Image Steganography Technique Using Algebraic Fractals

Oleg Sheluhin, Dzhennet Magomedova
Moscow Technical University of Communication and Informatics
Moscow, Russia
sheluhin@mail.ru, jimagomedova@gmail.com

***Abstract*—We consider a solution to the problem of copyright protection by embedding a secret message into a still color or grayscale image using algebraic fractals. The peculiarity of the proposed steganographic method is the use of a two-dimensional fractal image of algebraic type as an intermediate cover image. For this purpose, it is proposed to use a fractal image of algebraic type in the form of a Julia set as an intermediate cover image. An attacker will not be able to generate an identical fractal image without the exact value of some complex number , which is agreed in advance between the sender and the recipient, as well as a number of other parameters, which makes the proposed method resistant to attacks. The advantage of the proposed algorithm is the ability to extract the watermark without knowledge of the original cover image, since the intermediate container for the watermark is a fractal. In this paper we consider fractal image generation techniques. Then generated fractals are used as cover image in steganography system. Watermark embeds in blue component of JPEG image. To assess the quality of obtained results, a quality assessment was made based on the following metrics: normalized mean square error (NMSE) and peak signal-to-noise ratio (PSNR), expressed in decibels. The results of the original cover image and stego image, as well as the original and extracted watermark, are presented. Analysis of the results of the evaluation of the quality of the original cover image and watermarked image, as well as the original and extracted watermark showed that the proposed algorithm provides high quality of hiding confidential information. The use of fractals as an intermediate cover image for the extraction of watermark allows ensuring the extraction practically without losses, while the stego image visually does not differ from the container containing classified information.**

I. INTRODUCTION

One of the developing areas in digital image processing is the use of fractal analysis. With the help of fractal image analysis it is possible to determine its statistical regularities (self-similar areas) inherent in all images [1], [2].

Fractal geometry is a concept proposed by Benoit Mandelbrot to describe geometric objects having a rugged shape and possessing the property of self-similarity. Self-similarity as the main characteristic of the fractal means the immutability of the main geometric features of the fractal when the scale changes. From a mathematical point of view, a fractal can be defined as an object whose Hausdorff dimension is larger than the topological one and is fractional. There are natural fractals-those that can be found in nature (clouds, trees, coastlines) and algebraic, which are built on the basis of algebraic formulas.

The classical examples of algebraic fractals are the Mandelbrot set and the Jules set. The basis of the construction of these sets is the multiple calculation of the nonlinear function based on the given initial data.

The use of intermediate cover image while embedding watermarks in images allows you to preserve the high quality of the hidden data. This is because the hidden data is not embedded in the container itself, but in the intermediate cover image [3], [4].

The use of fractal intermediate cover images increases the secrecy of the stegosystem, as an attacker will not be able to generate an intermediate cover image without knowledge of the parameters used in the generation of fractal images [5], [6].

The aim of this work is to analyze the possibility of using fractal images as intermediate cover image when embedding watermark into still images.

II. FRACTAL IMAGE GENERATION

Fractal image is generated using the escape time algorithm [7]. The escape time algorithm is a method of displaying certain aspects of the system behavior during iteration.

The escape time algorithm determines the location of each pixel on the map of possible system conditions and assigns it a color. Taking the pixel location as the initial conditions, the algorithm iterates the system until one of the two possibilities occurs. If it is obvious that the iterations do not lead to the attractor whose boundary we want to illuminate, then the initial conditions do not belong to this attractor, and the corresponding pixel is assigned a color based on how many iterations it took to determine the non-membership of the attractor [8], [9].

“Escape time” in integrated systems is not really time, but the number of iterations. There is a possibility that the initial conditions will come to the attractor as the iterations continue. In this case, we could continue the iteration forever, trying to determine if the point belongs to the attractor. To avoid this, the escape time algorithm contains a predefined limit on the number of iterations. When this limit, called the iteration depth, is reached, the algorithm drops and colors the pixel.

The Julia set is constructed on the basis of the initial point on the complex plane $c = x+iy$. A quadratic complex

ISSN 2305-7254

polynomial is used for the construction are $f(z) = z^2+c$ [10].

The algorithm is based on complex maps that map one complex number $z_n = x_n + iy_n$ to another complex number $z_{n+1} = x_{n+1} + iy_{n+1}$ by iterative rule.

$z_{n+1} = f(z_n)$, where f(z) is a some non-linear function of z and n is the number of the iteration.

For correct generation of fractal images, both the receiving and sending side need to agree in advance on the parameters used. We will use the following parameters: cx, cy – coordinates of the center of the rectangle, l – width of the sides of the rectangle, maxstep – number of steps, m – size of the fractal image and the complex number c that affects the generation of the image.

Step by step algorithm can be described as follows [11]:

1) Enter initial values. Before forming a fractal set it is necessary to set some initial conditions, namely:

- maximum number of iterations k;
- size of the rectangle lxl, within which the set will be constructed;
- the coordinates of the center of the rectangle cx, cy;
- the size of the image that will contain the fractal mxm.

2) Generation of rectangle of size lxl;

3) By pixel division of the formed rectangle;

4) Calculation of radius R;

5) Determination of belonging of points to the set.

Fractal image generation algorithm in Matlab is described by the follow formulas:

$$x = linspace(cx - l, cx + l, m)$$

$$y = linspace(cy - l, cy + l, m)$$

where *linspace(x1,x2,n)* is special Matlab function, which generate linearly spaced vector of size lxn in interval from x1 to x2.

Function [X,Y] = *meshgrid(x,y)* returns 2-D grid coordinates based on the coordinates contained in vectors x and y. X is a matrix where each row is a copy of x, and Y is a matrix where each column is a copy of y. The grid represented by the coordinates X and Y has length(y) rows and length(x) columns.

After this complex number Z is defined using variables X and Y:

$$Z = X + i * Y$$

And finally in cycle from 1 to value of variable maxstep follow transformations are performed:

$$Z = Z^2 + c$$

$$W = e^{-|Z|}$$

As a result, a two-dimensional array W of the values of the obtained fractal image is formed.

In the case of black and white fractals, all points related to the set are painted white. In the case of color images, the color was chosen so that the absolutely red color (255) has points that do not go to infinity with the maximum number of iterations maxstep (belonging to the Julia set). For the green and blue channels, an inverse relationship was used.

Table I shows the parameter values of the generated fractals.

TABLE I. PARAMETERS OF GENERATED FRACTALS

Start point c	Size of image mxm	Size of rectangle lxl	Number of iterations
-0.74543+0.11301i	150x150	1.5	300
-0.74543+0.11301i	150x150	1.5	3000
-0.74543+0.11301i	150x150	0.5	3000
-0.74543+0.11301i	150x150	0.0005	3000
-0.74543+0.11301i	150x150	0.5	300
-0.74543+0.11301i	150x150	0.0005	300

The results of fractals generation are shown in Fig. 1.

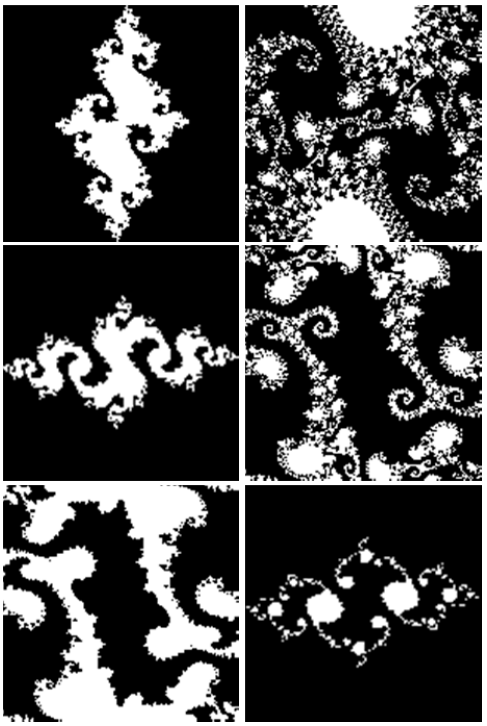


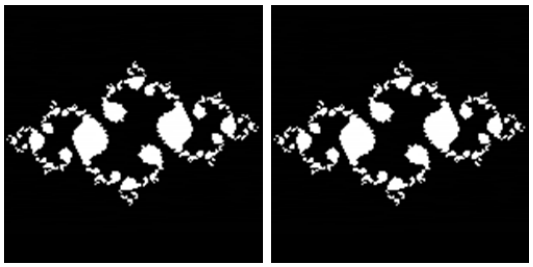
Fig.1. Generated fractal images

The image in BMP format containing a qr code of 50x50 pixels size, shown in Fig. 2, was used as a watermark.

Fig.2. A watermark

A fractal image with the following parameters was chosen as the intermediate cover image: c = -0.76643+0.16471*i, l = 1.5, k=30 and size 150x150 pixels (Fig. 3a). The formed set contains only two colors: black and white, all the points that got into the set at any number of iterations were colored white.

Next, the selected watermark was embedded in the generated intermediate cover image by replacing the least significant bit (LSB) [12]. While embedding, the fractal is converted to a one-dimensional binary array and each 8th bit of the array (i.e. each least significant bit of the next byte of the fractal) is replaced by a watermark bit. The results of embedding are shown in Fig. 3b.



a) b)

Fig. 3. a) generated fractal intermediate cover image; b) fractal intermediate cover image with embedded watermark

III. DARMSTAEDTER-DELAIGLE METHOD

The Darmstadter-Delage algorithm is block based and embeds the watermark by modifying the blue channel. Before being embedded, the watermark information is converted into bitstream. Each bit of the watermark is embedded in a single block. When splitting the image-container array into blocks, the 8x8 segment size is used.

The embedding of the bits is performed thanks to the following steps [13]:

1) Subdivision of the image in 8x8 blocks

2) Classification of the pixels of one block in zones of homogeneous luminance values;

3) Subdivision of each zone in categories defined by a specific grid;

4) Embedding of the bit through the relationship between categories mean values in each zone, following a given embedding rule.

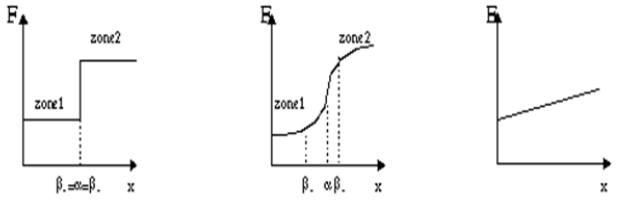
Before being embedded, the copyright information is converted into a bitstream. Each bit is spread over one block. In the current implementation, the block size is 8x8. When splitting a container image array into blocks, the 8x8 segment size is used. This choice is justified by the fact that blocks of JPEG compression have the same size. Accordingly, the compression effect will be reflected independently on each embedded block.

Next is the classification, which aims to divide the pixels of the block into groups of relatively homogeneous luminance. There are three types of contrast: hard contrast (Fig. 4a), progressive contrast (Fig. 4b) and noise contrast (Fig. 4c).

a) b) c)

Fig. 4. Contrasts of block: a) hard contrast b) progressive contrast c) noise contrast

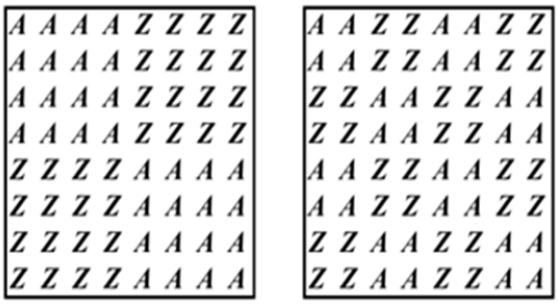
Then luminance values of the block are represented by an increasing function F(i). The slope of F(x), S(x), determines the block contrast type.



a) b) c)

Fig. 5. Contrast functions of block: a) hard contrast b) progressive contrast c) noise contrast

After classification, the zone is divided into two categories (A and Z). Masks are applied to sort the pixels into these categories. Examples of masks are shown in Fig. 6. The mask creation algorithm must be kept secret, and it is recommended to use more complex combinations and individual masks for each embedded bit. The purpose of masks is to ensure the secrecy of the embedding.



a) b)

Fig.6. Example of grids: a) 4x4 b) 2x2

From the first two steps, 4 different groups of pixels in the blocks are defined, depending on the zone (1 or 2) and the category (A or B). 6 values can be computed from this subdivision:

- the 4 means λ_{1A}, λ_{1Z}, λ_{2A} and λ_{2Z} of groups containing respectively n_{1A}, n_{1Z}, n_{2A}, and n_{2Z} pixels;
- the zones mean values Λ_1 and Λ_2.

The means from same zones are combined together. The bit is thus embedded two times through both zones. This increases the robustness and permits to embed the bit without altering the block excessively.

The embedding of a bit b in the block is performed through the relationship between categories luminance mean values. The embedding rule is given by:

$$\text{if b =0} \begin{cases} \lambda^*_{1Z} - \lambda^*_{1A} = E; \\ \lambda^*_{2Z} - \lambda^*_{2A} = E; \end{cases}$$

$$\text{if b = 1} \begin{cases} \lambda^*_{1A} - \lambda^*_{1Z} = E; \\ \lambda^*_{2A} - \lambda^*_{2Z} = E; \end{cases}$$

where λ^*_{1A}, λ^*_{1Z}, λ^*_{2A} и λ^*_{2Z} are the mean values required by b and E is the embedding level, i.e. the required difference between the mean values.

In order to make the embedding result as invisible as possible, low frequencies must be kept. Preservation of average values of intensity of each zone is provided by accomplishment of the following conditions:

$$\frac{n_{1A} \cdot \lambda^*_{1A} + n_{1Z} \cdot \lambda^*_{1Z}}{n_{1A} + n_{1Z}} = \Lambda_1;$$
$$\frac{n_{2A} \cdot \lambda^*_{2A} + n_{2Z} \cdot \lambda^*_{2Z}}{n_{2A} + n_{2Z}} = \Lambda_2;$$

The extraction of embedded information from a stego image requires the knowledge of the size of the blocks and the configuration of the masks used in the embedding. During the extraction process, the first three stages are identical to the corresponding stages of the embedding algorithm. The fourth stage is carried out as follows.

Let Σ_1 and Σ_2 be the values obtained by comparing the categories average values:

$$\Sigma_1 = \lambda_{1A} - \lambda_{1B};$$
$$\Sigma_2 = \lambda_{2A} - \lambda_{2B};$$

The sign of the calculated values allow to decode hidden bit. Moreover, the absolute values of $\Sigma_1$and Σ_2 give a confidence level to this determination. Three cases are possible:

1) $\Sigma_1 \cdot \Sigma_2 > 0$. In this case b*=1 if Σ_1>0 and b*=0 if Σ_1<0.

2) $\Sigma_1 \cdot \Sigma_2 < 0$. In this case, the additional parameter is calculated:

$$\Sigma' = \Sigma_1(n_{1A} + n_{1Z}) + \Sigma_2(n_{2A} + n_{2Z}).$$

Here b*=1 if Σ'>0 and b*=0 if Σ'<0.

3) $\Sigma_1 \cdot \Sigma_2 \approx 0$. Be $\Sigma' = \max(|\Sigma_1|, |\Sigma_2|)$, b*=1 if Σ' >0 and b*=0 if Σ'<0.

The global confidence level of the bit is the integrated value of each occurrence of this bit in the image. The bit is determined afterwards, according to the global confidence level.

The algorithm has a number of advantages, such as low computational complexity, high level of invisibility, and the possibility of blind (without using the original container) data extraction. However, it is not resistant to JPEG compression. When you extract a watermark from a JPEG image, noise occurs that distorts the label and makes it impossible to read data from the QR code (Fig. 7).

a) b)

Fig.7. a) original watermark b) a watermark extracted from JPEG stego image

IV. Experimental results

The results of the algorithm are shown in Fig. 8-11.

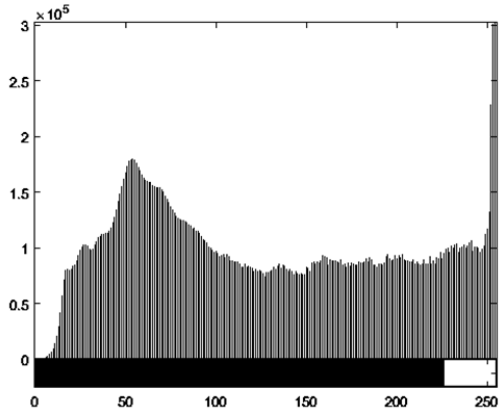
Fig. 8a shows the original cover image of size 6400x4096 pixels in JPEG format, the stego image with embedded secret intermediate cover image is shown in Fig. 8b.

a)

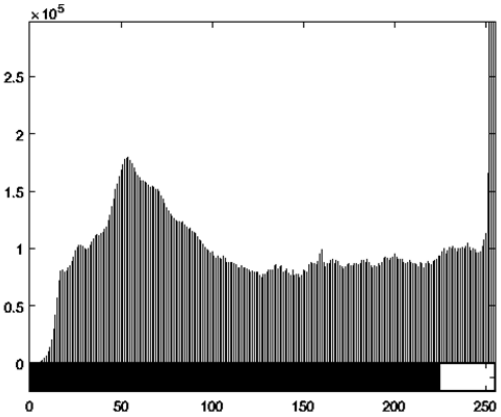
b)

Fig 8. a) original cover image b) stego image with embedded secret intermediate cover image

Fig. 9 shows histograms of the intensity distribution of the pixels of the original cover mage and the resulting stego image.



a)

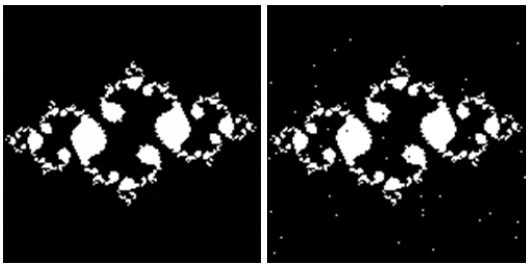


b)

Fig 9. Histograms a) original cover image b) stego image with embedded secret intermediate cover image

Visual analysis of the images from Fig. 8 and comparison of the histograms allows us to conclude that the use has achieved a high level of invisibility. It is impossible to determine the presence of hidden data without knowledge.

Fig. 10 shows the results of retrieving the private intermediate cover image from a stego image. Fig. 10a shows the original fractal image, Fig. 10b - the image after extraction.



a) b)

Fig.10. a) original fractal intermediate cover image b) fractal image extracted from stego image

Fig. 11 shows the results of extracting a watermark in the form of a QR code from an intermediate cover image. Fig. 11a contains the original watermark, Fig. 11b - a watermark extracted from the intermediate cover image. When the secret information is extracted after the intermediate cover image is received, a fractal is generated on the receiving side using pre-selected parameters. The watermark is restored by subtracting the resulting intermediate cover image and the generated fractal [14].

a) b)

Fig.11. a) original watermark b) a watermark extracted from secret intermediate cover image

As a result of the algorithm, a watermark was extracted. Despite the fact that you can observe a slight distortion of pixels in the extracted intermediate cover image, the resulting watermark is completely identical to the original. The quality of the extracted data is much higher compared to the original method without use of an intermediate cover image.

V. Evaluation of the algorithm quality

The visual distortion quality of the fractal container was evaluated based on the following metrics: normalized mean square error (NMSE) and peak signal-to-noise ratio (PSNR):

$$NMSE = \sum_{x,y}(C_{x,y} - S_{x,y})^2 / \sum_{x,y}(C_{x,y})^2$$

$$PSNR = XY \cdot \max_{x,y}(C_{x,y})^2 / \sum_{x,y}(C_{x,y} - S_{x,y})^2$$

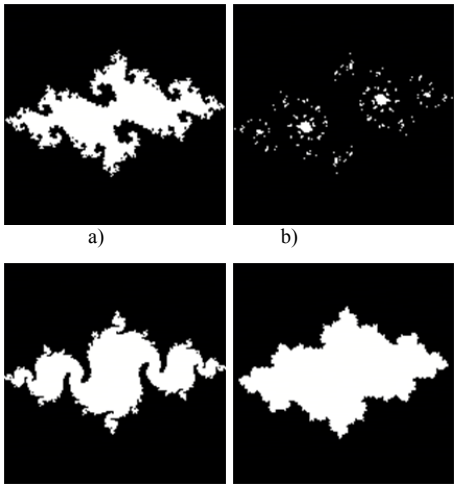
where $C_{x,y}$ is the pixel of original cover image with coordinates (x,y), and $S_{x,y}$ is the corresponding pixel of stego image.

To evaluate the developed system, metrics were calculated at different sizes of the embedded secret intermediate cover image and watermark. The results are shown in Table II.

TABLE II. NMSE AND PSNR FOR DIFFERENT SIZES OF EMBEDDED DATA

Size of watermark, pixels	Size of intermediate cover image, pixels	NMSE	PSNR
30x30	85x85	0,0021	31,19
35x35	100x100	0,0031	29,42
40x40	115x115	0,0056	26,77
45x45	130x130	0,006	26,53
50x50	150x150	0,0067	26,35

To assess the secrecy of the developed system, a situation was simulated, when the attacker took possession of an intermediate cover image with an embedded watermark [15]. In addition, we assume that the attacker also knows some parameters of the generated fractal, namely the size of the rectangle l, the maximum number of iterations k and the size of the image m. The only unknown parameter is the starting point c. As an example, four fractals with different values of parameter c were generated, as shown in Fig. 12, and a watermark was extracted using the obtained images and an intermediate cover image containing watermark, shown in Fig. 3b. The results of the extraction of watermark are shown in Fig. 13.



a) b) c) d)

Fig.12. Fractal images with different value of c a) -0.73949 + 0.16498*i; b) -0.74549 + 0.37841*i; c) -0.80939 + 0.12388*i; d) -0.63949 +0.19098*i.

a) b) c) d)

Fig.13. Watermarks are obtained during the extraction with the use of fractals, presented in Fig. 7

As can be seen from the figures, all the images obtained as a result of the experiment are significantly different from the original watermark, so that the extraction of information from the QR code becomes impossible. This allows us to conclude that to obtain information it is necessary to know with high accuracy all the parameters of the fractal intermediate cover image. The developed method allows reducing the probability of substitution or theft of classified information in such steganosystems to almost zero.

VI. CONCLUSION

As a result of the research it is shown that the use of fractal intermediate cover images improves the quality of the embedding and extraction of secret information in steganographic systems. Thanks to the intermediate cover image, when extracting secret information, it was possible to obtain a high quality watermark without visible visual distortions and the ability to read information from the QR code.

The main advantage of the developed technique is the high secrecy of embedding. Without the exact value of the complex number c, which is agreed in advance between the sender and the recipient, as well as a number of other parameters of the algebraic fractal, the attacker will not be able to generate an identical fractal image, which makes the proposed method resistant to attacks.

REFERENCES

[1] O.I. Sheluhin., D.I. Magomedova, “Analysis of methods for calculating the fractal dimension of color and gray-scale images”, *H&ES Research,* vol. 9, Dec.2017, pp. 6-16.

[2] A. Garg, “Article: An Improved Algorithm of Fractal Image Compression”, *Int. J. Comput. Appl*., vol. 34, no. 2, 2011, pp. 17–21.

[3] Y. Wu, J. Noonan, “Image Steganography Scheme using Chaos and Fractals with the Wavelet Transform”, *International Journal of Innovation, Management and Technology*, vol. 3, no. 3, June 2012, pp.285-289.

[4] K. Thamizhchelvy, G. Geetha. “Design of digital signature algorithm by fractals and chaos theory”, *International Journal of Computer Applications,* vol.37, no.5, Jan.2012, pp.50 – 57.

[5] O.I. Sheluhin, S.D. Kanaev, “Hiding watermarks in color images using algebraic fractals by 2D wavelet transform methods”, *T-Comm,* vol. 12, no.6., 2018, pp. 46-50.

[6] A.S. Nori, A M. Al-Qassab, “Steganographic technique using fractal image”, *International Journal of Information Technology and Business Management*, March2014, pp. 52-59.

[7] M. Rani, “Cubic Superior Julia Sets”, *Proceedings of the European Computing Conference*, 2011, pp. 80-84.

[8] T. A. Abbas, H. K. Hamza, “Steganography Using Fractal Images Technique”, *IOSR Journal of Engineering (IOSRJEN),* vol. 04, Feb. 2014, pp. 52-61.

[9] K. Thamizhchelvy, G. Geetha, “Data Hiding Technique with Fractal Image Generation Method using Chaos Theory and Watermarking”, *Indian Journal of Science and Technology*, vol 7(9), Sep.2014, pp. 1271–1278.

[10] Y. Xing, J. Tan, P. Hong, "Quaternion Julia Fractals", *IEEE Computer Society, The 9th International Conference for Young Computer Scientists*, 2008, pp. 797-802.

[11] K. Kiani, M. Arian, V. Soleimani, “Image Authentication using Fractal Water-marking and Chaos Theory”, *4th International Conference on Signal Processing and Communication Systems (ICSPCS),*Dec.2010, pp. 13–15.

[12] O.I. Sheluhin and S.D.Kanaev, *Steganography. Algorithms and software implementation. Under. editorship of Professor. Sheluhin O.I.* Moscow: Hot line – Telecom, 2017.

[13] V. Darmstaedter, J.-F. Delaigle, J.J. Quisquater, B. Macq, “Low cost spatial watermarking”, *In Computers and Graphics*, vol. 4, Aug. 1998, pp. 417-423.

[14] V. Hardikkumar Desai, A. Desai Apurva, “Image Steganography Using Mandelbrot Fractal,” *International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR)*, vol. 4, no. 2, 2014, pp. 71–80.

[15] M. Abbas, *'Digital Image Watermarking Robustness: A Comparative Study.* Netherlands: Faculty of Electrical Engineering, Mathematics and Computer Science, 2009.