International Journal of Pattern Recognition
and Artificial Intelligence
Vol. 32, No. 1 (2018) 1860010 (17 pages)

DOI: 10.1142/S0218001418600108

World Scientific
www.worldscientific.com

A New Optimization Strategy for Solving the Fall-Off Boundary Value Problem in Pixel-Value Differencing Steganography

Ismael R. Grajeda-Marín[*,§], Héctor A. Montes-Venegas[*,¶], J. Raymundo Marcial-Romero[*,‖], J. A. Hernández-Servín[*,**], Vianney Muñoz-Jiménez[*,††] and Guillermo De Ita Luna[†,‡‡]

[*]*Facultad de Ingeniería*
Universidad Autónoma del Estado de México, Toluca, Mexico
[†]*Facultad de Cs. de la Computación*
Benemérita Universidad Autónoma de Puebla, Mexico
[§]*tione_210@hotmail.com*
[¶]*hamontesv@uaemex.mx*
[‖]*jrmarcialr@uaemex.mx*
[**]*xoseahernandez@uaemex.mx*
[††]*vmunozj@uaemex.mx*
[‡‡]*deita@cs.buap.mx*

Received 1 December 2016
Accepted 27 June 2017
Published 27 July 2017

In Digital Image Steganography, *Pixel-Value Differencing* (PVD) methods use the difference between neighboring pixel values to determine the amount of data bits to be inserted. The main advantage of these methods is the size of input data that an image can hold. However, the *fall-off boundary problem* and the *fall in error problem* are persistent in many PVD steganographic methods. This results in an incorrect output image. To fix these issues, usually the pixel values are either somehow adjusted or simply not considered to carry part of the input data. In this paper, we enhance the *Tri-way Pixel-Value Differencing* method by finding an optimal pixel value for each pixel pair such that it carries the maximum input data possible without ignoring any pair and without yielding incorrect pixel values.

Keywords: Optimization; steganography; tri-way pixel-value differencing; TPVD; PVD.

1. Introduction

Steganography is the set of techniques for hidden communication. In digital multimedia, this is achieved by inserting arbitrary content inside a digital medium, keeping the existence of the carried message undetected. Three notable uses of digital steganography are (i) to provide an invisible copyright proof, (ii) to guarantee the

[¶]Corresponding author.

integrity of digital content, and (iii) to exchange large amounts of secret information. Because the secret message is hidden inside a carrier file, it should look innocent or completely imperceptible to an external observer.

The carrier media could be any digital file, but images, audio, and video files are very common choices.[5] These media are often selected because redundancies can be found in the structure of the file or even because data noise can be potentially replaced with the secret message. Scores of research papers as well as surveys and reviews of current steganographic methods are readily available in the literature.[5,19,20,24]

In digital image steganography, the secret data is hidden within an image using its pixel value intensities in a way that the quality of the carrier image is not affected. The image with the hidden data is called the stego-image. Steganographic methods usually work either in the *spatial domain* or in the *frequency domain* of the image.[5] When using the spatial domain, the message data is embedded by directly manipulating the pixel values of the image; whereas in the frequency domain, the image is first mapped into an alternate domain before hiding the secret information.

Steganography techniques strive for both a *high capacity* and a *low detectability* of the hidden data. The capacity should be as high as possible and the distortion caused to the original image should be as low as possible. The capacity is commonly measured in terms of the size of the stego content or in terms of a relative embedding rate value (e.g. bits-per-pixel). Similarly, the distortion in the stego-image is measured by the *peak signal-to-noise ratio* (PSNR). The higher the PSNR, the lower the distortion produced in the carrier image. Also, in a parallel area called *steganalysis*, a set of techniques have been developed to assess the detectability strength of a steganographic method.[10]

Steganography methods in the spatial domain have been grouped into different categories,[28] namely, Least Significant Bit (LSB) methods,[7] RGB methods,[9,29] Mapping-based methods,[1,31] Code-based methods,[11,22,36] and Palette-based methods.[34,38] Each category has its own strengths and weaknesses, and all of them strive for the ultimate goals of high capacity and low detectability mentioned above. In line with these goals, a number of methods have been developed that exploit the absolute difference between neighbor pixel pairs to hide the message data. These methods are classified as Pixel-Value Differencing methods (PVD). The seminal PVD method for gray-level images was designed by Wu and Tsai[33] and reported to produce a high capacity stego-image and a substantial image quality. Thereafter, several variations and enhancements to the PVD have been produced.[25] For instance, Pradhan *et al.*[23] tested PVD variations for two, three and four neighbor sizes. The results were as expected and showed that the capacity increases with the neighbor size, while the stego-image quality remains acceptable. Note that the difference between the chosen pixels can also be computed in any neighboring direction and in various neighbor sizes.[2,26,27]

Ideally, all pixels of the carrier image should be used to embed as much message data as possible. However, many PVD methods yield pixel values that fall off the valid interval and either ignore them or simply reduce the number of message bits to be inserted. This is known as the *fall-off boundary problem*. Ignoring pixels to hide

data either leads to a lower capacity or forces to include additional strategies to retrieve the embedded data that may reveal the existence of a hidden message.[6]

In this paper, we modify the Tri-way Pixel-Value Differencing (TPVD) method[3] and find an optimal pixel value for each computed pixel block such that their difference carries the maximum input data possible without omitting any block and without yielding any incorrect pixel values. The method reduces the size of the search space and computes a very small set of feasible solutions. In addition, two more strategies are discussed to further increase the size of the hidden message. An extensive experimental evaluation shows the feasibility of the method.

The paper proceeds by first covering the basics of the PVD in Sec. 2 and by surveying variations of this method. Section 3 presents a detailed description of our optimization strategy. Section 4 presents an extensive experimental evaluation and a summary of our results. Section 5 concludes the paper.

2. Pixel-Value Differencing

The original PVD method[33] by Wu and Tsai takes the bit representation of the secret message and embeds it into a gray-level image using the intensity difference of two consecutive pixels. The embedding capacity of a pixel pair depends on their difference value. Areas of the image with larger pixel intensities differences can embed more message bits than others. This happens more frequently in areas with edges and less often in smoother or flat regions. The chief idea is to modify the pixels by adding a decimal conversion of the message bits in a way that their value difference is kept and the image quality is preserved. Each computed difference is mapped into one of a predefined set of intervals arranged as a table, which determines the number of bits to be inserted. This table is designed by the experimenter using a power of two for each interval width, either to provide a large capacity or a high imperceptibility.[33,35] Other approaches have designed the intervals table based on the *perfect square number*,[30] or have replaced it with a well-crafted logarithmic function.[6]

The PVD algorithm exhibits a high embedding capacity but is vulnerable to statistical-based stegoanalysis methods such as RS and Chi-square analysis.[16] For instance, Zhang and Wang[37] showed that a close look to the stego-image histogram can reveal the presence of a secret message and that even its length could be estimated. To counter this weakness, they proposed a modified PVD algorithm that preserves both the high capacity and the low perceptibly of the method and avoids the secret message detection. Later, Chang *et al.*[4] modified the PVD to notably increase its capacity, while keeping an acceptable image quality.

Other directions for data embedding have also been explored based on the PVD method. Wang *et al.*[32] adapted the PVD to reduce the distortion on the stego-image, and also to reduce the risk of detection. First, the difference of a pixel pair is computed in the original PVD fashion. Then a modulus function is used to determine the remainder of the two consecutive pixels to hide the secret data by altering that remainder. The results show that the modulo operation reduces significantly the

image distortion and also increases imperceptibility. This method also avoids the *falling-off boundary problem* (i.e. pixel values out of the valid interval) at the cost of a lower capacity and lower security. Later, Joo *et al.*[8] improved this approach in terms of capacity and stego-image quality, which in turn, also increased the security of the hidden data.

Liao *et al.*[13] combined the PVD and a LSB substitution method. Here, the carrier image is divided into four nonoverlapping pixel blocks. LSB insertion is then used to embed a number of data bits determined by the average difference value of the pixel blocks. As this technique uses LSB substitution, the stego-image has less imperceptibility but a higher hiding capacity.

Instead of using as single difference pixel pair, Lin *et al.*[14] used a three nonoverlapping pixel block to compute two absolute difference values. The middle pixel of this block is then used to hide the secret information. Likewise the PVD, the amount of data bits embedded depends on the differences of the pixel block been examined. By using more pixel blocks, the average embedding capacity was increased. Luo *et al.*[15] then modified this algorithm to reduce the amount of data embedded in smoother regions to achieve a higher stego-image quality.

If more pixel blocks increase the hiding capacity of a steganographic method, more image channels are expected to produce a similar outcome. For this reason, Mandal and Das[17] extended the PVD technique to color images. Similarly, Swain[27] reported an adaptive PVD method using vertical and horizontal pixel directions with 2×2 and 3×3 pixel blocks. The first technique offers good capacity and the second one provides good quality. Other studies have showed that quality is higher with five neighbors while capacity is higher with eight neighbors.[26,27]

Trying to find a balance between high capacity and better quality, Chang *et al.*[3] proposed a modified version of the PVD named *Tri-way Pixel-Value Differencing* (TPVD). While the PVD inserts data in only one pixel pair, the TPVD uses horizontal, vertical and diagonal differences in 2×2 pixel blocks to hide input data, thus achieving a higher capacity in the stego-image. Lee *et al.*[12] later extended this technique to embed a larger image or a group of images into another image.

In PVD-based methods, the higher the difference values, the higher the data that can be inserted. However, one pervasive problem arising in PVD-based methods is that they frequently yield pixel values out of the valid interval. This is known as the *falling-off-boundary problem* (also FOBP). These pixels are either adjusted or ignored by the method, thus reducing the number of pixels available to carry message data.[3,18]

2.1. *Tri-way pixel-value differencing*

The TPVD method was designed to get more pixels involved in the data embedding process.[3] The TPVD divides the carrier image into nonoverlapping blocks of 2×2 consecutive pixels. Three difference values are computed in each block using the values of two neighbor pixels in three different directions. The first difference is computed between the pixel in the upper left corner, called the pivot, and the pixel

on its right. The second difference is between the pivot and the pixel in the opposite corner, and the third one is also between the pivot and the pixel below it. Each difference is then matched with one of an already defined set of intervals usually designed as a table. Every interval determines the amount of data bits to be hidden into the pixel block being examined.

2.1.1. *Designing the intervals table*

The intervals table is commonly designed by the experimenter, either looking to achieve a high capacity or a low perceptibility.[33] However, other alternatives are available to replace this table.[6,30] For example, if we follow the approach from Hernández-Servín *et al.*,[6] a particular interval table can be modeled with just one equation, for instance, a table with the intervals $[0,7]$, $[8,15]$, $[16,31]$, $[32,63]$, $[64,127]$, $[127,191]$, and $[192,255]$ could be represented with the following untidy equation:

$$\phi(z) = \lfloor \log_2(z + 8 * H(8 - z) - 64 * H(z - 128) - 128 * H(z - 192) + 64 * H(z - 192)) \rfloor, \tag{1}$$

where z is the difference of each pixel pair and H is a step function as described in Hernández-Servín *et al.*[6] Nevertheless, even when a more neat looking equation can be written, it is evident that some designing steps from the experimenter side are still necessary to produce such equation.

Whether an intervals table or an equation is used, the TPVD algorithm follows these steps (steps 2 and 3 could be readily replaced by Eq. (1)):

(1) Compute the differences $d_i = p_i - p_1, i \in \{2,3,4\}$ of the pixel block being examined.
(2) Locate for each d_i the interval k such that $l_k \leq |d_i| \leq u_k$. l_k and u_k are the lower an upper values of the interval.
(3) Compute the amount of input data bits t_i to be inserted as follows:

$$t_i = \begin{cases} 0 & \text{if } i = 1 \\ \lfloor \log_2(u_k - l_k + 1) \rfloor & \text{otherwise,} \end{cases} \tag{2}$$

(4) Compute the decimal representation b_i of the t_i bits.
(5) A new d'_i is computed for each d_i

$$d'_i = l_{k_i} + b_i, \tag{3}$$

where l_{k_i} is the lower value of the matching interval of d'_i.
(6) Then the TPVD uses each d'_i to compute new pixel values for the block being examined using a well-crafted set of rules.[3] These new pixels hold within their difference part of the secret message data.

We have modified the TPVD by replacing the rules of the last step with an *optimization* strategy to determine the best pixel values that will hold the maximum

secret message. Furthermore, this modification fixes the FOBP produced by the Chang *et al.*[3] TPVD algorithm.

3. An Optimization Strategy to Improve the TPVD

A closer look to the TPVD, shows that the FOBP is constantly present and that invalid pixel values are not used as data carriers. TPVD authors[3] do not seem to discuss how the extraction algorithm knows which pixels are being ignored.[6] This is fundamental to guarantee the integrity of the secret message.

Any PVD method can be seen as an optimization problem as follows: Given d'_i and p_i, search for a solution p'_i subject to the following set of conditions:

(1) The FOBP must be prevented subject to $0 \leq p'_i \leq 255$.
(2) Retrieving the message data is subject to $d'_i = |p'_i - p'_1|$, where p'_i and p'_1 are now variables to be searched as an optimization problem which will define the stego-image.
(3) Distortion of the resulting image must be subject to minimize the objective function

$$f(p_i, p'_i) = \sum_{i=1}^{4} (p_i - p'_i)^2. \tag{4}$$

We know that $p'_i = |d'_i|$ is a solution, i.e. $p_1 = 0$, that fulfills conditions 1 and 2, but does not fulfill condition 3 because it causes a major distortion to the resulting stego-image. Nonetheless, the solution shows that there exist at least one solution for any given input.

Since there are four pixels per block each in the range $[0, \ldots, 255]$, we can easily estimate the size of the search space to be 2^{32} possible pixel value combinations times the carrier image dimensions divided by 4. These number of solutions would take too long to be explored efficiently.

One alternative is to reduce the size of the search space so that it can be readily explored. Using equation from condition (2) it follows that

$$p'_i = \pm d'_i + p'_1. \tag{5}$$

This evidently means that we can compute p'_i using the following two variables:

(1) $\pm d'_i$ takes the different sign combinations for d'_i. These combinations are 8 because d'_1 is always 0 and d'_2, d'_3, d'_4 only can take two different values: one positive and one negative of equal magnitude.
(2) p'_1 must be subject to $0 \leq p'_1 \leq 255$. This means that p'_1 only can take 256 distinct values.

This reduces the size of the search space to 2^{11}. A search space of this size can be readily explored in its entirety. That is, all possible values for p'_1 must be combined with all possible values for $\pm d'_i$.

3.1. *An additional optimization strategy*

We now describe an additional optimization strategy to further increase the amount of message inserted by the method from Sec. 3. This strategy is based on the first derivative of the objective function with respect to p'_1 and discards the solutions incurring in the FOBP.

Using Eqs. (4) and (5), a quadratic function can be produced in terms of p'_1, namely:

$$f(p'_1) = \sum_{i=1}^{4} (\pm d_i - p'_1 + p_i)^2. \tag{6}$$

Eight different quadratic curves can be plotted from the eight different combinations of signs in $\pm d_i$. When computing the first derivative of these functions, a point for each curve can be found for which f is minimum:

$$p'_1 = \frac{1}{4} \sum_{i=1}^{4} p_i - \frac{1}{4} \sum_{i=1}^{4} \pm d_i. \tag{7}$$

The resulting eight candidate values for p'_1 can become 16 because Eq. (7) can yield real numbers that need to be converted into integers using either the *ceil* or *floor* functions.

In some cases, the optimal point can be outside the valid interval or can even cause some of the other three pixels to be off. It is necessary then to move that point to the proper interval as that value is potentially a solution.

Figure 1 shows two curves plotted using the objective function. These curves are bounded between a pair of dotted lines representing the upper and lower bounds valid for p'_1. It also shows that the points of minimum value are not always within the valid interval and is necessary to move that point to a valid area.

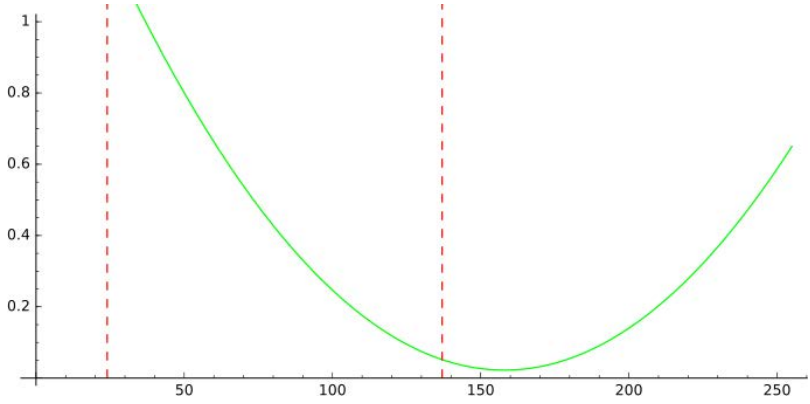
Equation (5) can yield valid intervals for each curve as $M = \max(\pm d_i) \leq p'_1 \leq m = \min(\pm d_i + 255)$. From this equation, we can define the adjustment function:

$$A(p, M, m) = \begin{cases} 0 & \text{if } M \leq p \leq m \\ M - p & \text{if } p < M \\ -(p - m) & \text{if } p > m. \end{cases} \tag{8}$$

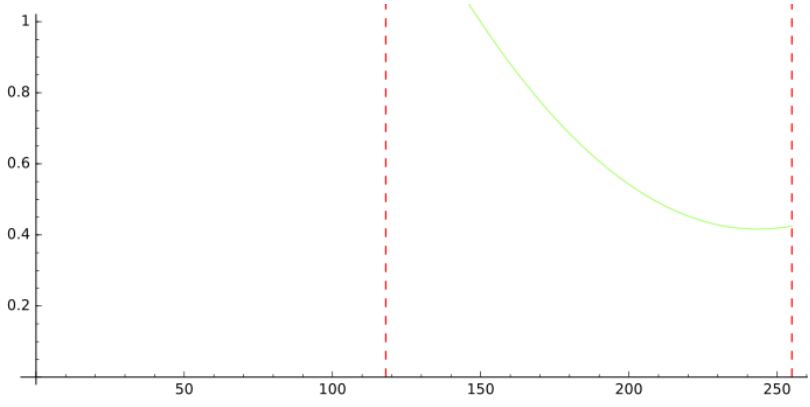
Therefore the optimal point in the valid interval would be defined as:

$$p'_1 = p'_1 + A(p'_1, \max(\pm d_i), \min(\pm d_i) + 255). \tag{9}$$

As mentioned before, this point needs to be adjusted using the *ceil* or *floor* functions. Both functions yield an identical or extremely close value. Because there are eight curves each with two solutions, we end up with a new search space of only 16 potential solutions.



(a) Optimal point outside the valid interval



(b) Optimal point inside the valid interval

Fig. 1. Two different objective function graphs.

The algorithm for the analysis above explained, follows these steps:

(1) Go through steps 1–5 of the algorithm from Sec. 2.1.
(2) Compute $s_i = \pm d_i + p'_1$ using the *ceil* or *floor* functions.
(3) The optimal solution is given by $p'_i = \min(f(p_i, s_i))$.
(4) Replace the original 2×2 pixel block with the optimal solution found.
(5) Repeat from step 1 for each 2×2 pixel block of the carrier image.

To recover the secret message, the inverse process is applied as follows:

(1) Divide the carrier image into nonoverlapping blocks of 2×2 consecutive pixels.
(2) Compute the differences $d_i = p_i - p_1, i \in \{2, 3, 4\}$ of the pixel block being examined.
(3) For each d_i, locate the table interval $r_i = k$ such that $l_k \leq |d_i| \leq u_k$.

(4) Compute the number of inserted bits in each difference

$$t_i = \begin{cases} 0 & \text{if } i = 1 \\ \lfloor \log_2(u_{r_i} - l_{r_i} + 1) \rfloor & \text{otherwise.} \end{cases}$$

(5) The entire message data is recovered by concatenating the binary representation of $b_i = d_i - l_{r_i}$.

3.2. *Inserting an extra bit*

The method can add an additional bit to further increase the secret message inserted in each 2×2 block with a minimal deterioration to the carrier image.

The *floor* and *ceil* functions yield two consecutive integer numbers that produce very close or even identical objective function results. These results constantly appear and are used as indication for inserting an additional bit of the secret message. This additional bit is called β. If $\beta = 0$, p_1' must be even, if $\beta = 1$, p_1' must be odd.

To find the optimal value, we say that $2c = p_1' - \beta$ and modify Eq. (5) as follows:

$$p_i' = \pm d_i + 2c + \beta, \tag{10}$$

$$f(c) = \sum_{i=1}^{4} (\pm d_i + 2c + \beta - p_i)^2. \tag{11}$$

Therefore, the valid interval for the optimization problem is given by:

$$c = \frac{1}{8} \sum_{i=1}^{4} p_i - \frac{1}{8} \sum_{i=1}^{4} \pm d_i - \frac{1}{2}\beta, \tag{12}$$

$$c = c + A\left(c, \max\left(-\frac{\beta}{2} - \frac{1}{2}(\pm d_i)\right), \min\left(-\frac{\beta}{2} - \frac{1}{2} \pm (d_i) + 255\right)\right). \tag{13}$$

The algorithm is also modified as follows:

(1) Go through steps 1–5 of the algorithm from Sec. 2.1.
(2) Compute $s_i = \pm d_i + c$ using the *ceil* or *floor* functions.
(3) The optimal solution is given by $p_i' = \min(f(p_i, s_i))$.
(4) Replace the original 2×2 pixel block with the optimal solution found.
(5) Repeat from step 1 for each pixel block of the carrier image.

To recover the message data, the same steps from Sec. 3.1 are used, and an extra 0 bit is added to the message if p_1 is even or a 1 otherwise.

4. Experimental Results

An extensive experimental evaluation was carried out to test the performance of our algorithms and to compare our results to those previously published in the literature. All test images are 8-bit grayscale images of size 512×512. These images have

Fig. 2. Original images (first row). Resulting stego images using the Optimal-TPVD (second row). Resulting stego images using the OTPVD and Extra Bit Insertion (third row).

become a *de facto* standard in Image Processing and Computer Vision experiments for testing new developments. Firstly, we have chosen a small set of only five images, shown in Fig. 2, in order to fairly compare our results with previous work by Peng *et al.*[21] and Hernández-Servín *et al.*[6] Both authors, in turn, compared their own results with work previously published. In addition, we also compare the performance of our algorithm with the results of the TPVD.[3] Secondly, we also present experimental results using a larger set of 30 images shown in Fig. 3.

The PSNR is used to measure the difference between the original carrier image and the image with the secret message data. The higher the PSNR, the better the quality of the stego image. The number of bits per pixel (bpp) for each test image, is computed simply by dividing the number of bits inserted by the number of pixels in the carrier image.

Table 1 shows a comparison between the Optimal-TPVD and the Extra Bit Insertion algorithms. While the former shows a better performance than recent results by Peng *et al.*[21] and Hernández-Servín *et al.*,[6] the latter further increases the overall results in terms of both the amount of data message inserted (i.e. the *bpp*), and the image distortion measured with the PSNR in all images tested. This results are somewhat expected as both the Optimal TPVD and the Extra Bit Insertion strategies use every 2×2 block to carry data payload. However, no pixel block is ignored and no off-valued pixel ever occurred.

We also compare our results with those from the TPVD[3] in Table 2. Since our algorithms search for the optimal pixel values for each block, the results are superior in terms of both data carried (*bpp*) and stego image quality (PSNR). The general

Fig. 3. A larger set of test images used in our experiments. These images show a wide variability of contrast and pixel intensity.

Table 1. Comparison between both our proposals Optimal-TPVD and Extra Bit OTPVD, and Hernández-Servín *et al.* (HS) and Peng *et al.* (Pg).

	bpp				PSNR			
	OTPVD	EOTPVD	HS[6]	Pg[21]	OTPVD	EOTPVD	HS[6]	Pg[21]
Barbara	2.54	2.79	1.38	1.20	36.50	36.43	36.04	30.75
Airplane	2.37	2.62	1.30	1.20	38.90	38.76	36.09	33.45
Boat	2.41	2.66	1.80	1.20	38.19	38.09	34.56	26.66
Goldhill	2.38	2.63	1.66	1.20	38.73	38.64	37.03	30.70
Lena	2.35	2.60	1.60	1.20	39.34	39.17	37.55	26.89
Average	2.41	2.66	1.55	1.20	38.33	38.22	36.25	29.69

Table 2. Comparison between the TPVD and a our Optimal-TPVD. The latter takes longer as it performs a larger number of operations to determine the best result.

	bpp		PSNR		Time	
	TPVD	OTPVD	TPVD	OTPVD	TPVD	OTPVD
Barbara	2.54	2.54	36.38	36.50	1.55	9.56
Airplane	2.37	2.37	38.23	38.90	1.90	10.02
Boat	2.40	2.41	37.72	38.19	1.88	8.35
Goldhill	2.38	2.38	38.09	38.73	1.84	17.72
Lena	2.35	2.35	38.61	39.34	2.19	11.80
Average	2.41	2.41	37.81	38.33	1.87	11.49

Table 3. Comparison results between both our proposals Optimal-TPVD and Extra Bit OTPVD and the TPVD using a larger set of 30 images.

	PSNR			bpp		
	OTPVD	EOTPVD	TPVD	OTPVD	EOTPVD	TPVD
Airfield	36.426	36.310	36.599	2.542	2.792	2.424
Airfield2	36.580	36.490	36.535	2.535	2.785	2.513
B2	41.340	41.056	39.950	2.272	2.522	2.236
Baboon	34.236	34.232	34.515	2.803	3.053	2.782
Bank	38.260	38.084	38.278	2.395	2.645	2.275
Barb	36.499	36.425	36.381	2.541	2.791	2.535
Bear1	37.905	37.777	37.483	2.438	2.688	2.428
Bear2	39.372	39.202	38.560	2.352	2.602	2.342
Blackwatch	32.036	32.026	32.674	3.193	3.443	3.112
Bridge	35.837	35.754	35.758	2.618	2.868	2.584
Chess	36.839	36.738	36.946	2.497	2.747	2.426
Couple	37.890	37.793	37.491	2.433	2.683	2.420
Crowd	38.448	38.352	37.913	2.399	2.649	2.369
Einstein	38.067	37.976	37.568	2.432	2.682	2.427
F16	38.897	38.756	38.280	2.369	2.619	2.366
Fishingboat	38.193	38.086	37.745	2.408	2.658	2.403
Foxnoise	31.310	31.222	32.631	3.382	3.632	2.917

Table 3. (*Continued*)

	PSNR			bpp		
	OTPVD	EOTPVD	TPVD	OTPVD	EOTPVD	TPVD
Fruit	39.260	39.157	38.486	2.357	2.607	2.356
Fruits	37.344	36.729	41.164	2.374	2.624	0.751
Girl	41.653	41.414	40.109	2.263	2.513	2.209
Girl512	39.859	39.590	39.316	2.324	2.574	2.181
Goldhill	38.733	38.635	38.116	2.382	2.632	2.382
Lax	36.145	36.101	36.112	2.571	2.821	2.562
Lena	39.344	39.166	38.596	2.349	2.599	2.348
Man	38.600	38.497	38.019	2.390	2.640	2.389
Peppers	38.494	38.271	38.552	2.378	2.628	2.210
Sailboat	37.645	37.541	37.329	2.444	2.694	2.441
Splash	40.627	40.415	39.607	2.289	2.539	2.283
Tiffany	39.689	39.495	39.101	2.331	2.581	2.275
Umas	37.867	37.760	37.500	2.438	2.688	2.424
Average	37.774	37.635	37.577	2.483	2.733	2.379

notion is that less data embedded should result in less distortion of the carrier image, which is not observed by comparing the PSNR values of our experiments.

Additional comparison results between both our proposals: Optimal-TPVD and Extra Bit OTPVD, and the TPVD using a larger set of 30 images is shown in Table 3. This table also shows favorable results in terms of both data carried and stego image quality.

5. Conclusions

This work has discussed an optimization strategy that modifies and improves the TPVD[3] steganographic method. It has been favorably compared against the TPVD and also against recent results by Peng *et al.*[21] and Hernandez-Servin *et al.*[6]

Our results show improvements in several important aspects, namely, (i) a higher capacity, (ii) a better stego-image quality, (iii) the falling-off boundary problem is completely overcome, and (iv) no blocks of pixels are ignored as secret data carriers.

PVD methods provide high embedding capacity as well as high imperceptibility for the stego-images. The optimal TPVD method enhances security and the quality of the resulting image and avoids the falling-off boundary problem. The major merit of our algorithms is to reduce the feasible set of possible pixel values for each block such that the search for the best solution in terms of both secret message and stego-image quality can be efficiently conducted.

After reviewing the wealth of PVD variations reported in the literature, there are a couple of directions in which PVD methods research may proceed. The first step is to use the method to hide data into color images aiming to achieve a higher image quality. This has already been tried with mixed results, as the effects in color and

general image distortion may either require to adapt or entirely change PVD algorithms to both increase capacity and imperceptibility.

Also, PVD methods using various difference directions need to increase their imperceptibility and test their strength against stego-analysis attacks. Most published techniques show advantages only in terms of capacity, a common practice in the field, but rarely show any results when subject to stego-analysis methods.

One more challenging research direction is in the design of the intervals table that determines the number of bits to be embedded. This table is always designed by the experimenter either to achieve a large capacity or a high imperceptibility. This set of intervals has a major influence on the performance of any steganography PVD technique and has been showed that it can be replaced by a single function, although the designing skills of the experimenter are still needed. However, an open research question is whether a general method for computing the amount of secret information to be inserted can be designed, such that both a higher capacity and a high imperceptibility can readily be achieved.

References

1. M. A. F. Al-Husainy, Image steganography by mapping pixels to letters, *J. Comput. Sci.* **5**(1) (2009) 33–38.
2. C. Balasubramanian, S. Selvakumar and S. Geetha, High payload image steganography with reduced distortion using octonary pixel pairing scheme, *Multimedia Tools Appl.* **73**(3) (2014) 2223–2245.
3. K. C. Chang, C. P. Chang, P. S. Huang and T. M. Tu, A novel image steganographic method using tri-way pixel-value differencing, *J. Multimed.* **3**(2) (2008) 37–44.
4. C. C. Chang, J. C. Chuang and Y. C. Hu, Spatial domain image hiding scheme using pixel-values differencing, *Fundam. Inform.* **70**(3) (2006) 171–184.
5. A. Cheddad, J. Condell, K. Curran and P. M. Kevitt, Digital image steganography: Survey and analysis of current methods, *Signal Process.* **90** (March 2010) 727–752.
6. J. A. Hernández-Servín, J. R. Marcial-Romero, V. Muñoz-Jiménez and H. A. Montes-Venegas, A modification of the TPVD algorithm for data embedding, in *Proc. of the Mexican Conference on Pattern Recognition*, eds. J. Carrasco-Ochoa, J. Martinez-Trinidad, J. Sossa-Azuela, J. Olvera López and F. Famili (Springer, 2015).
7. Y. K. Jain, A novel image steganography method with adaptive number of least significant bits modification based on private stego-keys, *Int. J. Comput. Sci. Secur. (IJCSS)* **4** (March 2010) 40–49.
8. J. C. Joo, H. Y. Lee and H. K. Lee, Improved steganographic method preserving pixel-value differencing histogram with modulus function, *EURASIP J. Adv. Signal Process.* **2010** (April 2010).
9. M. Juneja and P. Sandhu, Implementation of improved steganographic technique for 24-bit bitmap images in communication, *J. Am. Sci.* **5**(2) (2009) 36–42.
10. M. Kharrazi, H. T. Sencar and N. Memon, Performance study of common image steganography and steganalysis techniques, *J. Electron. Imaging.* **15**(4) (2006) 041104.
11. M. Khatirinejad and P. Lisonek, Linear codes for high payload steganography, *Appl. Math.* **157**(5) (2009) 971–981.
12. Y. P. Lee, J. C. Lee, W. K. Chena, K. C. Chang, I. J. Su and C. P. Chang, High-payload image hiding with quality recovery using tri-way pixel-value differencing, *Inf. Sci.* **191** (May 2012) 214–225.

13. X. Liao, Q. Y. Wen and J. Zhang, A steganographic method for digital images with four-pixel differencing and modified lsb substitution, *J. Vis. Commun. Image Represent.* **22**(1) (2011) 1–8.
14. C. Lin and N. Hsueh, A lossless data hiding scheme based on three-pixel block differences, *Pattern Recognit.* **41** (2008) 1415–1425.
15. W. Luo, F. Huang and J. Huang, A more secure steganography based on adaptive pixel-value differencing scheme, *Multimedia Tools Appl.* **52**(2) (2011) 407–430.
16. M. Mahajan and N. Kaur, Adaptive steganography: A survey of recent statistical aware steganography techniques, *Int. J. Comput. Netw. Inf. Sec.* **4** (September 2012) 76–92.
17. J. K. Mandal and D. Das, Color image steganography based on pixel value differencing in spatial domain, *Int. J. Inf. Sci. Tech.* **2**(4) (2012) 83–93.
18. J. K. Mandal and D. Das, Steganography using adaptive pixel value differencing (apvd) of gray images through exclusion of overflow/underflow, *CoRR* **abs/1205.6775** (2012).
19. M. Mishra, P. Mishra and M. C. Adhikary, Digital image data hiding techniques: A comparative study, *ANVESA* **7**(2) (2012) 105–115.
20. V. Nagaraj, V. Vijayalakshmi and G. Zayaraz, Overview of digital steganography methods and its applications, *Int. J. Adv. Sci. Technol.* **60**(1) (2013) 45–58.
21. F. Peng, X. Li and B. Yang, Adaptive reversible data hiding scheme based on integer transform, *Signal Process.* **92**(1) (2012) 54–62.
22. H. R. Pous and J. Rifa, Product perfect codes and steganography, *Digit. Signal Process.* **19** (July 2009) 764–769.
23. A. Pradhan, D. Sharma and G. Swain, Variable rate steganography in digital images using two, three and four neighbor pixels, *Indian J. Comput. Sci. Eng. (IJCSE)* **3** (July 2012) 457–463.
24. A. K. Sahu and G. Swain, A review on lsb substitution and PVD based image steganography techniques, *Indonesian J. Electr. Eng. Comput. Sci.* **2** (June 2016) 712–719.
25. J. Salunkhe and S. Sirsikar, Pixel value differencing a steganographic method: A survey, *IJCA Proceedings on International Conference on Recent Trends in Engineering and Technology 2013*, ICRTET(5)(2013) pp. 1–6.
26. G. Swain, Steganography in digital images using maximum difference of neighboring pixel values, *Int. J. Security Appl.* **7**(6) (2013) 285–294.
27. G. Swain, Adaptive pixel value differencing steganography using both vertical and horizontal edges, *Multimedia Tools Appl.* **75** (November 2016) 13541–13556.
28. G. Swain and S. K. Lenka, Classification of image steganography techniques in spatial domain: A study, *Int. J. Comput. Sci. Eng. Technol. (IJCSET)* **5**(3) (2014) 219–232.
29. N. Tiwari and M. Shandilya, Secure RGB image steganography from pixel indicator to triple algorithm-an incremental growth, *Int. J. Security Appl.* **4**(4) (2010) 53–62.
30. H. W. Tseng and H. S. Leng, A steganographic method based on pixel-value differencing and the perfect square number, *J. Appl. Math.* **2013**(1) (2013) 1–8.
31. R. Z. Wang and Y. S. Chen, High-payload image steganography using two-way block matching, *IEEE Signal Process. Lett.* **13**(3) (2006) 161–164.
32. C. M. Wang, N. I. Wu, C. S. Tsai and M. S. Hwang, A high quality steganographic method with pixel-value differencing and modulus function, *J. Syst. Softw.* **81** (January 2008) 150–158.
33. D. C. Wu and W. H. Tsai, A steganographic method for images by pixel-value differencing, *Pattern Recognit. Lett.* **24**(9) (2003) 1613–1626.
34. M. Y. Wu, Y. K. Ho and J. H. Lee, An iterative method of palette-based image steganography, *Pattern Recognit. Lett.* **25** (February 2004) 301–309.

35. 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.* **152**(5) (2005) 611–615.
36. Y. H. Yu, C. C. Chang and Y. C. Hu, Hiding secret data in images via predicting coding, *Pattern Recognit.* **38**(5) (2005) 691–705.
37. X. Zhang and S. Wang, Vulnerability of pixel value differencing steganography to histogram analysis and modification for enhanced security, *Pattern Recognit. Lett.* **25** (February 2004) 331–339.
38. X. Zhang, S. Wang and Z. Zhou, Multi-bit assignment steganography in palette images, *IEEE Signal Process. Lett.* **15**(1) (2008) 553–556.

Ismael R. Grajeda-Marín received his B.Sc. degree in Computer Science from the Universidad Autónoma del Estado de México in 2017. His research interests are computer vision, image processing and artificial intelligence.

Héctor A. Montes-Venegas is currently a full time research professor at the Universidad Autónoma del Estado de México. He received his B. Sc. degree in Computer Science from the ITL, Mexico and an M.Sc. degree from the ITESM, Mexico. His research interests include computer vision, image processing, robotics and artificial intelligence.

J. Raymundo Marcial-Romero received his Ph. D. in Computer Science from Birmingham University, UK in 2005. In the last 3 years, he has published 23 research conference and journal articles. Currently he is a full-time professor at the Universidad Autónoma del Estado de México.

J. A. Hernández-Servín is currently a full-time professor at the Universidad Autónoma del Estado de México. In 1999, he received his B.Sc. degree in Math & Phisics Science, and an M.Sc. degree in Mathematics in 2001 from the UMSNH (Univ. Aut. San Nicolás de Hidalgo). In 2005, he earned a Ph.D. from the University of Nottingham, UK.

Vianney Muñoz-Jiménez is currently a full time research-professor at the Universidad Autónoma del Estado de México. In 2009, she received her Ph.D. from Paris 13 University, France. Her research interests include computer vision, image processing and video compression.

Guillermo De Ita Luna obtained his Ph.D. in Electrical Engineering from CINVESTAV-IPN, Mexico. He worked for more than 10 years as a developer and consultant in Geographical Information Databases Systems for different enterprises in Mexico. He has done research visits at the University of Chicago, Texas A&M, INAOEP Puebla, and the INRIA Institute in Lille France. Currently, he is a researcher-professor at the School of Computer Sciences in the Benemerita Universidad Autónoma de Puebla, México. His research interests are artificial intelligence, logic symbolic and design of algorithms.