection of Data for Edge Computing in IoT Systems. *Sensors*, 2022, vol. 22, iss. 10, article no. 3751. DOI: 10.3390/s22103751.

3. Albahadily, H. K., Altaay, A. A. J., & Jabbar, I. A. Encryption of Military Maps Images Using Peter De

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Jong Chaotic Maps and Lightwaght Encryption. *2022 Fifth College of Science International Conference of Recent Trends in Information Technology (CSCTIT)*, Baghdad, IEEE, 2022, pp. 137-142. DOI: 10.1109/CSCTIT56299.2022.10145728.

4. Madhu, S., Hussain, M. A., Leelavathy, N., & Sujatha, B. Development of Secure and Novel Methods of Image Encryption Using an Image as Key. Research *Highlights in Mathematics and Computer Science Vol. 7*, 2023, pp. 89-102. DOI: 10.9734/bpi/rhmcs/v7/4297E.

5. Archana, N., Manogna, S., Hussain, D. M. A., & Razia, D. S. Secure Sharing of Documents using Image Encryption and Key Image. *International Journal of Recent Technology and Engineering (IJRTE)*, 2019, vol. 8, iss. 4, pp. 9415-9419. DOI: 10.35940/ijrte.D9724.118419.

6. Zeghid, M., Machhout, M., Khriji, L., Baganne, A., & Tourki, R. A Modified AES Based Algorithm for Image Encryption. *International Journal on Computer Science and Engineering*, 2007, vol. 1, iss. 1, pp. 745-750. DOI: 10.5281/zenodo.1334059.

7. Ye, G., Jiao, K., Huang, X., Goi, B., & Yap, W. An image encryption scheme based on public key cryptosystem and quantum logistic map. *Scientific Reports*, 2020, vol. 10, iss. 1, pp. 21044. DOI: 10.1038/s41598-020-78127-2.

8. Neamah, A. A., & Shukur, A. A. A Novel Conservative Chaotic System Involved in Hyperbolic Functions and Its Application to Design an Efficient Colour Image Encryption Scheme. *Symmetry*, 2023, vol. 15, iss. 8, article no. 1511. DOI: 10.3390/sym15081511.

9. Li, M., Fang, X., & Ernest, A. A Color Image Encryption Method Based on Dynamic Selection Chaotic System and Singular Value Decomposition. *Mathematics*, 2023, vol. 11, iss. 15, article no. 3274. DOI: 10.3390/math11153274.

10. Hussain, A. Z., & Khodher, M. A. A. Medical image encryption using multi chaotic maps. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 2023, vol. 21, iss. 3, article no. 556. DOI: 10.12928/telkomnika.v21i3.24324.

11. Choi, U. S., Cho, S. J., & Kang, S. W. New Color Image Encryption for Medical Images Based on Three Dimensional Generalized Chaotic Cat Map and Combined Cellular Automata. *Advances in Science, Technology and Engineering Systems Journal*, 2020, vol. 5, iss. 2, pp. 104-110. DOI: 10.25046/aj050213.

12. Hryshchuk, R., & Hryshchuk, O. A GENERALIZED MODEL OF FREDHOLM'S CRYPTOSYSTEM. *Cybersecurity: Education Science Technique*, 2019, vol. 4, iss. 4, pp. 14-23. DOI: 10.28925/2663-4023.2019.4.1423. (In Ukrainian)/

13. Kari, A. P., Navin, A. H., Bidgoli, A. M., & Mirnia, M. A new image encryption scheme based on hybrid chaotic maps. *Multimedia Tools and Applications*, 2021, vol. 80, iss. 2, pp. 2753-2772. DOI: 10.1007/s11042-020-09648-1.

14. Dawood, A., Thabit, Q., & Fahad, T. A Comprehensive Review of Color Image Encryption Technology. *Basrah journal for engineering science*, 2023, vol. 23, iss. 1, pp. 56-63. DOI: 10.33971/bjes.23.1.8.

15. Zhou, Y., Panetta, K., & Agaian, S. Image encryption using binary key-images. *2009 IEEE International Conference on Systems, Man and Cybernetics*, San Antonio, IEEE, 2009, pp. 4569-4574. DOI: 10.1109/ICSMC.2009.5346780.

16. Zhou, Y., Cao, W., & Chen, C. L. P. Image encryption using binary bitplane. *Signal Processing*, 2014, vol. 100, pp. 197-207. DOI: 10.1016/j.sigpro.2014.01.020.

17. Somaraj, S., & AliHussain, M. An Image Encryption Technique Using Scan Based Approach and Image as Key. *Advances in Intelligent Systems and Computing*, 2017, vol. 507, pp. 645-653. DOI: 10.1007/978-981-10-2471-9_62.

18. Somaraj, S., & Hussain, M. A. A Novel Image Encryption Technique Using RGB Pixel Displacement for Color Images. *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, Bhimavaram, IEEE, 2016, pp. 275-279. DOI: 10.1109/IACC.2016.59.

19. Somaraj, S., & Hussain, M. A. Performance and Security Analysis for Image Encryption using Key Image. *Indian Journal of Science and Technology*, 2015, vol. 8, iss. 35, pp. 1-4. DOI: 10.17485/ijst/2015/v8i35/73141.

20. Avramenko, V., & Demianenko, V. Serial encryption using the functions of real variable. *Radioelectronic and computer systems*, 2021, no. 2, pp. 39-50. DOI: 10.32620/reks.2021.2.04.

21. Karpenko, A. P. Integral'nye kharakteristiki neproportsional'nosti chislovykh funktsii i ikh primenenie v diagnostike [Integral characteristics of disproportionality of numerical functions and their application in diagnostics]. *Visnyk Sums'koho derzhavnoho universytetu – Newsletter of Sumy State University*, 2000, vol. 16, pp. 20-25. Available at: https://essuir.sumdu.edu.ua/bitstream-download/123456789/10931/1/4_Karpenko.pdf (accessed Jan. 3, 2024) (In Russian).

22. Avramenko, V., Bondarenko, M. Using the Sum of Real Type Functions to Encrypt Messages. *CEUR Workshop Proceedings*, 2021, vol. 3200, pp. 10-17.

Received 09.02.2024, Accepted 15.04.2024.

КРИПТОСИСТЕМА ДЛЯ ЗОБРАЖЕНЬ З ЗОБРАЖЕННЯМ-КЛЮЧЕМ З ВИКОРИСТАННЯМ ІНТЕГРАЛЬНОЇ НЕПРОПОРЦІЙНОСТІ

В. В. Авраменко, М. О. Бондаренко

Предметом вивчення в статті є процес захисту візуальних даних шляхом шифрування. **Метою** є розробка нового підходу до шифрування та дешифрування зображень, що використовує інше зображення в якості ключа, для якого стійкість та швидкодія співрозмірні або перевершують альтернативи. **Завдання**: Розробити симетричну криптосистему для чорно-білих та кольорових зображень, в якій довільне зображення використовується як ключ безпеки; адаптувати можливості інтегральної непропорційності до задачі шифрування і дешифрування зображень. **Метод** полягає у застосування формули інтегральної непропорційності для кожної пари компонентів пікселів починаючи з першого, як для вхідного зображення, так і для зображення-ключа. Зашифрований об'єкт представлений у вигляді бінарного масиву дійсних чисел, а дешифрування є оберненим процесом. **Результатом** є новий підхід до шифрування зображень, де довільне зображення будь-якого розміру, формату або наповнення може слугувати ключем. Запропоновано повна симетрична криптосистема застосовується як для чорно-білих, так і кольорових зображень будь-якого формату. Метод використовує частотний підхід, впливаючи на значення пікселів, а не на їхні позиції. На відміну від основних підходів у цій галузі, представлення шифру є не іншим зображенням, а бінарним масивом. Запропонований алгоритм простий у впровадженні. Метод відповідає необхідним вимогам як у безпеці, так і у продуктивності. **Висновки**. Наукова новизна отриманих результатів полягає в наступному: це дослідження вводить новий метод шифрування зображень, який є унікальним у своєму застосуванні інтегральної непропорційності для зміни значень пікселів, відрізняючись від традиційних методів шифрування. Це привносить практичну, але безпечну альтернативу до вже існуючих технік шифрування.

Ключові слова: симетричні криптосистеми; шифрування зображень; дешифрування зображень; інтегральна непропорційність.

Авраменко Віктор Васильович – канд. техн. наук, доц. каф. комп'ютерних наук, Сумський державний університет, Суми, Україна.

Бондаренко Микита Олегович – аспірант, Сумський державний університет, Суми, Україна.

Viktor Avramenko – Associate Professor, Department of Computer Science, Sumy State University, Sumy, Ukraine,
e-mail: vv.avramenko@cs.sumdu.edu.ua, ORCID: 0000-0002-6317-6711, Scopus Author ID: 57195345847.

Mykyta Bondarenko – PhD student, Sumy State University, Sumy, Ukraine,
e-mail: nikbond97@gmail.com, ORCID: 0000-0002-8849-7378.