Adaptive Remainder Modulo *m* Data Hiding

A. G. Chefranov, and G. Öz

***Abstract*—A problem of irreversible data hiding (DH), producing stego images resistant to steganalysis, is considered in spatial domain of gray-scale cover images. Stego image detection error (*DE*) is maximized when data is hidden (embedded) into noisy-like image areas where pixel values vary significantly. It is proved herein that generalization of the well-known least-significant bit (LSB) substitution to remainder modulo *m* (RM-*m*) DH method has an embedding invariant preserved after DH. A new adaptive remainder modulo *m* (ARM-*m*) method hiding data first in maximal noisy blocks by RM-*m* is proposed. ARM-*m* uses the invariant to construct a block complexity measure for adaptation. Ensemble classifiers and subtractive pixel adjacency matrix (SPAM) with 686 features were used to evaluate stego image *DE* on 886 images from UCID v.2 database. Compared to the state-of-the-art methods, ARM-4 with 2x2 blocks has *DE*=41.86% versus 24.42% of the best known method for 1 bit per pixel (bpp) embedding rate (*ER*). For *ER*=1.33 bpp, not reachable for known adaptive methods, ARM-4 and ARM-16, both with 8x8 blocks, have *DE*=27.33% and 27.91%, respectively. ARM-4 is confirmed to be better than other methods also for 2658 gray scale images. Steganalysis by two CNNs (SRNet and YeNet) revealed high resistance of ARM-2, with 2x1-sized blocks comparable to the state-of-the-art methods (HILL, WOW, S-UNIWARD). RS-diagram steganalysis conducted complies with *DE* evaluation results.**

Link to graphical and video abstracts, and to code: https://latamt.ieeer9.org/index.php/transactions/article/view/9901

***Index Terms*— Adaptation, data hiding, ensemble of classifiers, remainder modulo, RS-diagram, SPAM features, CNN**

I. Introduction

DATA-HIDING (DH) steganography methods aim hiding secret data (payload) in a cover (host) object [1], e.g. images. Resulting image with the payload embedded is known as a stego-image. Such images are represented in memory by matrices of pixels with *M* rows and *N* columns. For grayscale images, entries of the pixel matrix have integer values in the range from 0 to 255. The pixel value 255 represents full white (maximal intensity), and the value 0 encodes full black. The main aim of steganography is to get a stego-image not distinguishable from respective cover by the human visual system (HVS), as well as by statistical detectors. Image steganography methods [2] can work with colored, or grayscale images, cover image can be reversible, or irreversible, and embedding can be done in the pixels (spatial domain) or in transformed by Fourier, wavelet, or other transformations images.

The present paper considers Irreversible steganography methods for Grayscale images in Spatial domain. Such IGS-methods include [2], e.g. least-significant bit (LSB) substitution, pixel-value differencing (PVD), and multiple bit-planes (MBP).

In adaptive data hiding methods, image pixel characteristics, such as pixel value (PV), most significant bits (MSB), LSBs, and PVDs, are used to decide what, where, and how to embed in the pixels. The present research concerns adaptive steganography using MSB and PVD for deciding where and what to embed, and LSB method generalization for embedding. Adaptation can be to an image as a whole or to the parts of an image. Syndrome-trellis codes (STC) [3] and Optimal Optimal Pixel Adjustment Process (OOPAP) ([4], [5]), are examples of methods adapting to an entire image by optimizing distortion introduced by secret embedding. Adaptive to parts of an image methods define noisy-like parts where more payload can be embedded, and smooth parts for less (or not at all) embedding. Adaptation uses observation that changes in noisy-like parts are less detectable [2], [6]. Adaptive DH methods have the following features: 1, 2) The parts for splitting and embedding can be pixels, pairs of pixels, blocks of pixels, or blocks of bits in bit-planes of image. 3) Embedding can be made by PV change, PVD change, PV and pair-related function change, block change, block-related function change, inside block group-related function change, or block pixels permutation. 4) Selection decision criteria can use complexity function depending on PV, PVD, blocks of bits in bit-planes, or blocks of pixels. 5) Complexity function can be embedding invariant (i.e. preserved after embedding), or changing. In the case of non-invariant, changing criterion, adjustment can be applied to make criterion on the stego image matching to that of the cover image, the block can be left modified but the secret data are unassigned from it (the embedded portion of secret data is returned back to the secret stream for next embedding), or block marked as used/not used for embedding. 6) Parts of an image with high variability of PVs can represent edges. Complexity criteria used are represented as inequalities with thresholds, constant in [7], or tuned in [8]. Considered above features of IGS adaptive data hiding (ADH) methods are summarized in a compressed form in Fig. 1 where the second-level (dashed) blocks define methods' general features, and connected to them (solid lines) blocks below specify particular variants used by respective features.

No one adaptive method, to the best of our knowledge, sorts parts of an image by the complexity function value in order to embed first turn into the parts with higher complexity. In the case of the low payload (i.e. defined in Section II-A embedding rate ER and method's embedding capacity, EC, satisfy $ER < EC$), when not all of the image pixels are used to keep the embedded data, such sorting allows first turn embedding into

The associate editor coordinating the review of this manuscript and approving it for publication was Pedro Machado de Almeida (C*orresponding author: Gürcü Öz*).

A. G. Chefranov, and Gürcü Öz are with the Department of Computer Engineering of Eastern Mediterranean University, Famagusta, Cyprus (e-mails: Alexander.chefranov@emu.edu.tr, and gurcu.oz@emu.edu.tr).

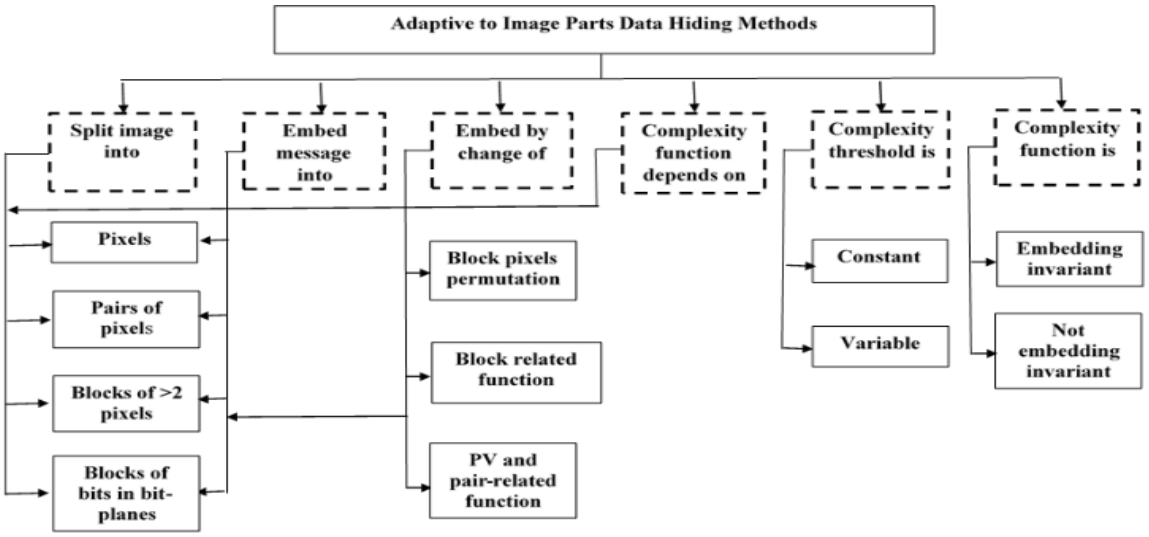

Fig. 1. Adaptive to image parts DH methods classification schema.

noisy-like areas and avoiding changes of the pixels in the smooth regions. The complexity function needs to be invariant (as in [9]) to avoid readjustment after embedding. However, the complexity function [9] can filter out many blocks because it is constructed by subtracting block elements from its left top element, thus, limiting its *EC*, and not allowing distinguishing various direction edges. Using embedding into bit-planes separately in [9] increases the computational time of the method compared to considering a pixel as a whole. The present paper contribution is proposing a new ADH method, adaptive RM-m (ARM-m) using:

- a new complexity embedding-invariant function defined on a block of pixels;
- embedding into pixels of the selected blocks by RM-m method generalizing k-LSB;
- ARM-m does not fall into the classification of Fig. 1 because it embeds first turn into the blocks with highest over entire image complexity.

Peak-signal-to-noise ratio (*PSNR*, see, e.g. [7], equation (8)), measured in dB is used as DH quality metric. Higher *PSNR* corresponds to less mean squared error (*MSE*) between the original and stego image. However, *PSNR* does not depend on the context, and, thus, embedding into the plain image regions and into noisy-like regions with high intensity gradients (textures, edges) yields the same *PSNR*. However, embedding into the noisy-like regions is less detectable by HVS and statistical detectors [2]. It is the reason for not discussing in the paper *PSNR* as not depending on the pixels selected for embedding that is the main idea of the proposal. Hence, ARM-m method is steganalized by ensemble of 23 classifiers using 2nd order 686 subtractive pixel adjacency matrix (SPAM) features. It has detection error (*DE*) higher than that for the state-of-the-art methods for the same payload (embedding rate, *ER*), $ER \leq 1$ bpp, bit-per-pixel. Also, it has rather high $DE \approx 28\%$ for $ER = 1.33$ bpp not reachable for the known adaptive methods. Also, CNNs (SRNet and YeNet) were used showing high DE comparable to that of HILL, WOW, S-UNIWARD methods for $ER \leq 0.5$ bpp. RS-diagram steganalysis confirms ARM-m resistance to it for $ER \leq 1.33$ bpp.

The rest of the paper is structured as follows. Section II introduces background and IGS ADH review. Section III proposes the ARM-m method. Section IV presents experimental results on SPAM and CNN steganalysis of the proposed and state-of-the art adaptive methods, and RS-diagram steganalysis of the proposed method. Section V concludes the paper.

II. Basic Notions and Adaptive Methods Review

In Subsection II-A, basic notions, related to embedding are introduced. Subsection II-B introduces adaptive to image parts DH methods according to Fig. 1.

A. Basic notions

EC is the average number of bits that can be embedded per pixel by the method provided that all cover image pixels are treated for embedding. ER is the average number of bits actually embedded per image pixel. *EC* and *ER* are measured in bits-per-pixel (bpp). PVD, pixel-value-difference, is defined as a difference between pixels of a block comprised of two pixels.

According to [10], if PSNR is greater than 30 dB, HVS is not able to discriminate between cover and stego images. However, statistical detectors using learning machines trained on images' feature vectors, such as SPAM [11], can easily detect high-PSNR stego images when embedding is done in smooth areas of an image. But embedding into coarse noisy-like areas, with high variety of pixel intensities inside them, allows resisting steganalysis better. The problem is in defining the noisy-like areas in the same way both by embedding and extraction parties.

When pixels for embedding are known, the well-known embedding method, k least-significant bit (k-LSB) substitution, can be used, substituting k LSBs of a pixel, p, by k-bit secret decimal data value, s, and producing the pixel, p'. Defining modulus m

$$m = 2^k, \tag{1}$$

such DH can be represented as:

$$p' = p - p \bmod m + s = \left\lfloor \frac{p}{m} \right\rfloor \cdot m + s, \quad (2)$$

where $s \in Z_m = \{0, .., m-1\}$, and $\lfloor x \rfloor$ is the floor function returning maximal integer not exceeding x. Similarly, the modified LSB DH method (MLSB, aka LSB OPAP [10], [12], [13], [5]) decreases error, $e(p, p') = |p - p'| \geq \frac{m}{2}$, by adding or subtracting the modulus, m, (1), to or from the pixel, p', (2) and returns stego pixel, pu represented by (3):

$$pu = \arg\min_x |x - p|, \quad (3)$$
$$x \in \{p', p' + m, p' - m\} \,\&\, 0 \leq x \leq 255,$$

Since adding/subtraction of m does not affect remainder after division by m, extraction of the secret data for the both methods, LSB and MLSB, is given in (4):

$$s' = p' mod\ m. \quad (4)$$

The k-LSB and MLSB methods embed in a defined pixel; they may be used both by adaptive and non-adaptive methods. PSNR of the methods is greater than $30\ dB$ for $k \leq 4$ (see, e.g. [5], Table 10 therein), and, hence, are good enough to be used for data hiding with respect to HVS detection.

B. Adaptive to Image Parts DH Methods Review

Particular adaptive to image parts DH methods are briefly introduced below mainly according to "Split image into" and "Embed message into" features shown in Fig. 1. Some of these methods are used in our comparison experiments presented in Section IV, A.

Splitting and embedding into individual pixels. Modulo operator embedding (MOE) method generalizes MLSB embedding (1)-(3) by using (2), (3) and omitting condition (1) on the modulus value, as in [10] where a threshold, T, splits pixel values into two sub-ranges. For the pixels with values less than T, lower range, MLSB embedding with optimal pixel adjustment procedure (OPAP) [12] is made using modulo m_l. Thus, pixel complexity function is defined by its value and threshold. For the pixels from the upper subrange with values greater or equal than T, another modulo, $m_u > m_l$ is used for DH. Thus, [10] splits image into individual pixels, all pixels are used for embedding, complexity function is not invariant, and data is embedded into pixels. The method has consistency problems discussed in [14].

In [15], an adaptive complexity based LSB matching (CB LSBM, or CBL) method is proposed using for embedding pixels with complexity not less than some threshold. Embedding is done by LSBM leaving LSB as is if it matches respective secret bit to be embedded. Otherwise, the pixel is modified by randomly adding/subtracting 1. The choice of 1 or -1 is done using a pseudo-random number generator (PRNG) seed of which being a secret shared by a sender and receiver. The complexity is calculated over 8-pixel neighborhood as a sum of absolute differences between them and the central pixel (for a specially constructed secondary image with all zeroed LSBs), and then the threshold is defined based on the secret message length. Thus, [15] splits image into individual pixels, noisy-like pixels only are used for embedding, with embedding invariant complexity function using a specially constructed secondary image, and data is embedded into pixels.

In [16], cover image pixels are classified as edge or non-edge pixels by their 3 MSBs. The rest 5 LSBs are used for secret bits embedding with x-LSB into edge, and y-LSB, into non-edge, pixels, where $5 \geq x > y \geq 1$. Values of x, y are also embedded allowing extractor using correct number of bits in the extraction process. Thus, [16] splits image into individual pixels, all pixels are used, embedding invariant complexity function uses MSB, and data is embedded into pixels.

Method [17] splits an image into individual pixels, not all pixels are used, embedding invariant complexity function uses MSB, and data is embedded into pixels' blue channel LSB using XOR operation of the message bit and a bit defined the password. The method works with colour images but uses for embedding just one colour that is why it can be treated as working with gray scale images.

Splitting into pairs and embedding into individual pixels. In method [7], data hiding uses neighbouring pixel pairs, PVD ranges, and a constant threshold on PV to define noisy-like areas with more bits embedding by k-LSB in the noisier blocks. In [18], an error of [7] related to the use of 7 bits embedding for one of the ranges is disclosed. It is fixed by proposing slightly different method avoiding the use of 7 bits embedding. In [19], the same error of [7] is also found, and fixed by using other threshold value (128 instead of 192 used in [7] and [18]), the number of bits embedded is from 3 to 5, the pairs are formed inside 2x2 blocks with one common for them reference pixel always getting 3 secret data bits. Adaptive modified LSB, AMLSB, [20] also uses for adaptation PVD ranges with more bits embedded by MLSB into pairs with higher PVD. IRMDR+PBPVD method is proposed in [21]. It also splits an image into disjoint pairs of neighbouring pixels, all pixels are used, embedding invariant complexity function uses PVD range, and data is embedded into pixels. Method [22] splits image into pixel pairs, all pairs are used, embedding invariant complexity function uses PVD range, and data is embedded into pixels by 3-LSB. Method [22] is improved in [23] replacing 3-LSB by 3-bit MLSB.

Splitting and embedding into pairs of pixels. PVD method [24] splits image into pixel pairs, all pixels are used, embedding invariant complexity function uses PVD range, and data is embedded into pixel pairs. Extraction (if falling-off-boundary condition is false) is done by $b = d' - l$, and converting b to a binary number of the bit size dictated by the current range, where d' is the new PVD value, and l is the lower border of the respective range. PVD method [24] using powers of two width ranges, is generalized to any width ranges in [25]**.** The method [25] needs converting an input binary stream to a multiple-base number. Method [26] aims improving [24] by considering inside 2×2 pixel blocks four pairs corresponding to edges with vertical, horizontal, and diagonal orientation.

Adaptive LSB matching revisited (ALSBMR) [27] embeds into pairs of cover pixels with absolute PVD not less than some threshold, $T \leq 31$, thus defining complexity function. ALSBMR splits image into pixel pairs, embedding invariant complexity function uses PVD range, and data is embedded into noisy-like pixel pairs.

Splitting into blocks of more than two pixels, and embedding into pixels, pairs, or blocks of pixels. Methods [28] and [29] split image into 3-pixel blocks, use all blocks, embedding invariant complexity function uses PVD range, and

data is embedded into pixels and pixel pairs. A method that splits a cover image, I, into two parts, one, $I1$, with six MSBs, and another one, $I2$, with two LSBs of every pixel, and embedding into each pixel of $I2$ and corner pixels of the 2x3-sized blocks of $I1$ is proposed in [30]. Methods [31] and [32] split image into 9-pixel blocks, use all blocks, embedding invariant complexity function uses PVD range, and data is embedded into pixels and pixel pairs. Similar methods but using embedding into quotient value (after division by 4) differences (QVD) between the block central and outer pixels for $3x3$ and $2x3$ blocks are proposed in [33], [34], [35], [36], and [37]. Some of them also keep secret data in the 2nd LSBs of the outer pixels, whereas the 1st LSBs keep either secret data or integrity verification bits.

A DH method MDPVD-MLSB is proposed [13]; it uses MLSB and MDPVD methods. It employs 9-pixel blocks, uses all blocks of an image, embedding invariant complexity function uses PVD range, and data is embedded into pixels and pixel pairs. Pixel rearrangement based steganography algorithm (PRSA) [38] splits an image into 9×9-pixel blocks, uses noisy-like blocks only, embedding complexity function uses JPEG non-zero coefficients number of which exceeds a threshold, and m-bit data is embedded into groups (blocks) of n pixels by transpositions (in the paper, $m = 2, n = 3$).

Splitting bit planes into blocks and embedding into blocks' groups of bits. MBP method [9] splits image into $n \times n$-bit-plane blocks, uses noisy-like blocks only, embedding invariant complexity function uses MSBs differences, and data is embedded into bit-plane blocks. Method [6] splits image into 8×8-bit-plane blocks, uses noisy-like blocks only, embedding invariant complexity function uses bit-change number, and data is embedded into bit-plane blocks. MPBDH (multi bit-planes block data-hiding) method [8] splits image into 8×8-bit-plane blocks (further divided into nine segments of seven consecutive bits used for embedding). It uses noisy-like blocks only, non-invariant embedding complexity function uses run-length irregularity [39], and the next 3-bit secret data is embedded into bit-plane blocks by flipping groups of at most two bits of the next segment.

III. Proposed ARM-m Method

As shown in Section II, adaptive data hiding methods use various splits of an image, structures, and embedding methods. Divisions use rectangular regions starting from the minimal row, r and column, c number in a block, $r \times c = 1 \times 1$-sized (separate pixels). The noise-like areas are most preferable for data embedding resistant to detection: "The most important fact here is that replacing a noise-like portion with any noise-like 8 × 8 binary blocks does not produce any visual change on the vessel", [6]. Hence, it is reasonable considering splitting into blocks with $r \cdot c > 1$ allowing estimating pixel intensity changes in a block characterizing their noisiness. Despite many methods use fixed m, n values (m-bit data is embedded into groups (blocks) of n pixels), for the sake of flexibility and better adaptation, it is reasonable having them as parameters with some default values, e.g., $r = c = 8$.

It is also seen from Section II that block complexity characterizing block noisiness may be defined by various ways. Since intensity is represented by numbers from $0..255$ for gray-scale images, it looks reasonable considering complexity functions defined on pixel intensities, not on bits. For example, in [9], despite embedding is done into bit-planes, complexity function is defined over numbers derived from the MSBs of the block pixels. When a particular bit-plane is considered and LSB is used for embedding into it, MSBs to the left of the bit-plane are not affected by the embedding and are invariant to it. Embedding in LSBs using MSBs as an embedding invariant is a reasonable choice.

As shown in Section II-A, LSB substitution uses modulo $m = 2^k$ operation for embedding/extraction. It can be easily generalized to remainder modulo m (RM-m) substitution defined by (2), (4), and not requiring (1). For k-LSB method, MSB part invariant to k-LSB embedding is $\left\lfloor \frac{p}{2^k} \right\rfloor$, and for RM-$m$, it is $I(p, m) = \left\lfloor \frac{p}{m} \right\rfloor$. Actually, from (2), it is easily seen that

$$I(p, m) = \left\lfloor \frac{p}{m} \right\rfloor = \left\lfloor \frac{p'}{m} \right\rfloor = I(p', m), \quad (5)$$

is invariant to RM-$\boldsymbol{m}$ embedding. For example, if. $\boldsymbol{m = 4, p = 239 = 59 * 4 + 3, I(239, 4) = \left\lfloor \frac{239}{4} \right\rfloor = 59}$. The secret $\boldsymbol{s = 1}$ is embedded replacing remainder of the pixel with resulting pixel $\boldsymbol{p' = 59 * 4 + 1 = 237}$.

In [9], blocks for embedding are selected such that all the differences between top-left pixel and other block's pixels after division by 2^{l+1}, where $l \in \{0, .., 7\}$ is the bit-plane number, i.e. $\left\lfloor \frac{p}{2^{l+1}} \right\rfloor$, are greater than a threshold selected as 0 or 1. The blocks selected may have quite different variances, but the order of using them for embedding is defined by a secret key, not by their variance. It can result in the use of blocks with less variance in embedding, and, thus, detection error will be less. It is reasonable sorting the blocks by variance decreasing, and embedding first turn into the blocks with the higher variance.

In [9], differences between the top-left and other block pixels are used to characterize variance, and it is required that all the differences shall be greater than the threshold. However, in natural images, neighbouring pixels with high probability are equal, and, hence, many high variance blocks are filtered out by this criterion. It is desirable having more robust criterion allowing tolerating such equalities. The following block, B, complexity invariant (BCI) function is proposed:

$$BCI(B) = \max_{p \in B} \left(\left\lfloor \frac{p}{m} \right\rfloor \right) - \min_{p \in B} \left(\left\lfloor \frac{p}{m} \right\rfloor \right). \quad (6)$$

BCI (6) is invariant to embedding because of (5). Compared to complexity function [9], (6) allows distinguishing more edges. For example, let $r = c = 2, m = 2, B = \begin{pmatrix} p_1 & p_2 \\ p_3 & p_4 \end{pmatrix} = \begin{pmatrix} 20 & 10 \\ 30 & 10 \end{pmatrix}$. Then, complexity [9] is $C = \min \left(\left| \frac{20}{2} - \frac{10}{2} \right|, \left| \frac{20}{2} - \frac{10}{2} \right|, \left| \frac{20}{2} - \frac{30}{2} \right| \right) = 5$, distinguishes edges between the left-top corner element and other elements in horizontal, diagonal, and vertical directions; and $BCI(B) = \max \left(\frac{20}{2}, \frac{10}{2}, \frac{30}{2}, \frac{10}{2} \right) - \min \left(\frac{20}{2}, \frac{10}{2}, \frac{30}{2}, \frac{10}{2} \right) = 10$, distinguishes additionally higher difference edges between the left-bottom, 30, and two right pixels, 10. Above discussions result in the following Adaptive RM-m (ARM-m) data hiding method proposal having embedding (Algorithm 1) and extraction (Algorithm 2) parts. From the viewpoint of Fig. 1 ARM-m splits an image into the

blocks and embeds into the pixels of the blocks. The blocks are sorted in the order of decreasing of the BCI. Note that this new approach is not reflected in the review of the known ADH methods, Section II, B, since no one of the methods considered uses sorting of the image parts to decide where first to embed the secret data.

Algorithm 1. *ARM-m Embedding(*
Inputs: gray-scale cover image, $CI(M,N)$; modulus, m; secret data stream, SDS, for embedding such that $SDS(i) \in Z_m, i = 1..S$, where $S \le M \cdot N$, the number of secret elements, is cardinality $|SDS|$ of SDS; row and column number in a block, r, c;
Outputs: stego-image $SI(M,N)$; the secret bit number actually embedded, S;
)
Begin
1. *Set $SI = CI$.*
2. *Define non-overlapping $r \times c$ blocks of SI: $B(i), i = 1..\left\lfloor \frac{M}{r} \right\rfloor \cdot \left\lfloor \frac{N}{c} \right\rfloor = |B|$.*
3. *Calculate block complexity invariant for each $B(i)$ according to (6):*
$$BCI(B(i)) = \max_{p \in B(i)} \left(\left\lfloor \frac{p}{m} \right\rfloor \right) - \min_{p \in B} \left(\left\lfloor \frac{p}{m} \right\rfloor \right), \ i = 1..|B|$$
4. *Sort B in descending order of BCI finding a sequence of indices, BI, such that $BCI\big(BI(i)\big) \ge BCI\big(BI(i+1)\big), i = 1..|B|-1$.*
5. *Represent blocks $B\big(BI(i)\big), i = 1..|B|$, as the sequence of pixels, $SP(M \cdot N)$*
6. *Embed secret data from SDS into the pixels of SP using RM-m according to (2):*
$$SP'(i) = SP(i) - SP(i) mod\ m + SDS(i), \quad i = 1..M \cdot N,$$
converting cover image pixels to the stego image pixels and counting the number S of embedded secret bits.
7. *Reshape stego-pixel sequence SP' to the output stego-image, $SI(M,N)$.*
End

Extraction of the secret data embedded by ARM-m is as follows:

Algorithm 2. *ARM-m Extraction(*
Inputs: gray-scale stego image, $SI(M,N)$; modulus, m; number of secret data items, $S \le M \cdot N$; row and column number in a block, r, c;
Outputs: extracted data stream, EDS, such that $EDS(i) \in Z_m, i = 1..S$;
)
Begin
1. *Define non-overlapping $r \times c$ blocks of SI: $B(i), i = 1..\left\lfloor \frac{M}{r} \right\rfloor \cdot \left\lfloor \frac{N}{c} \right\rfloor = |B|$.*
2. *Calculate block complexity invariant for each $B(i)$: $BCI(i) = \max_{p \in B(i)} \left(\left\lfloor \frac{p}{m} \right\rfloor \right) - \min_{p \in B} \left(\left\lfloor \frac{p}{m} \right\rfloor \right), \ i = 1..|B|$*
3. *Sort B in descending order of BCI finding a sequence of indices, BI, such that $BCI\big(BI(i)\big) \ge BCI\big(BI(i+1)\big), i = 1..|B|-1$.*
4. *Represent blocks $B\big(BI(i)\big), i = 1..|B|$, as the sequence of pixels, $SP(M \cdot N)$*
5. *Extract secret data from SP using RM-m according to (3): $EDS(i) = SP(i)\ mod\ m, i = 1..S$*
End

Beginning of Example 1 of embedding and extraction by ARM-m follows:
Beginning of Example 1. Consider embedding into a part (rows 87..90, columns 1..4) of the size 4×4 of the gray-scale version of the image ucid0001.tif shown in Fig. 2, (a). An arrowed square in Fig. 2, (a) specifies a noisy-like region (rows (87:103), columns (1:17)) shown in Fig. 2, (b) left-top corner of which is used in the example.

(a) (b)

Fig. 2. (a) Gray-scale version of the image UCID0001.tif, and (b) its part shown by an arrowed square (rows (87:103), columns (1:17)) left-top pixels of which are used in Beginning of **Example 1**.

Then let $\boldsymbol{M = N = 4, r = c = 2, m = 4}$,

$$SDS = (1,2,1,0,0,1,2), S = 7, CI = \begin{pmatrix} 239 & 238 & 239 & 255 \\ 255 & 250 & 253 & 253 \\ 241 & 255 & 254 & 254 \\ 254 & 254 & 254 & 248 \end{pmatrix}.$$

Then, according to Step 2 and using ordering by columns,
$$B(1) = \begin{pmatrix} 239 & 238 \\ 255 & 250 \end{pmatrix}, B(2) = \begin{pmatrix} 241 & 255 \\ 254 & 254 \end{pmatrix},$$
$$B(3) = \begin{pmatrix} 239 & 255 \\ 253 & 253 \end{pmatrix}, B(4) = \begin{pmatrix} 254 & 254 \\ 254 & 248 \end{pmatrix}.$$
According to Step 3, $BCI\big(B(1)\big) = 63 - 59 = 4$,
$BCI\big(B(2)\big) = 63 - 60 = 3$, $BCI\big(B(3)\big) = 63 - 59 = 4$,
$BCI\big(B(4)\big) = 63 - 62 = 1$.
According to Step 4, $BI = (1,3,2,4)$.

From BI, we see have that the noisisest block, B(1), where pixels differ by $17 = 255 - 238$ is to be embedded first, followed by B(3) with difference 16=255-239, and so forth. In this example, BCI and the maximal pixel difference in a block comply by order. Note that generally, these two orders may differ but not much. However, pixel difference cannot be used for blocks ordering since after embedding pixel difference can change contrary to BCI that remains invariant after embedding.
According to Step 5,
$$SP = (SI_{11} = 239, SI_{21} = 255, SI_{12} = 238, SI_{22} = 250, SI_{13} = 239, SI_{23} = 253, SI_{14} = 255, SI_{24} = 253, SI_{31} = 241, SI_{41} = 254, SI_{32} = 255, SI_{42} = 254, SI_{33} = 254, SI_{43} = 254, SI_{34} = 254, SI_{44} = 248).$$
According to Step 6,

$$SP = (SI_{11} = 237, SI_{21} = 254, SI_{12} = 237, SI_{22} = 248,$$
$$SI_{13} = 236, SI_{23} = 253, SI_{14} = 254, SI_{24} = 253,$$
$$SI_{31} = 241, SI_{41} = 254, SI_{32} = 255, SI_{42} = 254,$$
$$SI_{33} = 254, SI_{43} = 254, SI_{34} = 254, SI_{44} = 248),$$

and thus, according to Step 7,

$$SI = \begin{pmatrix} 237 & 237 & 236 & 254 \\ 254 & 248 & 253 & 253 \\ 241 & 255 & 254 & 254 \\ 254 & 254 & 254 & 248 \end{pmatrix}.$$

Let us consider now extraction from *SI*.

Then, according to Step 1,

$$B(1) = \begin{pmatrix} 237 & 237 \\ 254 & 248 \end{pmatrix}, B(2) = \begin{pmatrix} 241 & 255 \\ 254 & 254 \end{pmatrix},$$
$$B(3) = \begin{pmatrix} 236 & 254 \\ 253 & 253 \end{pmatrix}, B(4) = \begin{pmatrix} 254 & 254 \\ 254 & 248 \end{pmatrix}.$$

According to Step 2, $BCI\big(B(1)\big) = 63 - 59 = 4$, $BCI\big(B(2)\big) = 63 - 60 = 3$, $BCI\big(B(3)\big) = 63 - 59 = 4$, $BCI\big(B(4)\big) = 63 - 62 = 1$. It is seen that BCIs of the stego-image blocks are exactly the same as for the cover-image blocks: (1,3,2,4).

According to Step 3, $BI = (1,3,2,4)$.

According to Step 4,

$$SP = (SI_{11} = 237, SI_{21} = 254, SI_{12} = 237, SI_{22} = 248,$$
$$SI_{13} = 236, SI_{23} = 253, SI_{14} = 254, SI_{24} = 254,$$
$$SI_{31} = 241, SI_{41} = 254, SI_{32} = 255, SI_{42} = 254,$$
$$SI_{33} = 254, SI_{43} = 254, SI_{34} = 254, SI_{44} = 248).$$

And according to Step 5, the secret is extracted as

$$EDS(1..S) = EDS(1..7)$$
$$= (237\ mod\ 4, 254\ mod\ 4, 237\ mod\ 4, 248\ mod\ 4,$$
$$236\ mod\ 4, 253\ mod\ 4, 254\ mod\ 4\ ),$$

and thus $EDS(1..7) = (1,2,1,0,0,1,2) = SDS(1..7)$.

End of Beginning of Example 1.

Note that for the practical use, the number, S, of data items embedded, can be embedded in some predefined image part as side information.

IV. Experiments on ARM-m and Known Methods

Lenovo laptop having Intel® TM i5-62000 CPU @ 2.30 GHz, 8GB RAM, Windows 8, and Matlab R2017a was used in the experiments of Sections IV-A, IV-C. Section IV-A presents experimental results on SPAM steganalysis of ARM-*m* and known methods. Section IV-B shows results of CNN steganalysis of ARM-m by SRNet and YeNet [40] implemented in Python on 9MR10L1 IntelCore™ i9-14900 KF 3.20 GHz desktop with Windows 11 Pro and GPU Nvidia GeForce RTX 4090. Section IV-C applies RS-steganalysis to ARM-*m*.

A. SPAM steganalysis of ARM-m and known adaptive DH methods

Experiments on the 2nd order SPAM with threshold $T = 3$ steganalysis (686 features, available from [41]) were conducted on ARM-m and several looking the best adaptive to image parts DH methods discussed in Section II, B, including MBP [9] with $B \times B$-sized blocks, $B \in \{2,4,8\}$, and two bit planes used; AMLSB [20] with parameters $D_{1,2} = 15$, *l-h=2-3*; and ALSBMR [27] with $B \in \{4, 8, 12\}$. Also, adaptive to an entire image minimizing distortion syndrome-trellis code (STC) method [3], with 8×4-sized generating matrix $\hat{H} =$ [253 199 251 167], available from [42] was used.

They show superiority of the proposed method in terms of detection error (*DE*) and *EC*. Statistical detection of stego-images was made by Classification Learner application of MatlabR2017a with 23 classifiers.

A dataset of 886 color images from standard UCID [43], [44] was used. Gray scale images were obtained from the red channel of the images. Detection error is defined as

$$\boldsymbol{DE = \frac{FP+FN}{Total} \cdot 100\%}, \quad (7)$$

where FP, FN, and Total are the number of false positives (cover images that are recognized as stego), false negatives (stego images that are classified as cover), and total number of images used, respectively. Five-fold cross validation was used in training on 1600 images (800/800 cover/stego images), and other 172 images (86 cover and 86 stego) for testing for each payload (1st settings). A classifier showing in training the maximal accuracy was selected for testing.

Detection error dependence on embedding rate in bits per pixel (bpp) for four most efficient methods is shown on Fig. 3. To validate the results, the experiments were extended to 2658 gray scale image obtained from all three channels (red, green, and blue) of the original 886 colored images. Then 4000 images (2000 cover and 2000 respective stego) were used for training and the rest 1316 images (658 cover and 658 respective stego) were used for testing (2nd settings). Results of this experiment are shown by lines with '2000' in their legend. Generally *DE* decreases with the payload (bpp) increasing since distortion of an image grows that is easier to detect.

For the 1st settings, AMLSB has the lowest DE dropping from 22% to 1% for *ER* raising from 0.17 to 0.85 bpp. MBP for the block size $B = 2$ has the least *DE* (dropping from 36% to 21%) but the maximal *ER* range for that method spanning from 0.17 to 0.48 bpp. For MBP, with *B* growth, *DE* grows, and *ER* range shrinks.

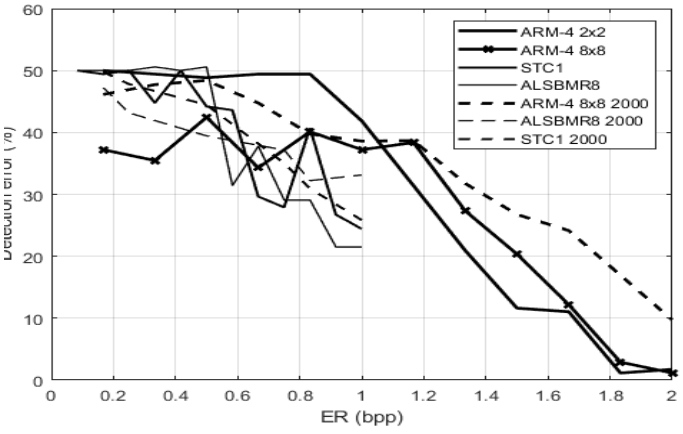

Fig. 3. Detection error by ensemble of classifiers using the 2nd order SPAM with T=3 (686 features) for ARM-4 $B \times B$, $B \in \{2,8\}$), STC (STC1), ALSBMR (ALSBMR8 for $B = 8$). Results of testing on 658 images of the models trained on 2000 images are shown by dash lines denoted 2000 in the legend.

ALSBMR for $B = 8$ has DE, the highest among this method's variants with $B \in \{4,8,12\}$, of about 50% for $ER \leq 0.5$ bpp. Since for ALSBMR, $EC = 1$ bpp, ER is limited by it. ALSBMR with $B = 8$ has $DE = 21.51\%$ for $ER = 1\ bpp$. Optimizing distortion STC method has DE comparable to the best ALSBMR variant for $ER \leq 0.5$ bpp, and slightly higher for $ER \in [0.5, 1]$ bpp. STC method has $DE = 24.42\%$ for $ER = 1\ bpp$.The proposed ARM-4 method with $B = 2$ supersedes by DE all other compared with methods for $ER \leq 1$ bpp. Moreover, it and other ARM-m variants for $m \in \{4, 8, 16\}$ and $B \in \{2,8\}$ used in the experiments have acceptable $DE > 20\%$ for $ER \in [1, 1.35]$ bpp. In that ER range, variants with $\{m = 4, B = 8\}$, $\{m = 8, B = 2\}$, and $\{m = 16, B = 8\}$ are better by DE than the variant with $\{m = 4, B = 2\}$.

For the 2nd settings (2000 in legend), it is seen that behaviour of the methods, ALSBMR, STC, and ARM, is similar for that in the 1st settings, but DE for ARM-4 with $B = 8$ slightly increased from 37.1% to 38.60% for $ER = 1\ bpp$, and DE for ALSBMR with $B = 8$ increased from 21.51% to 33.13% still preserving superiority of ARM. STC is better than ALSBMR for $B = 8$ for $ER \leq 0.5\ bpp$. Both STC and ALSBMR are worse than ARM with $m = 4$ and $B = 8$ for all payloads $ER \leq 1\ bpp$.

B. CNN-steganalysis of ARM-m

Two CNNs, SRNet and YeNet [40] available from [45] were used on UCID images resized by Matlab *imresize()* to 256×256, converted to three gray-scale images resulting in 2658 images, and split into $1500/500/658$-element subsets for respectively training/validation/testing. DE for ARM-m $2 \times B$ for $m \in \{2,4\}, B \in \{1,2\}$ is given in Fig. 4 for the payload $ER \in [0.05,0.5]$ bpp (six runs a point). It is seen that ARM-4 $2x2$ (denoted by 1) has DE falling from 30% to 15% that is not good. Better results shows ARM-2 2×2 (denoted by 2). However, ARM-2 2×1 shows DE compatible with best to date known results both for SRNet (denoted by 3) and YeNet (denoted by 4). In [40], Table 1 the highest $DE = 23.53\%$ when using SRNet and YeNet for $ER = 0.2$ bpp is reached for HILL for BOSSbase and BOSS2 image datasets. In [46], Table 5, maximal $DE = 46.21\%$ for HILL is reached by YeNet for this ER and BOSSbase and BOSS2 datasets. And in [47], Table 2, maximal $DE = 48.8\%$ for HILL-P and the same $ER = 0.2$ bpp for BOSSbase and UCID datasets. In our experiments with ARM-2 2×1, minimal, average, and maximal DE reached by SRNet for this ER and UCID dataset are, respectively, 22.80%, 42.51%, and 47.12% that is comparable to the best currently known results. ARM-2 2×1 on BOSSbase split into $5000/1000/4000$-element subsets of images for training/validation/testing for 0.1 bpp payload showed $DE \in [34.66\%, 41.36\%]$ that is better than 31.34% of HILL shown in Table 1 of [40]. On the other hand, computational complexity of ARM-m is significantly lower and defined by BCI sorting. Average total time of embedding 0.4 bpp payload followed by extraction for a 256×256-sized grayscale image on the laptop is 9 ms.

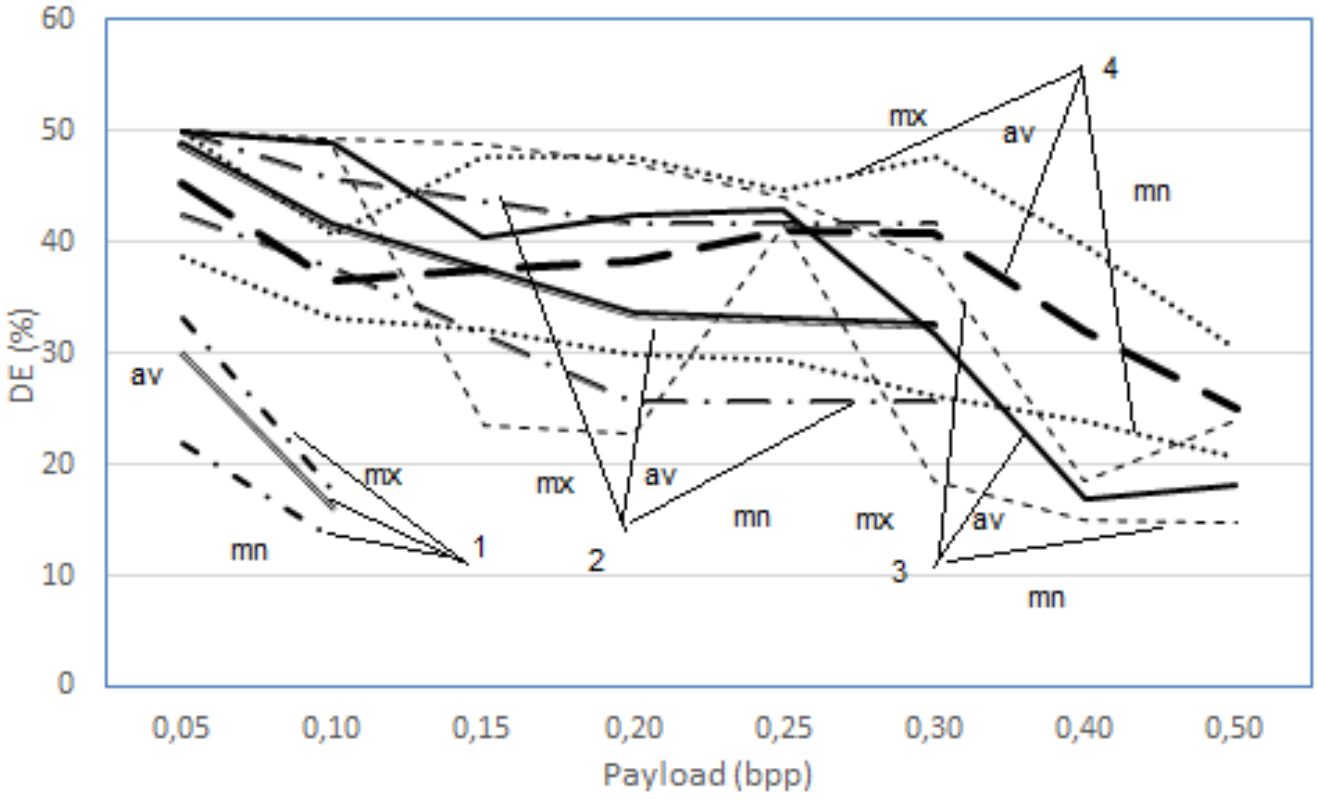

Fig. 4. Minimal (mn), maximal (mx), and average (av) DE (out of six runs) for 2658 UCID dataset images for ARM-4, 2×2 (1); ARM-2, 2×2 (2); and ARM-2, 2×1 (3) by SRNet, and for ARM-2, 2×1 (4) by YeNet.

C. RS-steganalysis of ARM-m

To verify SPAM steganalysis results, RS-steganalysis [48] was also used but applied to 512×512-sized grayscale images from [43] not used in SPAM analysis Section IV-A; they are given in Fig. 5 (top row: (a) Baboon, and (b) Barbara).

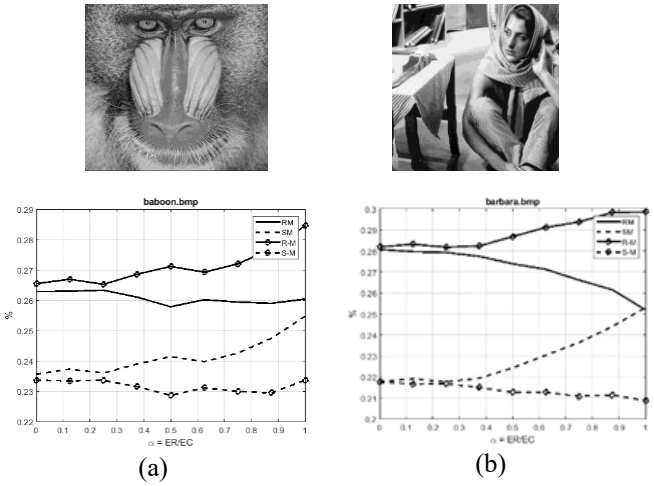

Fig. 5. Images used in our experiments on RS-steganalysis (top row) and corresponding RS-diagrams (bottom row) for ARM-*m* with *m*=4 and *B*=2 for (a) Baboon (b) Barbara.

RS-diagrams with mask $M = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ for ARM-m with $m = 4, B = 2$ for images presented are also displayed in Fig. 5 (bottom row).

The main idea of RS-steganalysis is to split an image into disjoint groups of neighboring pixels, disturbing them by ± 1 by some mask and its negation, calculating discrimination function for the groups, and classifying the groups into regular (R), singular (S), and unusable (U) ones. In RS-steganalysis, cardinalities of the classes divided by the number of groups and expressed in percentages, for the mask M, are denoted by R_M, S_M, U_M such that $R_M + S_M + U_M = 100\%$. The statistical hypothesis used is: percentage of R (S) groups for mask and its negation is approximately the same for images without payload, and differs for stego images: the greater payload, the greater difference. Payload is shown by the horizontal axis in the range [0,1], with 0 corresponding to no payload, and 1 to the maximal possible for a method payload equal its embedding capacity; $EC = log_2 m$ for ARM-m.

From Fig. 5, bottom row, it is seen that $\boldsymbol{R_M, R_{-M}}$, solid lines, and $\boldsymbol{S_M, S_{-M}}$, dashed lines, for $\boldsymbol{\alpha \approx 0.5}$ go side-by-side, and then tend diverging. This complies with SPAM steganalysis results that *DE* is rather high for $\boldsymbol{ER \le 1 = EC \cdot \alpha = 2\alpha}$ bpp, since $\boldsymbol{EC = log_2 4 = 2}$ bpp.

V. CONCLUSION

The paper considers a problem of irreversible secret DH, producing stego images resistant to steganalysis in spatial domain of gray-scale cover images. Stego image detection error is maximized when data are hidden (embedded) into noisy-like image areas where pixel values vary significantly. It is proved herein that generalization of the well-known LSB substitution to RM-m DH method has an embedding invariant (5) preserved after DH. A new adaptive remainder modulo *m*, ARM-m, method hiding data first in the maximal noisy blocks by RM-m is proposed that shows good performance due to the content aware adaptive embedding. ARM-m uses the embedding invariant (5) of RM-m to construct the block complexity embedding invariant BCI function (6) for adaptation to an image. The use of BCI (6) guarantees that the data are extracted exactly from the same blocks and in the same order as in the embedding process. Ensemble of 23 classifiers and the 2nd order SPAM with 686 features were used to evaluate stego-image detection error *DE*. Compared to the state-of-the-art methods, for the 1st settings, ARM-4 with 2×2 blocks has $DE = 41.86\%$ versus 24.42% of the best known STC method for $ER = 1$ bpp. For $ER = 1.33$ bpp, not reachable for known adaptive methods, ARM-4 and ARM-16, both with 8×8 blocks, have $DE = 27.33\%$ and 27.91%, respectively. ARM-4 is confirmed to be better than other methods also for 2658 gray-scale images derived from UCID v.2 dataset in the 2nd settings. Steganalysis by two CNNs (SRNet and YeNet) revealed high resistance of ARM-2, with 2x1-sized blocks comparable to the state-of-the-art methods (HILL, WOW, S-UNIWARD). RS-diagram steganalysis conducted complies with *DE* evaluation results. Future research will be devoted to more comprehensive study of the proposed method.

REFERENCES

[1] R. Bohme, Advanced Statistical Steganalysis, vol. 16, Springer, 2010, pp. 288. doi: https://doi.org/10.1007/978-3-642-14313-7.

[2] I. J. Kadhim , P. Premaratne, P. J. Vial and B. Halloran, "Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research," *Neurocomputing,* vol. 335, p. 299–326, 2019. doi: https://doi.org/ 10.1016/j.neucom.2018.06.075.

[3] T. Filler, J. Judas and J. Fridrich, "Minimizing additive distortion in steganography using syndrome-trellis codes," *IEEE Transactions on Information Forensics and Security,* vol. 6, no. 3, pp. 920-935, September 2011. doi: 10.1109/TIFS.2011.2134094.

[4] E. Ansari, M. Keshtkaran, R. Wallace, S. M. Mirsadegh and F. Ansari, "OOPAP and OPVD: Two innovative improvements for hiding secret data into images," *Iran J Sci Technol Trans Electr Eng,* vol. 43, pp. 55-65, 2019. doi: https://doi.org/10.1007/s40998-018-0090-4.

[5] A. Chefranov and G. Öz, "On LSB, LSB OPAP, and LSB OOPAP published experimental results correctness," *Iranian Journal of Science and Technology, Transactions of Electrical Engineering,* vol. 46, p. 609–619, June 2022. doi: https://doi.org/ 10.1007/s40998-022-00491-8.

[6] E. Kawaguchi, "BPCS-steganography - principle and applications," in *9th International Conference, KES 2005, Proceedings, Part IV*, Melbourne, Australia, September 14-16, 2005, 2005.

[7] M. Khodaei, B. Bigham and K. Faez, "Adaptive data hiding, using pixel-value differencing and LSB substitution," *Cybernetics and Systems,* vol. 47, no. 8, pp. 617-628, 2016. doi: 10.1080/01969722.2016.1214459.

[8] T. D. Nguyen, S. Arch-int and N. Arch-int, "An adaptive multi bit-plane image steganography using block data-hiding," *Multimed Tools Appl,* vol. 75, p. 8319–8345, 2016. doi: 10.1007/s11042-015-2752-9.

[9] B. C. Nguyen, S. M. Yoon and H.-K. Lee, "Multi bit plane image steganography," in *Digital Watermarking. IWDW 2006, 4283, pp. 61-70*, Berlin Heidelberg, 2006. doi: https://doi.org/ 10.1007/11922841_6.

[10] S.-J. Wang, "Steganography of capacity required using modulo operator for embedding secret image," *Applied Mathematics and Computation,* vol. 164, pp. 99-116, 2005. doi: https://doi.org/10.1016/j.amc.2004.04.059.

[11] T. Pevný, P. Bas and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *IEEE Transactions on Information Forensics and Security,* vol. 5, no. 2, pp. 215-224, June 2010. doi: 10.1109/TIFS.2010.2045842.

[12] C.-K. Chan and L. M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognition,* vol. 37, p. 469 – 474, 2004. doi: https://doi.org/ 10.1016/j.patcog.2003.08.007.

[13] K. A. Darabkh, A. K. Al-Dhamar and I. F. Jafar, "A new steganographic algorithm based on Multi Directional PVD and Modified LSB," *Information Tevhnology and Control,* vol. 46, no. 1, pp. 16-36, 2017. doi: 10.5755/j01.itc.46.1.15253.

[14] A. G. Chefranov and G. Öz, "Remainder with Threshold Substitution Data Hiding Scheme: Counterexamples and Modification," *IEEE Latin America Transactions,* vol. 21, no. 12, pp. 1255-1265, December 2023. doi: 10.1109/TLA.2023.10305236.

[15] V. Sabeti, S. Samavi and S. Shirani, "An adaptive LSB matching steganography based on octonary complexity measure," *Multimed. Tools Appl.,* vol. 64, p. 777–793, 2013. doi: 10.1007/s11042-011-0975-y.

[16] J. Bai, C.-C. Chang, T.-S. Nguyen, C. Zhu and Y. Liu, "A high payload steganographic algorithm based on edge detection," *Displays,* vol. 46, pp. 42-51, 2017. doi: http://dx.doi.org/10.1016/j.displa.2016.12.004.

[17] M. R. Islam, T. R. Tanni, S. Parvin and M. J. Sultan, "A modified LSB image steganography method using filtering algorithm and stream of password," *Information Security Journal: A Global Perspective,* pp. 1-12, November 2020. doi: https://doi.org/ 10.1080/19393555.2020.1854902.

[18] M. Hussain, Q. Riaz, S. Saleem, A. Ghafoor and K.-H. Jung, "Enhanced adaptive data hiding method using LSB and pixel value differencing," *Multimedia Tools and Applications,* vol. 80, p. 20381–20401, 2021. doi: https://doi.org/ 10.1007/s11042-021-10652-2.

[19] W. Wu and H. Li, "A novel scheme for random sequential high-capacity data hiding based on PVD and LSB," *Signal, Image and Video Processing,* vol. 10, p. 2277–2287, 2024. doi: https://doi.org/ 10.1007/s11760-023-02900-9.

[20] C. H. Yang, C. Y. Weng, S. J. Wang and H.-M. Sun, "Adaptive data hiding in edge areas of images with spatial LSB domain systems.," *IEEE Transactions Information Forensics and Security,* vol. 3, no. 3, p. 488–497, 2008. doi: 10.1109/tifs.2008.926097.

[21] M. Hussain, A. W. A. Wahab, A. Hoc, N. Javed and K.-H. Jung, "A data hiding scheme using parity-bit pixel value differencing and improved rightmost digit replacement," *Signal Processing: Image Communication,* vol. 50, p. 44–57, 2017. doi: https://doi.org/ 10.1016/j.image.2016.10.005.

[22] H.-C. Wu, N.-I. Wu, C.-S. Tsai and M.-S. Hwang, "Image steganographic scheme based on pixel-value differencing and LSB replacement methods," *IEE Proc.-Vis. Image Signal Process.,* vol. 152, no. 5, pp. 611-615, October 2005. doi: 10.1049/ip-vis:20059022.

[23] C.-H. Yang, C.-Y. Weng, S.-J. Wang and H.-M. Sun, "Varied PVD+ LSB evading detection programs to spatial domain in data embedding systems," *The Journal of Systems and Software,* vol. 83, p. 1635–1643, 2010. doi: https://doi.org/10.1016/j.jss.2010.03.081.

[24] D.-C. Wu and W.-H. Tsai, "A steganographic method for images by pixel-value differencing," *Pattern Recognition Letters,* vol. 24, p. 1613–1626, 2003. doi:10.1016/S0167-8655(02)00402-6.

[25] D.-C. Wu and D. Chun, "Image Steganography by Pixel-Value Differencing Using General Quantization Ranges," *Computer Modeling in Engineering & Sciences,* vol. 141, no. 1, pp. 353-383, 2024. doi: https://doi.org/10.32604/cmes.2024.050813.

[26] K.-C. Chang, C.-P. Chang, P. S. Huang and T.-M. Tu, "A novel image steganographic method using tri-way pixel-value differencing," *Journal of Multimedia,* vol. 3, no. 2, pp. 37-44, June 2008. doi:10.4304/jmm.3.2.37-44.

[27] W. Luo and F. Huang, "Edge adaptive image steganography based on LSB matching revisited," *IEEE Transactions on Information Forensics and Security,* vol. 5, no. 2, pp. 201-214, June 2010. doi: 10.1109/TIFS.2010.2041812.

[28] H.-H. Liu, P.-C. Su and M.-H. Hsu, "An improved steganography method based on least-significant-bit substitution and pixel-value differencing," *KSII Transactions on Internet and Information Systems,* vol. 14, no. 11, pp. 4537-4556, Nov. 2020. doi: http://doi.org/10.3837/tiis.2020.11.016.

[29] M. Khodaei and K. Faez, "New adaptive steganographic method using least-significant-substitution and pixel-value differencing," *IET Image Processing,* pp. 1-10, August 2012. doi: 10.1049/iet-ipr.2011.0059.

[30] S. Singh, "Adaptive PVD and LSB based high capacity data hiding scheme," *Multimedia Tools and Applications,* vol. 79, p. 18815–18837, 2020. doi: https://doi.org/ 10.1007/s11042-020-08745-5.

[31] S. Kang, H. Park and J. Park, "Combining LSB embedding with modified Octa-PVD embedding," *Multimedia Tools and Applications,* vol. 79, pp. 21155-21175, August 2020. doi: https://doi.org/ 10.1007/s11042-020-08925-3.

[32] G. Swain, "Digital image steganography using eight-directional PVD against RS analysis and PDH analysis," *Advances in Multimedia,* vol. 2018, no. 4847098, p. 13, 2018. doi: 10.1155/2018/4847098.

[33] S. Swain, "Two new steganography techniques based on quotinet value differencing with addition-subtraction logic and PVD with modulus function,," *Optik,* vol. 180, pp. 807-823, February 2019. doi: https://doi.org/10.1016/j.ijleo.2018.11.015.

[34] R. Sonar and G. Swain, "Steganography based on quotient value differencing and pixel value correlation," *CAAI Transactions on Intelligence Technology,* vol. 6, no. 4, pp.

504-519, December 2021. doi: https://doi.org/10.1049/cit2.12050.

[35] S. Kosuru, A. Pradhan, K. A. Basith, R. Sonar and G. Swain, "Digital image steganography with error correction on xxtracted data," *IEEE Access,* vol. 11, pp. 80945-80957, 2023. doi: 10.1109/ACCESS.2023.3300918.

[36] G. Swain, "Very high capacity image steganography technique using quotient value differencing and LSB substitution," *Arabian Journal for Science and Engineering,* vol. 44, p. 2995–3004, 2019. doi: https://doi.org/ 10.1007/s13369-018-3372-2.

[37] D. B. Khadse and G. Swain, "Data hiding and integrity verification based on quotient value differencing and Merkle tree," *Arabian Journal for Science and Engineering,* vol. 48, p. 1793–1805, 2023. doi: https://doi.org/ 10.1007/s13369-022-06961-9.

[38] A. Sur, V. Ramanathan and J. Mukherjee, "Pixel rearrangement based statistical restoration scheme reducing embedding noise," *Multimed. Tools Appl.,* vol. 68, p. 805–825, 2014. doi: 10.1007/s11042-012-1078-0.

[39] S. Sun, "A new information hiding method based on improved BPCS steganography," *Advances in Multimedia,* vol. 2015, p. 698492, 2015. doi: http://dx.doi.org/10.1155/2015/698492.

[40] M. Boroumand, M. Chen and J. Fridrich, "Deep Residual Network for Steganalysis of Digital Images," *IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 14, NO. 5, MAY 2019,* vol. 14, no. 5, pp. 1181-1193, May 2019.

[41] J. Fridrich, V. Holub and T. Denemark, July 2021. [Online]. Available: http://dde.binghamton.edu/download/feature_extractors/.

[42] T. Filler, J. Fridrich and J. Judas, July 2021. [Online]. Available: http://dde.binghamton.edu/download/syndrome/.

[43] Dataset, "Dataset of standard 512X512 grayscale test images," 2018. [Online]. Available: http://decsai.ugr.es/cvg/CG/base.htm. [Accessed 21 May 2025].

[44] G. Schaefer and M. Stich, "UCID: an uncompressed color image database," in *Storage and Retrieval Methods and Applications for Multimedia 2004, 20 January 2004, 5307, 472–481*, San Jose, CA, USA, 2004. doi: https://doi.org/ 10.1117/12.525375.

[45] G. Li, "GitHub," 26 12 2023. [Online]. Available: https://github.com/albblgb/Deep-Steganalysis. [Accessed 11 10 2025].

[46] S. Agarwal, C. Kim and K.-H. Jung, "Steganalysis of Context-Aware Image Steganography Techniques Using Convolutional Neural Network," *Appl. Sci.,* vol. 12, no. 10793, pp. 1-15, 2022.

[47] S. Li, Z. Wang, X. Zhang and X. Zhang, "Robust Image Steganography Against General Downsampling Operations With Lossless Secret Recovery," *IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 21, NO. 1, 2024,* vol. 21, no. 1, pp. 340-352, JANUARY/FEBRUARY 2024.

[48] J. Fridrich, M. Goljan and R. Du, "Detecting LSB steganography in color and gray-scale images," *IEEE Multimedia,* vol. 8, no. 4, pp. 22-28, October-December 2001. doi: 10.1109/93.959097.

Alexander G. Chefranov received his Engineer in Applied Mathematics, PhD and Doctor of Engineering Sciences from Taganrog State Radio-Engineering University, Taganrog, Russia. Currently, he is Professor in the Department of Computer Engineering of Eastern Mediterranean University. His research interests include information security, parallel processing, distributed systems, real-time systems, scientific computing.

Gürcü Öz received her B.S, M.S. degrees from the Electrical and Electronic Engineering Department and Ph.D. degree from the Computer Engineering Department of Eastern Mediterranean University, in Famagusta, North Cyprus. Currently, she is Professor in the Department of Computer Engineering of Eastern Mediterranean University. Her research interests include computer networks, wireless ad hoc networks, distributed systems, cloud computing, system simulation, information security.