INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2017, VOL. 63, NO. 4, PP. 417–422
Manuscript received June 29, 2017; revised October 23, 2017. DOI: 10.1515/eletel-2017-0061

Least Significant Bit Steganography using Hitzl-Zele Chaotic Map

Krasimir Kordov and Borislav Stoyanov

***Abstract*—We propose a novel least significant bit steganography algorithm based on a Hitzl-Zele chaotic function. On the first step a pseudorandom generator is constructed for chaotic pixel selection for hiding the secret message. Exact study has been provided on the novel scheme using visual inspection, peak signal-to-noise ratio, and histogram analysis. The experimental data show excellent performance of the novel stego technique.**

***Keywords*—Steganography, Steganalysis, Chaotic functions, Pseudorandom number generator.**

I. Introduction

STEGANOGRAPHY is art and science of hiding secret information (message) in plain sight without being noticed within an innocent cover data (container) so that it can be securely broadcasted over a communications channel. First used containers usually were hand written text, images ets. but with technology progress one of the most used containers these days are raster graphic files.

In recent years, with rapid development of information and communication technologies, web-based services (tools) are becoming increasingly favourite. Steganography, as a part of science of securely data transmission [6], [20], is an art of inconspicuous message sending. There is a need for new stego algorithms, that provide the necessary security and privacy in data communication [29]. Using chaotic maps in steganography algorithms become more popular, because of the resistance of increasing stego attacks.

In two fundamental papers [15], [16] F. Pichler and J. Scharinger proposed the chaotic Baker map to securing digital communications. J. Fridrich in [9] extended their work by adapting nonlinear two-dimensional maps on a torus and on a square, with aims to achieve better encryption.

Least significant bit (LSB) method based on Henon map is presented in [14]. In [17], novel least significant bit method based on 1D logistic map is proposed. In [12], a novel steganographic algorithm using Baker and Logistic maps is designed. Two-level steganographic methods based on chaotic maps are presented in [18] and in [13]. Novel LSB hiding algorithm, which uses chaotic rotation equations, is proposed in [26].

In Section II we propose a novel pseudorandom bit generator based on Hitzl-Zele chaotic function. In Section III we present a novel LSB image steganography algorithm, and extended performance analysis is given. Finally, the last section concludes the paper.

This work is supported by the Scientic research fund of Konstantin Preslavski University of Shumen under the grant No. RD-08-124/06.02.2017.

Krasimir Kordov and Borislav Stoyanov are with the Department of Computer Informatics, Konstantin Preslavsky University of Shumen, 9712 Shumen, Bulgaria, e-mails: krasimir.kordov@shu.bg, borislav.stoyanov@shu.bg.

II. Pseudorandom bit generator based on Hitzl-Zele chaotic map

Pseudorandom generators are basic primitives used in cryptography algorithms but in our case we apply the random properties of chaotic pseudorandom bit generator to steganography algorithm. Pseudorandom generators are software realized methods for extracting sequence of random values.

A. Proposed Pseudorandom Bit Generation Algorithm

The Hitzl-Zele function [10], [19], is a three-dimensional discrete-time dynamical system given by:

$$\begin{aligned} x(k+1) &= 1 + y(k) - z(k) \cdot x^2(k) \\ y(k+1) &= a \cdot x(k) \\ z(k+1) &= b \cdot x^2(k) + z(k) - 0.5, \end{aligned} \tag{1}$$

with bifurcation parameters $a = 0.25$ and $b = 0.87$ for chaotic behaviour. Figure 1(a), Figure 1(b) and Figure 1(c) represents graphic plotting points of x, y and z dimensions.

We propose a new pseudorandom bit generation algorithm with the following steps:

Step 1: The initial values $x(0)$, $y(0)$, and $z(0)$ from Eq. (1), and a bit stream limit L are determined.

Step 2: The Hitzl-Zele chaotic map is iterated initially for L times.

Step 3: The iteration of the Eq. (1) continues, and as a result, three real fractions $x(i)$, $y(i)$, and $z(i)$, are calculated.

Step 4: The first two numbers $x(i)$ and $y(i)$ are post-processed as follows:

$$\begin{aligned} h_0 &= abs(mod(integer(x(i) \times 10000000), 2) \\ h_1 &= abs(mod(integer(y(i) \times 10000000), 2), \end{aligned} \tag{2}$$

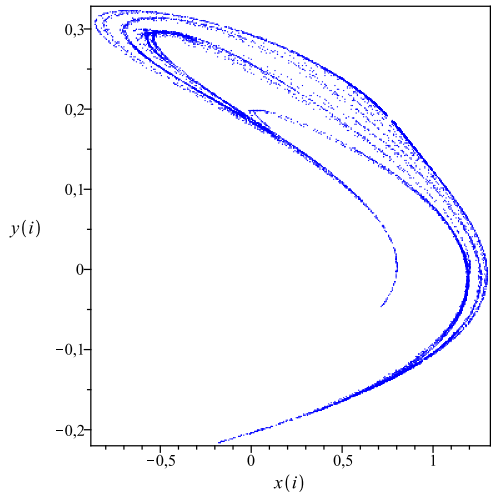
where $abs(x)$ returns the absolute value of x, $integer(x)$ returns the the integer part of x, truncating the value at the decimal point, $mod(x, y)$ returns the reminder after division.

Step 5: Perform XOR operation between h_0 and h_1 to get a single output bit.

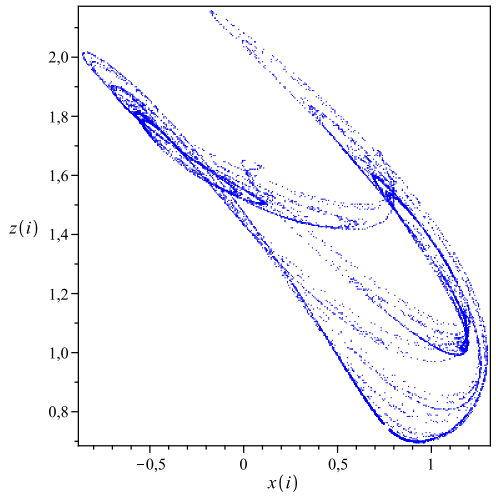
Step 6: Return to Step 3 until the bit stream limit is reached.

The presented pseudorandom bit generator is implemented in C++, using the following initial values: $x(0) = 0.9134$, $y(0) = 0.6324$, $z(0) = 0.0975$, and $L = 550$.

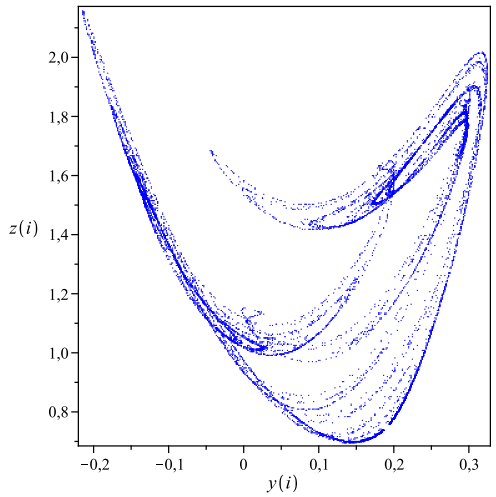
Similar chaos based PRG can be found in [3], [5], [7], [8], [22], [23], [24], [25], [27].



(a) 2D plot of the Hitzl-Zele chaotic function using x and y values.



(b) 2D plot of the Hitzl-Zele chaotic function using x and z values.



(c) 2D plot of the Hitzl-Zele chaotic function using y and z values.

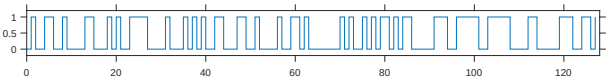
Fig. 1. 2D plot of the Hitzl-Zele chaotic function.

B. Key space

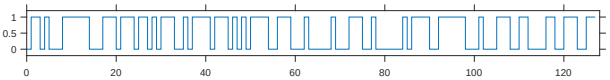
The key space is configured by the four numbers $x(i)$, $y(i)$, and $z(i)$, and L. With number of about 16 decimal digits precision [30] the key space is more than 2^{172}. This is large enough against exhaustive attack methods [1].

C. Key sensitivity

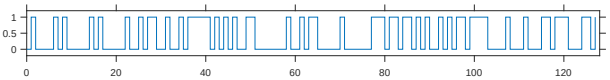
Testing key sensitivity requires comparison of result output binary sequences using very similar secret keys. In order to do the experiments we changed a single digit of variables from key space of the proposed pseudorandom generator. The secret key 1 uses the initial values described in Section II, the secret key 2 is formed by changing $x(0)$ to 0.9135, for secret key 3 $x(0)$ is changed to 0.9324, for secret key 4 $y(0)$ is changed to 0.6325, for secret key 5 $y(0)$ is changed to 0.6334, and for the last two secret keys 6 and 7, $z(0)$ is changed to 0.0976 and 0.0985 respectively. The next figures visually show the result sequences calculated using different but similar secret keys.



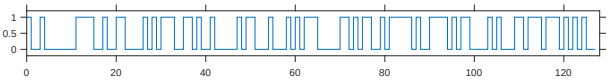
(a) Plot of binary sequence using secret key 1.



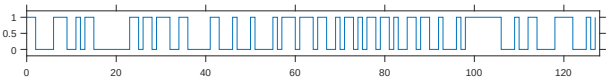
(b) Plot of binary sequence using secret key 2.



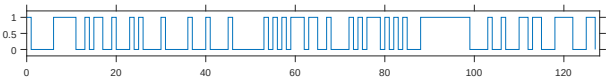
(c) Plot of binary sequence using secret key 3.



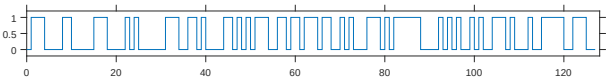
(d) Plot of binary sequence using secret key 4.



(e) Plot of binary sequence using secret key 5.



(f) Plot of binary sequence using secret key 6.



(g) Plot of binary sequence using secret key 7.

Fig. 2. Plot of binary sequences with similar keys.

All the figures are very different, demonstrating that even if a single digit is changed from the initial values of $x(0)$, $y(0)$ or $z(0)$, the end result is entirely changed. The results indicates high level of security provided by the proposed pseudorandom generator as far as the secret key is concerned.

D. Statistical tests

Three software test programs are used in order to measure the behaviour of the output binary streams.

The NIST test suite [21] includes 15 statistical tests: Frequency (monobit), Block-frequency, Cumulative sums (forward and reverse), Runs, Longest run of ones, Rank, Fast Fourier Transform (spectral), Non-overlapping templates, Overlapping templates, Maurers "Universal Statistical", Approximate entropy, Random excursion, Random-excursion variant, Serial, and Linear complexity. The proposed pseudorandom bit generator passed successfully NIST test suite, Table I.

TABLE I
NIST TEST SUITE RESULTS.

NIST statistical test	**Proposed Algorithm**	
	P-value	**Pass rate**
Frequency (monobit)	0.709558	994/1000
Block-frequency	0.473064	991/1000
Cumulative sums (Forward)	0.753844	993/1000
Cumulative sums (Reverse)	0.624627	994/1000
Runs	0.844641	989/1000
Longest run of Ones	0.526105	994/1000
Rank	0.433590	990/1000
FFT	0.723804	990/1000
Non-overlapping templates	0.477861	990/1000
Overlapping templates	0.504219	986/1000
Universal	0.094285	987/1000
Approximate entropy	0.997698	988/1000
Random-excursions	0.567112	634/641
Random-excursions Variant	0.509168	634/641
Serial 1	0.000976	988/1000
Serial 2	0.128874	987/1000
Linear complexity	0.370262	988/1000
The minimum pass rate for each statistical test with the exception of the random excursion (variant) test is approximately = 980 for a sample size = 1000 binary sequences. The minimum pass rate for the random excursion (variant) test is approximately 627 for a sample size 641 binary sequences.		

The DIEHARD package [11] is set of 19 statistical tests: Birthday spacings, Overlapping 5-permutations, Binary rank (31 x 31), Binary rank (32 x 32), Binary rank (6 x 8), Bitstream, Overlapping-Pairs-Sparse-Occupancy, Overlapping-Quadruples-Sparse-Occupancy, DNA, Stream count-the-ones, Byte-count-the-ones, Parking lot, Minimum distance, 3D spheres, Squeeze, Overlapping sums, Runs (up and down), and Craps. The tests return $P-values$, which should be uniform in [0,1), if the input file contains pseudorandom numbers. The $P-values$ are obtained by $p = F(y)$, where F is the assumed distribution of the sample random variable y, often the normal distribution. The proposed pseudorandom bit generator passed successfully DIEHARD tests, Table II.

The ENT package [28] includes 6 tests to pseudorandom sequences: Entropy, Optimum compression, χ^2 distribution, Arithmetic Mean value, Monte Carlo Value for π, and Serial Correlation Coefficient. The sequences of bytes are stored in files. The suite outputs the results of those tests. We tested output sequences of 125000000 bytes of the novel pseudorandom bit generation algorithm. The novel pseudorandom bit generation algorithm passed successfully ENT test, Table III.

From key space evaluation and the positive test results, we can conclude that the novel pseudorandom bit generation algorithm has good statistical properties and provide high level of security and randomness.

TABLE II
DIEHARD STATISTICAL TEST RESULTS.

DIEHARD statistical test	**Proposed Algorithm** ***P-value***
Birthday spacings	0.393909
Overlapping 5-permutation	0.518083
Binary rank (31 x 31)	0.346194
Binary rank (32 x 32)	0.340976
Binary rank (6 x 8)	0.561724
Bitstream	0.499964
OPSO	0.669539
OQSO	0.578114
DNA	0.635990
Stream count-the-ones	0.047321
Byte count-the-ones	0.452763
Parking lot	0.612213
Minimum distance	0.486939
3D spheres	0.638137
Squeeze	0.057748
Overlapping sums	0.533742
Runs up	0.571370
Runs down	0.236529
Craps	0.357239

TABLE III
ENT STATISTICAL TEST RESULTS.

ENT statistical test	**Proposed Algorithm results**
Entropy	7.999998 bits per byte
Optimum compression	OC would reduce the size of this 125000000 byte file by 0 %.
χ^2 distribution	For 125000000 samples is 415.50, and randomly would exceed this value 0.01 % of the time.
Arithmetic mean value	127.5004 (127.5 = random)
Monte Carlo π estim.	3.141717554 (error 0.00 %)
Serial correl. coeff.	0.000012 (totally uncorrelated = 0.0)

III. Least Significant Bit Steganography using Hitzl-Zele Chaotic Map

A. Proposed Least Significant Bit Steganography Algorithm

Here we design a novel LSB image steganography scheme by using the pseudorandom bit generation algorithm based on the Hitzl-Zele chaotic function, Section II.

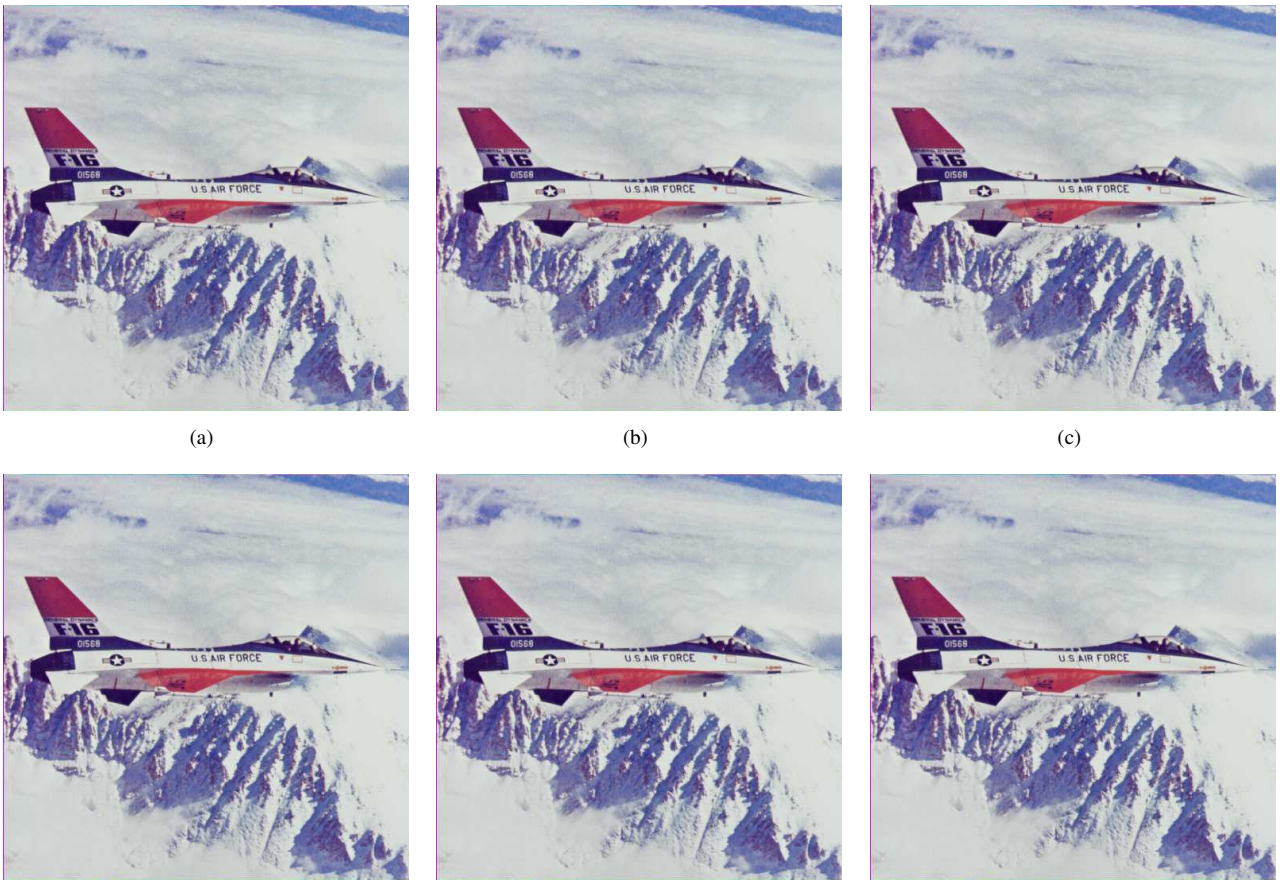
We consider plain and stego images of $n \times m$ size. The bytes of the BMP pixel's grid are randomly passed based on Hitzl-Zele pseudorandom bit generation. The header information bits are directly transferred into the stego image.

The novel steganography algorithm consists of the following steps:

Step 1: Symbol for the end of the plain text is added.

Step 2: Convert plain text to binary sequence using ASCII table.

Step 3: The pseudorandom generator based on Hitzl-Zele function is iterated forty eight times to produce 24 bits for i-value and 24 bits for j-value. These bits are converted modulo n and modulo m, respectively. Integer values i and j are produced.



(a) (b) (c)

(d) (e) (f)

Fig. 3. Visual comparison of the 4.2.05 Airplane (F-16) plain image and the stego images: (**a**) original image, (**b**) stego image with 100 chars, (**c**) stego image with 200 chars, (**d**) stego image with 300 chars, (**e**) stego image with 400 chars, and (**f**) stego image with 500 chars.

TABLE IV
MSE AND PSNR FOR IMAGES WITH 100 CHARS (800 BITS), 200 CHARS (1600 BITS), 300 CHARS (2400 BITS), 400 CHARS (3200 BITS), AND 500 CHARS (4000 BITS) EMBEDDED.

Images	100 chars		200 chars		300 chars		400 chars		500 chars	
	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR
4.1.01	0.0021	74.9390	0.0041	71.9873	0.0061	70.2714	0.0079	69.1384	0.0101	68.0761
4.1.02	0.0021	74.9923	0.0041	71.9927	0.0061	70.2569	0.0082	68.9986	0.0101	68.0717
4.1.03	0.0021	74.9390	0.0041	72.0089	0.0061	70.2569	0.0082	69.0175	0.0102	68.0241
4.1.04	0.0021	74.9816	0.0041	72.0413	0.0062	70.2175	0.0081	69.0229	0.0102	68.0500
4.1.05	0.0020	75.0680	0.0040	72.1292	0.0061	70.2569	0.0082	68.9770	0.0103	67.9876
4.1.06	0.0020	75.0680	0.0041	71.9981	0.0061	70.2569	0.0082	68.9959	0.0103	68.0047
4.1.07	0.0020	75.2009	0.0040	72.0905	0.0059	70.3887	0.0081	69.0610	0.0101	68.0739
4.1.08	0.0020	75.1008	0.0041	72.0089	0.0061	70.2642	0.0082	69.0094	0.0102	68.0565
4.2.01	0.0005	80.7933	0.0011	77.8654	0.0016	76.2132	0.0021	74.9629	0.0026	73.9805
4.2.02	0.0005	80.9490	0.0011	77.7983	0.0016	76.1010	0.0021	74.9126	0.0026	73.9403
4.2.03	0.0006	80.7225	0.0011	77.8086	0.0016	76.1010	0.0021	74.9047	0.0026	73.9235
4.2.04	0.0005	81.0129	0.0010	78.1002	0.0015	76.3247	0.0020	75.0435	0.0025	74.0858
4.2.05	0.0005	81.0668	0.0010	78.0026	0.0015	76.3466	0.0020	75.0844	0.0025	74.1405
4.2.06	0.0006	80.6826	0.0010	77.9334	0.0016	76.2061	0.0021	74.9073	0.0026	73.9996
4.2.07	0.0005	80.9702	0.0010	78.0619	0.0015	76.2956	0.0020	75.0219	0.0026	74.0125
house	0.0005	81.2215	0.0010	78.1609	0.0015	76.3797	0.0020	75.0734	0.0025	74.0728

Step 4: Repeat Step 3 until unused pixel position (i, j) is detected.

Step 5: Embed three bits from the input sequence into the last bit of the RED, GREEN and BLUE values of the selected pixel position (i, j).

Step 6: Repeat Steps 3–5 until stego image is produced.

Decoding algorithm consists of the following steps:

Step 1: The pseudorandom generator based on Hitzl-Zele function is iterated forty eight times to produce 24 bits for i-value and 24 bits for j-value. These bits are converted modulo n and modulo m, respectively. Integer values i and j are produced.

TABLE V
PSNR (DB) OF THE PROPOSED ALGORITHM AND SOME OTHER ALGORITHMS.

Images	Ref. [2]	Ref. [4]	Ref. [17]	Ref. [18]	Proposed
4.2.03	51.11	44.26	44.54	44.16	73.92
4.2.04	51.12	41.62	44.53	44.19	74.09
4.2.05	51.14	44.24	44.42	-	74.14

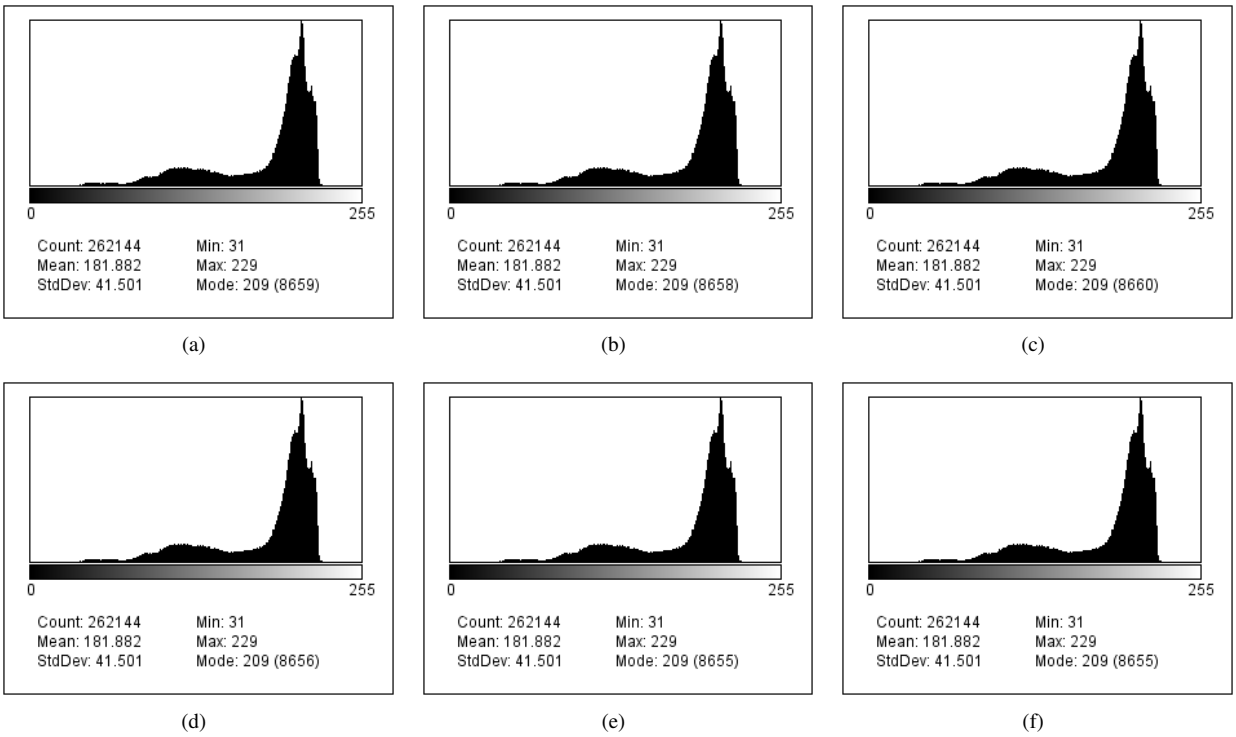


Fig. 4. Hystogram analysis of the 4.2.05 Airplane (F-16) plain image and the stego images: (**a**) original image, (**b**) stego image with 100 chars, (**c**) stego image with 200 chars, (**d**) stego image with 300 chars, (**e**) stego image with 400 chars, and (**f**) stego image with 500 chars.

Step 2: Repeat Step 1 until unused pixel position (i, j) is detected.
Step 3: Retrieve three bits to the output binary sequence of the last bit of the RED, GREEN and BLUE values of the selected pixel position (i, j).
Step 4: Every 8 bits are converted into char symbol and added to extracted text.
Step 5: Repeat Steps 1–4 until the end symbol is detected.

The designed LSB algorithm is implemented in C++ programming language and the color images are selected from USC-SIPI image database (sipi.usc.edu/database/).

B. Visual Inspection Analysis

As an example, Figure 3 shows the image 4.2.05 Airplane (F16), Figure 3(a), and its stego images, Figure 3(b)–3(f). The visual inspection does not find differences between the plain image and the stego images. The images are identical optically with no indication of hidden messages.

C. Peak Signal-to-Noise Ratio Analysis

Peak Signal-to-Noise Ratio (PSNR) is the ratio between the maximum possible power of a signal and the power of noise contained in the signal. It is defined as follows:

$$PSNR = 10 \log_{10} \frac{(2^d - 1)^2}{MSE} (dB), \tag{3}$$

where d is the bit depth of the pixel and MSE is the mean square error between the plain and stego images. MSE is defined as:

$$MSE = \frac{1}{nn} \sum_{i=1}^{n} \sum_{j=1}^{n} (p[i, j] - p'[i, j])^2, \tag{4}$$

where $p[i, j]$, $p'[i, j]$ is the ith-row jth-column pixel in the plain and stego images, respectively.

In Table IV, we provide the computed values for MSE and PSNR for the proposed stego algorithm, where MSE and PSNR are calculated for images with 100 chars (800 bits), 200 chars (1600 bits), 300 chars (2400 bits), 400 chars (3200 bits), and 500 chars (4000 bits) embedded, are presented.

From the obtained results, Table IV, it is clear that the PSNR values are very high, above 68 dB, which is an indication that the new LSB chaos-based steganography algorithm has a good level of security.

In Table V, we have compare the PSNR of our embedding scheme with similar references [2], [4], [17], and [18].

Compared to other LSB steganography schemes, we can see that proposed scheme has higher PSNR values.

D. Histogram Analysis

Image histograms are graphical representation of the tonal distribution in digital images. This test compares histograms of plain and stego images. In addition, histogram analysis was performed using ImageJ (https://imagej.nih.gov/ij/) histogram analysis of the distribution of gray values in the cover and the stego image 4.2.05 Airplane (F16), see Figure 4. It is examined that the histograms of the stego images are very similar to that of the plain image with no indication of hidden messages in stego images.

IV. Future work

In the future steps, we intend to experiment with hiding messages not only in least significant bit in order to increase container capacity without any sign of steganography. We also intend to use Bent boolean functions to increase chaos behaviour and pseudorandomness of PRG. Hardware implementation of the proposed algorithm is one of our goals.

V. Conclusion

Novel least significant bit steganography algorithm based on a Hitzl-Zele chaotic function, is designed. Empirical experiments are provided for testing the security of the proposed scheme. Our stego analysis, visual inspection, peak signal-to-noise ratio, and histogram analysis, show that the novel algorithm has enough key size, high key sensitivity and excellent performance.

Acknowledgment

The authors are grateful to the anonymous referees for valuable and helpful comments, editorial board and support of INTL Journal of Electronics and Telecommunications.

References

[1] Alvarez, G., Li, S. (2006) Some Basic Cryptographic Requirements for Chaos-Based Cryptosystems, *International Journal of Bifurcation and Chaos*, **16**, 2129 - 2151.

[2] Amirtharajan, R., Rayappan, J.B.B. (15 June 2012), An intelligent chaotic embedding approach to enhance stego-image quality, *Information Sciences*, **Volume 193**, Pages 115-124

[3] M. Andrecut, Logistic Map as a Random Number Generator, *International Journal of Modern Physics B,* **12** (1999), 921 - 930.

[4] Aziz, M., Tayarani-N, M.H., and Afsar, M. (2015), A cycling chaos-based cryptic-free algorithm for image steganography, *Nonlinear Dynamics*, **80** (3), 1271-1290.

[5] Dăscălescu, A., Boriga, R.E., Diaconu, A., Study of a New Chaotic Dynamical System and Its Usage in a Novel Pseudorandom Bit Generator, *Mathematical Problems in Engineering,* **2013** (2013), Article ID 769108, 1-10.

[6] Diaconu, A.V. (2015), A Parallel Architecture Design for Ultra-Fast Image Encryption within WMSNS, *Proceedings of the Romanian Academy*, **Series A 16** (SI), 313-320.

[7] François, M., D. Defour, P. Berthomé, A Pseudo-Random Bit Generator Based on Three Chaotic Logistic Maps and IEEE 754-2008 Floating-Point Arithmetic, *Lecture Notes in Computer Science,* **8402** (2014), 229 - 247.

[8] François, M., T. Grosges, D. Barchiesi, R. Erra, Pseudo-Random Number Generator Based on Mixing of Three Chaotic Maps, *Communications in Nonlinear Science and Numerical Simulation,* **19** (2014), 887 - 895.

[9] Fridrich, J. (1998), Symmetric Ciphers Based on Two-Dimensional Chaotic Maps. *Int. J. Bifurcation Chaos*, **8**, 1259-1284.

[10] Hitzl, D.L., Zele, F. (1985), An exploration of the Hénon quadratic map, *Physica D: Nonlinear Phenomena*, **Volume 14**, Issue 3, Pages 305-326.

[11] Marsaglia, G. *DIEHARD: a Battery of Tests of Randomness*, http://www.fsu.edu/pub/diehard/.

[12] Nassar, S.S., Ayad, N.M., Kelash, H.M., El-sayed, H.S., El-Bendary, M.A.M., Abd El-Samie, F.E., Faragallah, O.S. (2016), Secure Wireless Image Communication Using LSB Steganography and Chaotic Baker, *Ciphering Wireless Personal Communications*, **91** (3), pp. 1023-1049.

[13] Nithin Kumar, S.S.V., Charan, G.S., Karthikeyan, B., Vaithiyanathan, V., Rajasekhar Reddy, M. (2016), A hybrid approach for data hiding through chaos theory and reversible integer mapping, *Advances in Intelligent Systems and Computing*, **412**, pp. 483-492.

[14] Parvees, M.Y.M., Samath, J.A., Kaspar Raj, I., Madhavan Nirmal, R. (2017), Chaos-based steganocryptic approach to protect medical images with text data of patients, *Journal of Medical Imaging and Health Informatics*, **7** (1), pp. 118-125.

[15] Pichler, F. and Scharinger, J. (1995), Ciphering by Bernoulli-shifts in finite Abelian groups, *Contributions to General Algebra*, **Vol. 9**, eds. Kaiser, H.K., Muller, W.B. and Pilz G.F., A-4040 Linz, 249-256.

[16] Pichler, F. and Scharinger, J. (1995), Finite dimensional generalized Baker dynamical systems for cryptographic applications, *Lecture Notes in Computer Science*, **Vol. 1030**, 465-476.

[17] Rajendran, S., Doraipandian, M. (2017), Chaotic Map Based Random Image Steganography Using LSB Technique, *International Journal of Network Security*, **Vol.19**, No.4, 593-598.

[18] Ranjith Kumar, R., Jayasudha, S., Pradeep, S. (2016), Efficient and secure data hiding in encrypted images, *A new approach using chaos Information Security Journal*, **25** (4-6), pp. 235-246.

[19] Saha, P., Stragatz, S.H. (1995), The Birth of Period Three, *Mathematics Magazine*, **68** (1), pp. 42-47.

[20] Saturday, J.C., Udofia, K.M., and Jimoh, A.J. (2016), Design of Dual Band Microstrip Antenna Using Reactive Loading Technique, *Mathematical and Software Engineering*, **2** (2), 114-121.

[21] Rukhin, A., J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo (2010), "A Statistical test suite for random and pseudorandom number generators for cryptographic application," *NIST Special Publication 800-22*, Revision 1a (Revised: April 2010), Lawrence E. Bassham III, http://csrc.nist.gov/rng/.

[22] Stoyanov, B., Kordov, K., Novel Image Encryption Scheme Based on Chebyshev Polynomial and Duffing Map, *The Scientific World Journal* **2014**, Article ID 283639, 2014, 1-11.

[23] Stoyanov, B., Kordov, K., A Novel Pseudorandom Bit Generator Based on Chirikov Standard Map Filtered with Shrinking Rule, *Mathematical Problems in Engineering* **2014**, Article ID 986174, 2014, 1-4.

[24] Stoyanov, B., Kordov, K., A Novel Secure Pseudo-Random Number Generation Scheme Based on Two Tinkerbell Maps, submitted, under review.

[25] Stoyanov, B., Kordov, K., Novel Zaslavsky Map Based Pseudorandom Bit Generation Scheme, Applied Mathematical Sciences, Vol. 8, 2014, no. 178, 8883-8887, http://dx.doi.org/10.12988/ams.2014.410879.

[26] Stoyanov, B.P., Zhelezov, S.K., Kordov, K.M. (2016), Least significant bit image steganography algorithm based on chaotic rotation equations, *Comptes Rendus de L'Academie Bulgare des Sciences*, **69** (7), 845-850.

[27] Sun, F., Liu, S., Cryptographic pseudo-random sequence from the spatial chaotic map, *Chaos, Solitons & Fractals,* **41** (2009), 2216 - 2219.

[28] Walker, J., *ENT: a pseudorandom number sequence test program*, http://www.fourmilab.ch/random/.

[29] Yang, C, Liu, F., Lian, S., Luo, X., Wang, D. (2012), Weighted Stego-Image Steganalysis of Messages Hidden into Each Bit Plane. Comput J., 55, 717-727.

[30] IEEE Computer Society (2008), 754-2008 - IEEE Standard for Floating-Point Arithmetic, *Revision of ANSI/IEEE Std 754-1985*, DOI: 10.1109/IEEESTD.2008.4610935.