HIGH-PERFORMANCE EMBEDDING IN DUAL IMAGES BASED ON PIXEL VALUE ORDERING METHOD

Cao Thị Luyen[1,*], **Pham Minh Thai**[2]
[1]*University of Transport and Communications, Hanoi, Vietnam*
[2]*University of Economics - Technology for Industries, Hanoi, Vietnam*

ARTICLE INFORMATION

Journal: *Vinh University Journal of Science Natural Science, Engineering and Technology*
p-ISSN: *3030-4563*
e-ISSN: *3030-4180*

Volume: 53
Issue: 4A
****Correspondence:***
caoluyengt@gmail.com

Received: *14 September 2024*
Accepted: *29 October 2024*
Published: *20 December 2024*

Citation:
Cao Thị Luyen, Pham Minh Thai (2024). High-performance embedding in dual images based on Pixel Value Ordering method
Vinh Uni. J. Sci.
Vol. 53 (4A), pp. 36-46
doi: 10.56824/vujs.2024a099a

ABSTRACT

Reversible data hiding (RDH) not only protects information but also extends its applications across various fields. This paper improves the Pixel Value Ordering (PVO) algorithm to enhance embedding capacity in dual images. We developed an optimized embedding method, allowing for higher data embedding performance compared to related works. Experimental results demonstrate that, relative to similar methods, this improvement enhances both embedding capacity and image quality.

Keywords: Reversible data hiding; PVO; dual image.

1. Introduction

In today's digital world, data is frequently exchanged online, making it crucial to protect against hacker attacks. The two primary solutions are encryption and data hiding. Encryption has the drawback that encrypted data is noticeable, while data hiding conceals information within digital media such as images, reducing the likelihood of detection by hackers. Data hiding methods are categorized into two types: irreversible (where only the data can be extracted) and reversible (RDH), which allows both data extraction and original image recovery, making it suitable for sensitive fields like medicine and the military.

Various RDH methods have been proposed, with Tian's difference expansion (DE) method [1] being one of the most notable. This technique embeds data in the difference between two neighboring pixels, requiring an embedded location map for recovery. Improvements to this approach aim to increase capacity [2] and enhance image quality by using larger pixel blocks, broadening the prediction error (PEE), and reducing location map size [3], [4].

Another approach, histogram shifting (HS) by Ni *et al.* [5], embeds data based on the histogram's peak points. However, its embedding capacity is low for images with flat histograms. Subsequent studies have combined DE and HS to improve efficiency by using thresholds and shifting prediction errors to distinguish embedded bits [6], [7].

Many studies have leveraged flat pixel regions to boost embedding capacity and image quality in RDH methods. Li *et al.*'s PVO method [8] divides images into blocks and predicts the largest and smallest pixel values for data embedding. This method excels in image quality and embedding capacity. Peng *et al.*'s improved version, IPVO [9], increases embedding capacity by utilizing pixel position information within blocks.

To address the issue of not embedding bits into the largest or smallest pixels when many pixels are equal, Ou *et al.* introduced the PVO-K method [10], embedding bits into all largest pixels. Weng *et al.* [11] extended PVO-K to GePVO-K, embedding additional bits but facing challenges with pixel variation and location map size, later resolved by Hoa *et al.* [12], [13].

Traditional RDH methods embed data within a single image, limiting embedding capacity. Recently, dual-image RDH techniques have emerged, where a dual image contains data derived from the original image, enhancing both capacity and security. The first method group uses EMD (Exploit Modification Direction), employing a 256 × 256 module function matrix [14]. For each pixel, a matrix position is chosen as the center, defining a square from which two pixels containing information are selected to ensure reversibility.

Chang *et al.* [15] used the MF matrix to embed four bits into pixel pairs within a 5×5 square, enhancing quality by selecting points on two diagonals. To further increase capacity, they developed a new MF matrix to embed three bits per pixel. Lee *et al.* [16] selected two locations around the central pixel pair for embedding, while Lu *et al.* [17] proposed a two-image RDH method with a middle folding strategy (CFS) to reduce distortion and achieve a higher embedding capacity. Luyen *et al.* [18] recently improved Lu *et al.*'s method by expanding the embedding domain, enhancing the capacity and quality of information-containing images.

As noted, RDH methods based on PVO in single images often have low embedding capacity. Recently, Niu *et al.* [19] introduced a PVO-based RDH method for dual images, demonstrating a superior embedding capacity compared to traditional techniques. This study proposes a reversible information-hiding method based on PVO for dual images, improving Niu *et al.*'s [19] technique by embedding 4 bits per block: 2 bits in the largest pixel and 2 in the smallest. Thus, each image block in the proposed method embeds 4 bits, surpassing the maximum 3.5 bits per block achieved in [19].

The remainder of the paper is structured as follows: Section 2 provides an overview of the PVO method and Niu *et al.*'s [19] approach. Section 3 describes the proposed method. Section 4 presents experiments and discussions comparing the proposed method with related techniques regarding embedding capacity and image quality. Finally, conclusions are drawn in Section 5.

2. Related works

2.1. PVO method [8]

The PVO method [8] divides the input image I of size $M \times N$ into non-overlapping sub-blocks of size $m \times n$. These sub-blocks are represented as a sequence of numbers, denoted by: $X = (x_1, \ldots, x_k)$ with $k = m \times n$.

Next, the sequence X is sorted in ascending order, represented by:

$X_\sigma = (x_{\sigma(1)}, \dots, x_{\sigma(k)})$ where $x_{\sigma(1)} \le \cdots \le x_{\sigma(k)}$, $i < j$, if $x_{\sigma(i)} = x_{\sigma(j)}$ then $\sigma_i < \sigma_j$.

The maximum and minimum prediction errors are then calculated as follows:

$dmax = x_{\sigma(k)} - x_{\sigma(k-1)},\ dmin' = x'_{\sigma(1)} - x'_{\sigma(2)}$

Below is the technique for embedding 1 bit into $dmax$; embedding 1 bit into $dmin$ is done similarly to ensure that the order remains unchanged after embedding.

$$x'_{\sigma(k)} = \begin{cases} x_{\sigma(k)} + b, & \text{if } dmax = 1, \\ x_{\sigma(k)} + 1, & \text{if } dmax > 1,\ x'_{\sigma(n)} \\ x_{\sigma(k)}, & \text{if } dmax = 0. \end{cases}$$

In the reverse process, the embedded bit *b* and the original image block are restored as follows:

- Calculate $dmax'$:

$$dmax' = x'_{\sigma(k)} - x'_{\sigma(k-1)}$$

- Restore embedded information and original image:

$$b = dmax' - 1, if\ dmax' \in \{1,2\}$$

$$x_{\sigma(k)} = \begin{cases} x'_{\sigma(k)} - b, & \text{if } dmax' \in \{1,2\}, \\ x'_{\sigma(k)} - 1, & \text{if } dmax' > 2 \\ x'_{\sigma(k)}, & \text{if } dmax' = 0 \end{cases}$$

The array *X* Only changes at the largest (smallest) pixel, while the remaining pixels are preserved.

2.2. Niu's algorithm [19]

Similar to the PVO method, Niu *et al.* divide the original image X of size $M \times N$ into non-overlapping sub-blocks of size $m \times n$. Each block is represented as a sequence, denoted by $X = (x_1, \dots, x_k)$, $k = m \times n$.

Next, the sequence X is sorted in ascending order, denoted by: $X_\sigma = (x_{\sigma(1)}, \dots, x_{\sigma(k)})$ where $x_{\sigma(1)} \le \cdots \le x_{\sigma(k)}$, $i < j$ if $x_{\sigma(i)} = x_{\sigma(j)}$ then $\sigma_i < \sigma_j$.

The maximum and minimum prediction errors are calculated as follows:

$dmax = x_{\sigma(k)} - x_{\sigma(k-1)},\ dmin' = x'_{\sigma(1)} - x'_{\sigma(2)}$

Table 1 below shows the technique for embedding data into the pixel with the maximum value $x_{\sigma(k)}$. Embedding information into $x_{\sigma(1)}$ is performed similarly to ensure the sequence order remains preserved after embedding.

Table 1: *Rules for embedding data at the largest pixel $x_{\sigma(k)}$ of Niu's algorithm*

Case	**Embedding bit**	$\boldsymbol{x'_{\sigma(k)}}$	$\boldsymbol{x''_{\sigma(k)}}$
If $dmax \in \{0,1\}$			
1	0	$x_{\sigma(k)}$	$x_{\sigma(k)}$
2	100	$x_{\sigma(k)} + 1$	$x_{\sigma(k)}$
3	101	$x_{\sigma(k)}$	$x_{\sigma(k)} + 1$
4	110	$x_{\sigma(k)} + 2$	$x_{\sigma(n)}$
5	111	$x_{\sigma(k)}$	$x_{\sigma(k)} + 2$

Case	**Embedding bit**	$\boldsymbol{x'_{\sigma(k)}}$	$\boldsymbol{x''_{\sigma(k)}}$
If $dmax > 1$			
1	0	$x_{\sigma(k)}$	$x_{\sigma(k)}$
2	10	$x_{\sigma(k)} + 1$	$x_{\sigma(k)}$
3	11	$x_{\sigma(k)}$	$x_{\sigma(k)} + 1$

Comments:

- Min$\{x'_{\sigma(k)},\ x''_{\sigma(k)}\} = x_{\sigma(k)}$. Therefore, it is easy to recover the pixel with the largest value.

- The order of the two subsequences X' and X'' obtained from X after data embedding, preserves the sorting order. Consequently, the original image block can be fully restored.

- The embedding capacity for a block is 2 bits when $dmax \in \{0,1\}$ and 1.5 bits when $dmax > 1$. The total embedding capacity for one block is therefore 3.5 bits. If divided across two output images, the embedding capacity for each sub-image block is 1.75 bits.

We propose a method to embed 2 bits in each of the minimum (or maximum) value pixels while still preserving the PVO properties. As a result, the embedding capacity of the proposed algorithm is 4 bits per image block, which is higher than that of the original algorithm. Below are the details of the proposed algorithm.

3. Proposed algorithm

3.1. Embedding algorithm

Input: Host image I, secret data *B*, sub-block size $m \times n$

Output: Stego images I' and I''.

Steps:

1. Divide the image: Divide the image into non-overlapping sub-blocks of size $m \times n$.

2. **Create a marking map:** Create a marking map for the blocks that will embed data. Sub-image blocks that do not contain pixels with values outside $\{0, 1, 2, 253, 254, 255\}$ are marked, then compressed using the arithmetic encoding algorithm. This compressed map is used in the image restoration process.

3. Embed 4 bits: Embed 4 bits, with 2 bits in the pixel with the maximum value and 2 bits in the pixel with the minimum value, as follows:

Step 3.1: Represent the sub-block as a sequence of pixels, denoted by $X = (x_1, \dots, x_k)$ with $k = m \times n$.

Step 3.2: Sort X in ascending order, denoted as $X_\sigma = (x_{\sigma(1)}, \dots, x_{\sigma(k)})$ where $x_{\sigma(1)} \leq \cdots \leq x_{\sigma(k)}$; if $i < j$ and $x_{\sigma(i)} = x_{\sigma(j)}$, then $\sigma_i < \sigma_j$.

Step 3.3. Embed data by embedding the bit pair $b_1 b_2$ into the pixel with the largest value $x_{\sigma(k)}$ and $b_3 b_4$ into the pixel with the smallest value $x_{\sigma(1)}$, according to Table 2 below.

Table 2: *Embedding rules of the proposed algorithm*

$\boldsymbol{b_1 b_2}$	$\boldsymbol{x'_{\sigma(k)}}$	$\boldsymbol{x''_{\sigma(k)}}$	$\boldsymbol{x''_{\sigma(k)} - x'_{\sigma(k)}}$	$\boldsymbol{x'_{\sigma(1)}}$	$\boldsymbol{x''_{\sigma(1)}}$	$\boldsymbol{x''_{\sigma(1)} - x'_{\sigma(1)}}$
00	$x_{\sigma(k)}$	$x_{\sigma(k)}$	0	$x_{\sigma(1)}$	$x_{\sigma(1)}$	0
01	$x_{\sigma(k)} + 1$	$x_{\sigma(k)}$	-1	$x_{\sigma(1)} - 1$	$x_{\sigma(1)}$	1
10	$x_{\sigma(k)}$	$x_{\sigma(k)} + 1$	1	$x_{\sigma(1)}$	$x_{\sigma(1)} - 1$	-1
11	$x_{\sigma(k)} + 2$	$x_{\sigma(k)}$	-2	$x_{\sigma(1)} - 2$	$x_{\sigma(1)}$	2

The result is the obtained sub-sequences $X_\sigma{}' = (x'_{\sigma(1)}, \cdots, x'_{\sigma(k)})$ and $X_\sigma{}'' = (x''_{\sigma(1)}, \cdots, x''_{\sigma(k)})$.

Step 3.4: Re-represent $X_\sigma{}'$ and $X_\sigma{}''$ corresponding to the sub-blocks of the output stego images I' and I''.

4. Repeat Step 3 until all data has been embedded, resulting in I' and I''.

3.2. Data extraction algorithm

Input: The stego images I' and I''.

Output: The original image I and the extracted secret data B.

Steps:

1. Extract auxiliary information, including the block size$m \times n$, tthe size of the compression map, and the compression map itself. Decompress the compression map to obtain the location map *LM*.

2. Partition both stego images I' and I'' into non-overlapping blocks of size $m \times n$. Represent each block as a numerical sequence denoted by $X' = (x'_1, \cdots, x'_k)$ and $X'' = (x''_1, \cdots, x''_k)$, respectively. Based on the location map, if a sub-block is marked as used for embedding, proceed to recover the original image and embedded data in Step 3.

3. Data recovery

Step 3.1: Sort the elements of the subsequences $X' = (x'_1, \cdots, x'_k)$ and $X'' = (x''_1, \cdots, x''_k)$ in ascending order to obtain the sequences: $X_\sigma{}' = (x'_{\sigma(1)}, \cdots, x'_{\sigma(k)})$ and $X_\sigma{}'' = (x''_{\sigma(1)}, \cdots, x''_{\sigma(k)})$.

Step 3.2: Extract the embedded data according to the rule specified in Table 3.

Step 3.3: Restore the pixels with the maximum and minimum values using the formulas:

$$x_{\sigma(1)} = \max\{x'_{\sigma(1)}, x''_{\sigma(1)}\}, \qquad x_{\sigma(k)} = \min\{x'_{\sigma(k)}, x''_{\sigma(k)}\}$$

As other elements are unaffected, recovering the original image sub-block is straightforward. Thus, $X_\sigma = (x_{\sigma(1)}, \ldots, x_{\sigma(k)})$.

Step 3.4: Rewrite X_σ to obtain the subsequence X corresponding to the sub-block of the original image.

4. Iterate Step 3 until all embedded data blocks are traversed, yielding the embedded secret data string B and the original image *X*.

Table 3: *The recovery rule of the proposed algorithm*

$\boldsymbol{x''_{\sigma(k)} - x'_{\sigma(k)}}$	$\boldsymbol{b_1}b_2$	$\boldsymbol{x''_{\sigma(1)} - x'_{\sigma(1)}}$	$\boldsymbol{b}_3b_4$
0	00	0	00
-1	01	1	01
1	10	-1	10
-2	11	2	11

3.3. Example

Assuming the secret information to be embedded is 1001 into the sub-block of the following image, the embedding process is as follows:

55	56
57	55

1. Represent the image sub-block as a sequence to obtain the array (55, 56, 57, 55).
2. Sort the array in ascending order to get (55, 55, 56, 57).
3. At the pixel with the maximum value $x_{\sigma(k)} = 57$, embed the bit pair 10. According to Table 2, the values $x'_{\sigma(k)}$ and $x''_{\sigma(k)}$ will be 57 and 58, respectively.
4. At the pixel with the minimum value $x_{\sigma(1)} = 55$, embed the bit pair 01. According to Table 2, the values $x'_{\sigma(1)}$ and $x''_{\sigma(1)}$ will be 54 and 55, respectively.

The result yields two subsequences of the output image: $X_\sigma{}' = (54, 55, 56, 57)$, $X_\sigma{}'' = (55, 55, 56, 58)$.

By rewriting the two subsequences above, we obtain: $X' = (54, 57, 56, 55)$, $X'' = (55, 58, 56, 55)$.

Represent the subsequences $X'\ and\ X''$ in matrix form to obtain the two corresponding image blocks of I' and I''.

54	56
57	55

55	56
58	55

Recovery process:

The input consists of two image blocks I' and I''.

54	56
57	55

55	56
58	55

1. Represent I' and I'' as the sequences X' and X'', respectively:
$X' = (54, 57, 56, 55)$, $X'' = (55, 58, 56, 55)$.
2. Sort arrays X' and X''in ascending order to get $X_\sigma{}'$ and $X_\sigma{}''$:
$X_\sigma{}' = (54, 55, 56, 57)$, $X_\sigma{}'' = (55, 55, 56, 58)$.

Use Table 3 to recover the embedded information at the pixel pairs with the maximum and minimum values:

- Since $x''_{\sigma(k)} - x'_{\sigma(k)}$=1, the recovered information is 10.
- Since $x''_{\sigma(1)} - x'_{\sigma(1)} = 1$, the recovered information is 01.

Thus, the embedded data in these image blocks is 1001.

4. Restore the original pixel at the pixel pair with the maximum value:

$x_{\sigma(k)} = min\{x''_{\sigma(k)}, x'_{\sigma(k)}\} = 57.$

5. Restore the original pixel at the pixel pair with the minimum value:

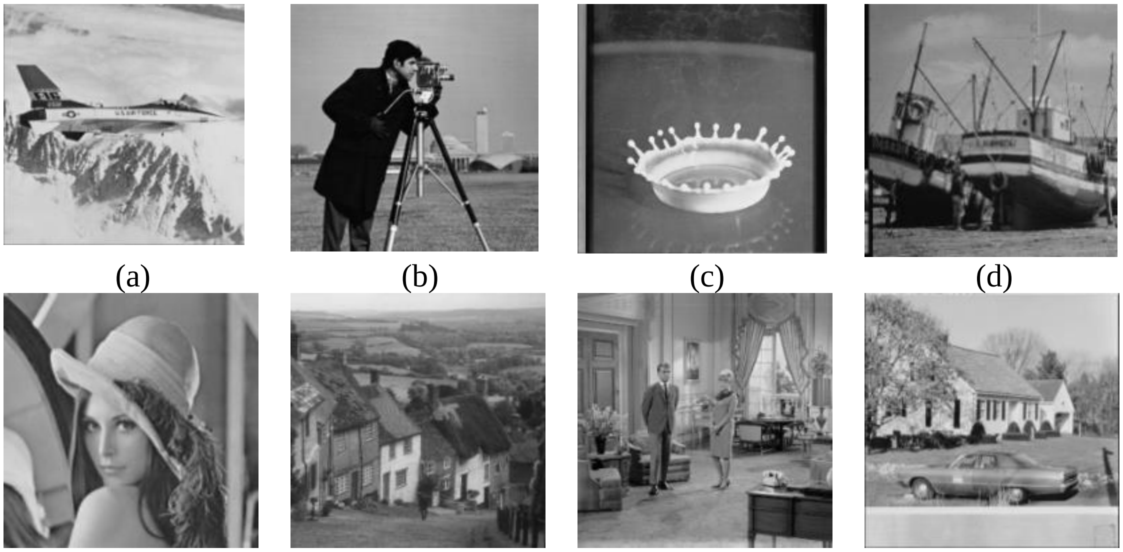
$x_{\sigma(1)} = max\{x''_{\sigma(1)}, x'_{\sigma(1)}\} = 55.$

This gives $X_{\sigma} = (55, 55, 56, 57).$

6. After sorting X_{σ}, we obtain the sequence X=(55, 56, 57, 55), which corresponds to the original image block.

4. Experimental results

We compared the embedding capacity and image quality of the proposed algorithm against related schemes - PVO [8], PVOK [10], GePVOK [11], and Niu [19] - through experiments conducted using R2019a software and 512x512 grayscale images, as shown in Figure 1.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 1: *Test images*

As presented in Table 4, the proposed algorithm outperforms the other methods in terms of embedding capacity. The table displays the number of bits that can be embedded into each image using algorithms with a block size of 2. It can be observed that our method achieves a significantly higher embedding capacity.

Table 4: *Comparison of Embedding Capacity*

Test image	[8]	[10]	[11]	[19]	Proposed Algorithm
(a)	18350	28836	49807	231410	262144
(b)	13107	23593	39322	226870	262140
(c)	18350	31457	47186	232690	262144
(d)	13107	18350	28836	220580	262016

Test image	[8]	[10]	[11]	[19]	Proposed Algorithm
(e)	15729	23593	47186	227170	262144
(f)	10486	15729	20972	216960	261388
(g)	13107	18350	23593	217010	262144
(h)	15729	23593	34079	223480	262144
Average	14746	22938	36372	224521	**262033**

The proposed algorithm demonstrates a substantially higher embedding capacity compared to the related schemes, aligning with theoretical analysis. Each block can be embedded up to 4 bits, compared to 3.5 bits in [19].

Next, we evaluate the image quality of the algorithms by embedding the same amount of secret information into the test images. We use the Peak Signal-to-Noise Ratio (PSNR) as the quality metric, and the results are shown in Table 5.

Table 5: *PSNR value for embedding 10000 random bits*

Test image	[8]	[10]	[11]	[19]	Proposed Algorithm
(a)	59.11	58.33	56.32	69.18	69.80
(b)	57.23	57.51	56.19	68.89	69.80
(c)	58.31	57.24	56.55	68.89	69.80
(d)	58.56	57.69	56.97	68.48	69.80
(e)	63.31	60.5	57.47	67.76	69.80
(f)	58.42	59.28	59.61	68.23	69.80
(g)	55.35	55.45	54.32	69.11	69.80
(h)	58.49	58.81	57.72	68.09	69.80
Average	58.52	58.10	56.89	68.58	**69.80**

PSNR is calculated using the following formula:

$$PSNR = l0log_{10}\left(\frac{255}{\sqrt{\frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (I(i,j) - I'(i,j))^2}}\right)$$

where *I*, I' represent the original image and the image containing embedded data, respectively, with dimensions *M* rows and *N* columns.

In general, a higher *PSNR* indicates better image quality. Table 5 shows that our proposed algorithm achieves a marginally higher *PSNR* than Niu's scheme while significantly outperforming the other competing methods. This finding underscores the superior image quality preservation capabilities of our algorithm.

To validate the robustness of the proposed method, we conducted an extensive evaluation involving 20 trials with random data sizes starting from 100,000 bits. The results, summarized in Table 6, provide compelling evidence of the proposed algorithm's effectiveness compared to the algorithm in [19]. The results in Table 6 indicate that the image quality of the proposed algorithm is consistently good and slightly better than that of the algorithm in [19].

Table 6: *Comparing the image quality of the proposed scheme with [19]*

Test image	**[19]**	**Proposed Algorithm**	**[19]**	**Proposed Algorithm**	**[19]**	**Proposed Algorithm**
	100000 bits		*150000 bits*		*200000 bits*	
(a)	58.667	59.764	56.923	58.012	55.619	56.776
(b)	58.654	59.764	56.986	58.012	55.697	56.776
(c)	58.815	59.764	56.954	58.012	55.667	56.776
(d)	58.667	59.764	56.218	58.012	55.784	56.776
(e)	58.011	59.764	56.656	58.012	55.649	56.776
(f)	59.049	59.764	57.302	58.012	56.055	56.776
(g)	58.939	59.764	57.364	58.012	56.201	56.776
(h)	58.811	59.764	57.121	58.012	55.882	56.776
Average	58.702	**59.764**	56.941	**58.012**	55.895	**56.776**

5. Conclusion

Experimental results and analysis have shown that our proposed reversible data hiding method, which leverages Pixel Value Ordering (PVO), outperforms existing techniques. The proposed method achieves a significant increase in embedding capacity while maintaining high image quality, making it suitable for applications requiring both security and data integrity. This optimized approach is expected to contribute substantially to the advancement of state-of-the-art steganography, with real-world applications across diverse domains.

Acknowledgments: This research is funded by the University of Transport and Communications (UTC) under grant number T2024-CN-005.

REFERENCES

[1] J. Tian, “Reversible data embedding using a difference expansion,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890-896, 2003. DOI: 10.1109/TCSVT.2003.815962

[2] A. Alattar, “Reversible watermark using the difference expansion of a generalized integer transform,” *IEEE Trans. Image Process.*, vol. 13, no. 8, pp. 1147-1156, 2004. DOI: 10.1109/TIP.2004.828418

[3] D. M. Thodi and J. J. Rodríguez, “Expansion embedding techniques for reversible watermarking,” *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721-730, 2007. DOI: 10.1109/TIP.2006.891046

[4] I. C. Dragoi and D. Coltuc, “Local-prediction-based difference expansion reversible watermarking,” *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1779-1790, 2014. DOI: 10.1109/TIP.2014.2307482

[5] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, “Reversible data hiding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354-362, 2006. DOI: 10.1109/TCSVT.2006.869964

[6] C. C. Lin, W. L. Tai, and C. C. Chang, “Multilevel reversible data hiding based on histogram modification of difference images,” *Pattern Recogn.*, vol. 41, pp. 3582-3591, 2008. DOI: 10.1016/j.patcog.2008.05.015

[7] Y. C. Li, C. M. Yeh, and C. C. Chang, “Data hiding based on the similarity between neighboring pixels with reversibility,” *Digit. Signal Process.*, vol. 20, pp. 1116-1128, 2013. DOI: 10.1016/j.dsp.2009.10.025

[8] X. Li, J. Li, B. Li, and B. Yang, “High fidelity reversible data hiding scheme based on pixel value ordering and prediction error expansion,” *Signal Process.*, vol. 93, pp. 198-205, 2013. DOI: 10.1016/j.sigpro.2012.07.025

[9] F. Peng, X. Li, and B. Yang, “Improved PVO-based reversible data hiding,” *Digit. Signal Process.*, vol. 25, pp. 255-265, 2014. DOI: 10.1016/j.dsp.2013.11.002

[10] X. Qu and H. J. Kim, “Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding,” *Signal Process.*, vol. 111, pp. 249-260, 2015. DOI: 10.1016/j.sigpro.2015.01.002

[11] J. J. Li, Y. H. Wu, C. F. Lee, and C. C. Chang, “Generalized PVO-K Embedding Technique for Reversible Data Hiding,” *Int. J. Netw. Secur.*, vol. 20, pp. 65-77, 2018.

[12] L. Q. Hoa, C. T. Luyen, N. K. Sao, and P. V. At, “An Improved Reversible Watermarking Based on Pixel Value Ordering and Prediction Error Expansion,” *Lecture Notes in Computer Science*, vol. 12034, pp. 582–592, 2020. DOI: 10.1007/978-3-030-42058-1_49

[13] Z. Pan and E. Gao, “A reversible data hiding based on novel embedding structure PVO and adaptive block-merging strategy,” *Multimedia Tools Appl.*, 2019. DOI: 10.1007/s11042-019-7692-3

[14] C. C. Chang, T. D. Kieu, and Y. C. Chou, “Reversible data hiding scheme using two steganographic images,” In *Proc. IEEE Region 10 Int. Conf. (TENCON)*, 2007, pp. 1-4. DOI: 10.1109/TENCON.2007.4483783

[15] X. Chen and C. Hong, “An efficient dual-image reversible data hiding scheme based on exploiting modification direction”, *J. Inf. Secur. Appl.*, vol. 58, pp. 1-9, 2021. DOI: 10.1016/j.jisa.2020.102702

[16] C. F. Lee and Y. L. Huang, “Reversible data hiding scheme based on dual Stegano-images using orientation combinations,” *Telecommun. Syst.*, vol. 52, pp. 2237-2247, 2013. DOI: 10.1007/s11235-011-9529-x

[17] T. C. Lu, J. H. Wu, and C. C. Huang, “Dual-image-based reversible data hiding method using center folding strategy,” *Signal Process.*, vol. 115, pp. 195-213, 2015. DOI: 10.1016/j.sigpro.2015.03.017

[18] C. T. Luyen, N. K. Sao, and P. V. At, "Dual-image reversible data hiding method using maximum embedding ability of each pixel," *J. Inf. Secur. Appl.*, vol. 74, no. 103470, 2023. DOI: 10.1016/j.jisa.2023.103470

[19] Y. Niu and S. Shen, "A novel pixel value ordering reversible data hiding based on dual-image," *Multimedia Tools Appl.*, vol. 81, no. 10, pp. 13751-13771, 2022. DOI: 10.1007/s11042-022-12149-y

TÓM TẮT

MỘT PHƯƠNG PHÁP NHÚNG HIỆU QUẢ CAO CHO ẢNH KÉP DỰA VÀO SẮP XẾP THỨ TỰ GIÁ TRỊ ĐIỂM ẢNH

Cao Thị Luyên[1], Phạm Minh Thái[2]
[1]*Trường Đại học Giao thông Vận tải, Hà Nội, Việt Nam*
[2]*Trường đại học Kinh tế - Kỹ thuật Công nghiệp, Hà Nội, Việt Nam*

Ngày nhận bài 14/9/2024, ngày nhận đăng 29/10/2024

Giấu tin thuận nghịch không chỉ bảo vệ thông tin mà còn mở rộng khả năng ứng dụng trong nhiều lĩnh vực. Bài báo này cải tiến phương pháp sắp xếp thứ tự giá trị điểm ảnh (PVO - Pixel Value Ordering) để tăng khả năng nhúng tin trên ảnh kép. Chúng tôi đã phát triển một phương pháp nhúng tối ưu, cho phép thuật toán nhúng thông tin với hiệu suất cao hơn so với các phương pháp hiện có. Kết quả thực nghiệm cho thấy rằng, so với các phương pháp truyền thống, cải tiến này không chỉ nâng cao khả năng nhúng mà còn bảo toàn chất lượng của ảnh.

Từ khóa: Giấu tin thuận nghịch; PVO; ảnh kép.