AN EFFICIENT REVERSIBLE DATA HIDING BASED ON IMPROVED PIXEL VALUE ORDERING METHOD

Article *in* Journal of Computer Science and Cybernetics · May 2022

DOI: 10.15625/1813-9663/38/2/16880

CITATIONS
0

READS
24

4 authors, including:

Cao Luyen
University of Transport and Communications
14 PUBLICATIONS **9** CITATIONS
SEE PROFILE

Nguyen Kim Sao
University of Transport and Communications
6 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Ta Minh Thanh
Le Quy Don Technical University
107 PUBLICATIONS **254** CITATIONS
SEE PROFILE

Journal of Computer Science and Cybernetics, V.38, N.2 (2022), 165–180
DOI no 10.15625/1813-9663/38/2/16880

AN EFFICIENT REVERSIBLE DATA HIDING BASED ON IMPROVED PIXEL VALUE ORDERING METHOD

CAO THI LUYEN[1,*], NGUYEN KIM SAO[1], LE QUANG HOA[2] AND TA MINH THANH[3]

[1] *University of Transport and Communications, Ha Noi, Viet Nam*
[2] *Institute of Applied Mathematics and Informatics, Hanoi University of Science and Technology, Ha Noi, Viet Nam*
[3] *Le Quy Don Technical University, Ha Noi, Viet Nam*

Abstract. Pixel value ordering (PVO) has been considered as an effective reversible data hiding method for high embedding capacity and good imperceptibility. This paper proposes a novel reversible data hiding based on four sorted pixels for improving the conventional PVO method. Three bits will be embedded into each 4-pixel sub-block without changing the order while the original PVO method only embeds 2 bits. In case the amount of payload is less than the embedding capacity, flatter blocks will be prioritized for embedding to improve image quality. To determine the flat of the block, we use 12 neighborhood pixels of the current block. Only blocks with satisfactory flatness are used for embedding. The proposed reversible data hiding not only retrieves high capacity but also gets good imperceptibility. Experimental results also show that the proposed reversible data hiding scheme outperforms several widely schemes using pixel value ordering method in terms of both image quality and embedding capacity.

Keywords. Reversible data hiding; Pixel value ordering (PVO).

1. INTRODUCTION

1.1. Overview

Nowadays, the security and integrity verification of the data has been a becoming supreme important problem, especially, data is exchanged through the Internet. To protect data from tampering various methods and unauthorized access for data hiding like cryptography, hashing, watermarking and authentication have been researched. This paper relates to special watermarking in the spatial domain that bases on data hiding technique called reversible data hiding (RDH), the original data and the host image are restored to their original state also [6]. RDH is used effectively in health, military, security – defense fields [10, 16, 22]. The main challenges of RDH are embedding capacity, image quality and computational complexity. The primary approaches are listed by time as follows: lossless compression [4, 9, 27], difference expansion (DE) [1, 3, 23, 30], integer transform [5, 7, 29], prediction error expansion (PEE) [5, 8, 28], histogram shifting (HS) [12–15, 24–26, 31, 32] and pixel value ordering (also

*Corresponding author.
E-mail addresses: luyenct@utc.edu.vn (C.T.Luyen); saonkoliver@utc.edu.vn (N.K.Sao); hoa.lequang1@hust.edu.vn (L.Q.Hoa); thanh@mta.utc.edu (T.M.Thanh)

called PVO) [2, 11, 17–21]. If the lossless compression method has low embedding then DE is highly capable based on the difference of two pixels. However, the limitation of DE is the construction of a location map. Integer transform and PEE methods had overcome that drawback to gain higher embedding capacity. Besides DE, HS methods have been seen as the price for good performance, reversible data hiding based on the ordering of pixel values (PVO) has been of great interest. PVO technique reduces pixel changes, contributing to higher image fidelity. So far, some of the improvements of the PVO method have been proposed to improve their embedding capacity and maintain image quality, most authors use good image characteristics such as pixel correlation, block textures and region complexity, etc. PVO method first proposed by Li [18] divides cover image into equal-size noneoverlapping subblocks. Each subblock is sorted in ascending manner. Next, maximum and minimum prediction errors are generated by utilizing pixel value order information to embed one bit into each prediction error that equals one. According to the PVO method, each block can embed 2 bits maximum and change up to 2 units. The bigger the size of block is, the better quality of image is. PVO method performs well in the case of low embedding capacity. The authors of [18] abandoned block that has maximal prediction error or minimal prediction error equal to zero; If data is embedded into the block that has zero prediction error then quality image will be better. Obviously, natural images contain a lot of blocks having zero prediction error. IPVO method in [20] has improved that drawback so both embedding capacity and quality image of IPVO are higher. There is still room to improve IPVO. Block having the largest pixel value and the second largest one the same is ignored by IPVO. This is a waste. In [2] the authors proposed the scheme called PVOK to solve that. PVOK considered K pixels with the largest (or smallest) pixel values as one unit to embed one bit of secret data (PVOK method). Therefore, more blocks can be utilized to embed so the capacity is increased. However, all K pixels are changed to embed only one bit. This cost is quite expensive. Generalized PVOK (also known as GePVOK method) proposed by [17] expanded PVOK in order to embed K bits into K largest (smallest) valued pixels. As a result, the image quality of the GePVOK scheme is greater than the PVOK method. GePVOK uses two bits to build a location map to overcome overflow (or underflow) as well to ensure recovery. Then, compressing the location map gets PM and PM is embedded into the original image. The longer the PM the less embedding capacity for given cover image. In [11] improved GePVOK (called IGePVOK) by using one bit to build location map consequently the length of location map therefore the compression map is less than the GePVOK scheme's. Moreover, the IGePVOK scheme has more blocks used to embed whereas the GePVOK scheme is not so the embedding capacity of IGePVOK is larger than the GePVOK. But, the drawback of IGePVK is imperceptibility, in some cases to embed 1-bit IGePVOK has to change 2 units. In [19] is also an improvement of the IPVO method. The maximum number of embedded bits of the PVO or IPVO method is $2k/(m \times n)$ where $m \times n$ is the subblock size and k is the number of pixels. Thus, the larger the subblock size, the lower the embedding capacity. TPVO recommends the size of subblocks to be 3×1 or 1×3 for more subblocks i.e. higher embedding capacity. In [19] also proposed to hide information into high flatness block, the flatness evaluation is based on 9 pixels of 9 neighborhood blocks of the block in question. In this paper, we will use simple and effective flat block selection criteria to select embedded blocks. We also propose an embedding method so that each subblock with 4 pixels will embed 3 bits to improve embedding ability.

1.2. Our contribution

Among the PVO development methods, the improved PVO (IPVO) scheme is considered a more efficient one both in terms of embedding capacity and good image quality. Our paper tries to improve IPVO to achieve the embedded efficiency and good imperceptibility with the following improvements:

- Firstly, in order to increase the embedding capacity, we propose to divide subblocks with size 4 bits that each embed 3 bits instead of a 2-bit embedding block like the IPVO method. Therefore, our paper can improve the embedding capacity while almost remaining the quality of the embedded images.
- Secondly, to control the quality of the embedded images, we propose the method to select the suitable blocks for secret information embedding. Therefore, we can improve the quality of the image containing the secret information.
- Thirdly, the selected flat blocks for embedding information are found in a simple but effective method, thereby, we also can reduce the computational complexity.

1.3. Roadmap

The rest of the paper is organized as follows: Section 2 describes the related works. The proposed embedding and extracting procedure are presented and proved in Section 3. In Section 4, the experimental results are reported in detail. Final conclusion of this paper is shown in Section 5.

2. RELATED WORKS

2.1. Pixel value ordering method (PVO)

PVO was first proposed by Li *et al.* [18] in 2013 and evaluated as a method with high embedding capacity with good image quality, so there have been many extended schemes based on it. According to PVO, the original image is divided into non-overlapped blocks. Next, for each given block with n pixels $(x_1, x_2, \ldots, x_n)$ is sorted in ascending order to get the sequence $(x'_{\sigma(1)}, x'_{\sigma(2)}, \ldots, x'_{\sigma(n)})$, where $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ is one – one mapping and $x'_{\sigma(1)} \leq x'_{\sigma(2)} \leq \ldots \leq x'_{\sigma(n)})$. Then, the largest prediction error d_{max} and smallest prediction error d_{min} are calculated by the formula below

$$d_{max} = x_{\sigma(n)} - x_{\sigma(n-1)},$$
$$d_{min} = x_{\sigma(1)} - x_{\sigma(2)}.$$

After that, the secret data b is embedded by the following formula

$$d'_{max} = \begin{cases} d_{max}, & \text{if } d_{max} = 0 \\ d_{max} + b, & \text{if } d_{max} = 1 \\ d_{max} + 1, & \text{if } d_{max} > 1, \end{cases} \tag{1}$$

$$d'_{min} = \begin{cases} d_{min}, & \text{if } d_{min} = 0 \\ d_{min} - b, & \text{if } d_{min} = -1 \\ d_{min} - 1, & \text{if } d_{min} < -1. \end{cases} \tag{2}$$

At last, a stego pixel sequence $(x'_{\sigma(1)},\ x'_{\sigma(2)}, \ldots,\ x'_{\sigma(n)})$ is determined as follows

$$\begin{aligned} x'_{\sigma(n)} &= x_{\sigma(n-1)} + d'_{max}, \\ x'_{\sigma(1)} &= x_{\sigma(2)} + d'_{min}. \end{aligned} \tag{3}$$

After applying the above steps, the embedded image is obtained. It is easy to find that, all pixel values in the block remain unchanged in the embedding phase except the largest and smallest pixel values. The largest pixel values may become larger and the smallest pixel values get smaller after applying equation (3). Therefore, the order of pixel values in the block remains unchanged. Based on such a feature, the secret data and the host image can be easily restored.

2.2. Improved PVO scheme (IPVO)

IPVO [20] is an improvement of PVO in that it uses zero prediction error for embedding. Obviously, blocks that have many pixels with the largest or smallest value ($d_{max} = 0$, or $d_{min} = 0$), especially natural images. PVO ignores such blocks while IPVO uses them for information embedding. That is reason embedding capacity of the IPVO method is higher than that of the PVO method.

Similar to the PVO method, the original image is divided into none-overlapped blocks. Next, for each given block with n pixels $(x_1,\ x_2, \ldots,\ x_n)$ is sorted in ascending order to get the sequence $(x'_{\sigma(1)},\ x'_{\sigma(2)}, \ldots,\ x'_{\sigma(n)})$. Then, indexes u and v are calculated by using formula below

$$u = \min(\sigma(n),\ \sigma(n-1)),\ v = \max(\sigma(n),\ \sigma(n-1)).$$

Next, d_{max} is computed as following formula

$$d_{max} = x_u - x_v. \tag{4}$$

Embedding into the largest pixels is computed as formula bellows: if $d_{max} = 1$ or 0 then $x'_{\sigma(n)} = x_{\sigma(n)} + b$ else $x'_{\sigma(n)} = x_{\sigma(n)} + 1$. Embedding into the smallest pixels is determined as formula follows: if $d_{mim} = 1$ or 0 then $x'_{\sigma(n)} = x_{\sigma(n)} - b$ else $x'_{\sigma(n)} = x_{\sigma(n)} - 1$, where $d_{min} = x_s – x_t,\ s = \min(\sigma(1),\ \sigma(2)),\ t = \max(\sigma(1),\ \sigma(2))$.

To solve over/underflow problem, both PVO and IPVO methods use the location map to mark blocks whether embedding or not. The block is marked by 1 if it contains at least a pixel value 0 or equal 255, otherwise, the block is labeled by 0. The blocks with 1 label will be ignored for embedding data because they can cause over/underflow.

2.3. B.Ou et al.' scheme (PVOK)

The PVO scheme ignores blocks with $d_{max} = 0$ (or $d_{min} = 0$). In flat images, especially natural images with many blocks, this property is satisfied, but it is not used for embedding. This limited the embedded capacity. Ou *et al*. (called PVOK) [2] overcomes the above limitation by considering all the largest (or smallest) valued pixels as a unit, consequently. According to the method of PVOK, there are more blocks to be embedded than that in previous schemes. Assume that the current block has K largest pixel values and the ordered sequence is

$$x_{\pi(1)} \leq x_{\pi(2)} \cdot \leq x_{\pi(s-K)} \leq x_{\pi(s-K=+1)} = \cdots = x_{\pi(m\times n)}$$

.

The prediction error of the largest pixels is computed as below formula

$$d_{max} = x_{\pi(m\times n)} - x_{\pi(m\times n-K)}$$

If $K = 1$ then PVOK becomes IPVO. If d_{max} equals 1 or 0 the secret data is hidden into the current block similar to the way of IPVO. That means, K largest valued pixels are modified. Obviously, the block is not used in IPVO but in contrast with PVOK hence the embedding capacity of PVOK is higher than IPVO. However, the image quality of PVOK is not as good as IPVO because the PVOK scheme has to change K bits to embed much more bit information, so this scheme is not appreciated.

2.4. Pan and Gao' scheme (TPVO)

According to PVO as well as IPVO scheme, each subblock can be embedded maximum 2 bits. Thus, the more blocks, the more embedding capacity. In [19], the authors proposed to divide the subblock with size of 3 pixels for that purpose. The scheme used the reversible data hiding technique of the IPVO scheme. That means two bits can be embedded at the minimum and maximum prediction error. In order to improve the image quality, they proposed to hide information into high flatness block. The flatness calculation formula is based on the correlation of 8 adjacent sub-blocks. In addition, TPVO considered merging contiguous sub-blocks which by themselves are not able to embedding, but the block after merging can embed data to improve embedding capacity. The flatness of the block is calculated as follows

$$\delta = \sqrt{\frac{1}{\sum_{i=1}^{8} \frac{1}{d_i}} \times \sum_{i=1}^{8} \frac{1}{d_i}(P_i - P)^2}, \tag{5}$$

where δ is complexity of the current subblock, $P, P_1, \ldots, P_8$ are the value of pixel middle pixels of eight neighborhood blocks as shown in Figure 1, d_i is the Euclidean distance between P_i and P.

Subblock with the size of 3 pixels is represented as vertical type or horizontal type as Figure 2. The correlation of the image block based on the nearest point will gain better interoperability. In this paper, we propose the evaluation method based on the flatness of the block 12 neighboring points. The formula calculates the flatness also simpler methods.

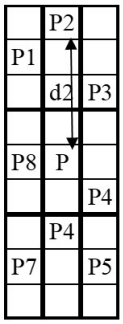

Figure 1: Complexity-computation method

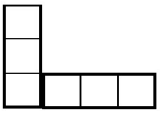

Figure 2: Type of subblock

3. OUR PROPOSED REVERSIBLE DATA HIDING SCHEME

According to PVO method, maximum two bits can be embedded into each block. Therefore, in this method, the larger the block size, the lower the embedding capacity is achieved. If the block has 3×3 pixels of size, the maximum embedding is 2 bits, hence the embedding capacity is 2/3 of image size. Our paper proposes dividing the block size of 4 pixels, combining 1 pixel element with 3 remaining pixels to get 3 pairs, 1 bit can be embedded into each pair. Therefore, there 3 bits can be embedded into one block. that means the embedding capacity of the proposed scheme is 3/4 image size. For images that embed very little secret data compared to the embedding capacity, we select the high flatness blocks for embedding to increase image quality. The next, we propose how to calculate the flatness of a block.

3.1. Determining the flatness of the block

Li et. al. [18] showed that the higher the flatness of the embedding, the better the image quality of the schema. They compute flatness based on the difference between the largest value and the second smallest value in the block. The correlation of a block that has more pixels is better, so we use two rows and one column of the current block to evaluate the flatness instead of two values as [18]. The Figure 3 illustrated the current block flatness (white area). The diagonal area is the neighborhood pixels for judging the flatness of the current block.

Suppose F is the flatness of sub-block I_i (white region), F is calculated using the following

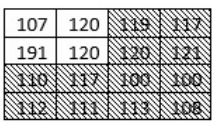

Figure 3: Illustration of the flatness of a block

formula

$$F(I_i) = \sqrt{\sum_{i=1}^{12}(c_i - avg)^2}, \tag{6}$$

where $avg = \sum_{i=1}^{12} c_i$, c_i is pixel in diagonal region.

The diagonal region is called the flatness context of the white region. Because in the nature images, almost adjacent pixels have very small differences, so that, if the flatness context is flat, the visited block has a high flat rate. $F(I_i)$ is a parameter that determines the difference between adjacent pixels. The smaller $F(I_i)$ is, the flatter a block is.

In this paper, we use a threshold to define the flatness of a block. This threshold is calculated such that the cover image can embed all of the secret messages and the threshold is the smallest. This threshold is defined by iterating process determining the condition for which block is selected for embedding. Assuming the loop stops at the value θ, that means, the block I_i will be used to embed the message if $F(I_i) \leq \theta$.

3.2. Determining the location map and compressed location map

In many reversible data hiding methods, a location map is used to determine which blocks are assigned to accommodate secret data and which blocks are ignored.

The blocks that cause underflow/overflow will be skipped and be marked 1 in the location map. So, the location map is constructed as follows

$$LM(I_i) = \begin{cases} 1, & \text{if } \exists\ x_1 \in \{0, 1, 254, 255\}, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

So, the length of the location map is equal to the number of blocks of the image. To decrease the size of the location map, a compressing algorithm is used. In this paper, we use the Run Length algorithm for compressing the location map.

This algorithm is very simple, first, counting the consecutive alike elements, then we get an array that contains these numbers for each change. Choosing the max number of this array to define the length of each changed element. The compressed location map includes the first bit and the above number array in max-bit. For example $x = [11000011111]$ we will get an array $y = [245]$, the max number is 5, 5 is presented in three bits, the first bit is 1, the compressed array of x is 1010100101.

In this paper, we improve the Run Length compress algorithm in cases:

Case 1: x has only one value 0 (or 1), we only marked this case, and the size of the compressed location map is zero.

Case 2: x has only one different bit in the first or bottom, we only marked this case, and the size of the compress location map is zero.

Case 3: x has one different bit in the middle of x, we only save the location of the different bit.

In other cases, we follow the algorithm that is explained above.

When applying Run-length algorithm on some images in Section 4, the length of compress location map of (a), (b), (c),(d), (f), (g) equal zero, and (e), (h) equal 51. So that, with these examples, the length of compress location map is very small if compare with capacity.

3.3. Embedding and extracting algorithm in a block

3.3.1. Embedding in a block

Input: A original block with sized 2×2, three secret bits. Output: A stego block. Firstly, sorting I_i to get an array denoted by $(x_{\pi(1)}, x_{\pi(2)}, x_{\pi(3)}, x_{\pi(4)})$. Next, we survey three pairs $(x_{\pi(1)}, x_{\pi(3)})$, $(x_{\pi(2)}, x_{\pi(3)})$, $(x_{\pi(3)}, x_{\pi(4)})$ for embedding.

Case 1: Embedding secret data into $(x_{\pi(1)}, x_{\pi(3)})$, $(x_{\pi(2)}, x_{\pi(3)})$. For simplicity we present the embedding into the pair $(x_{\pi(1)}, x_{\pi(3)})$, embedding into pixel pair $(x_{\pi(2)}, x_{\pi(3)})$ is performed similar. Error prediction is computed by using the following formula

$$d = x_{\pi(1)} - x_{\pi(3)}.$$

Case 1.1: If $d < -1$ then decreasing $x_{\pi(1)}$ one unit and no secret data is hidden.

Case 1.2: If $d = -1$ and $(\pi(1) > \pi(3))$ then the secret data b is embedded as the formula below

$$x'_{\pi(1)} = x_{\pi(1)} - b.$$

Otherwise

$$x'_{\pi(1)} = x_{\pi(1)} - 1.$$

Case 1.3: If $d = 0$ then the secret data b is embedded as the formula below

$$x'_{\pi(1)} = x_{\pi(1)} - b.$$

Case 2: Embedding the secret data into $(x_{\pi(3)}, x_{\pi(4)})$. Error prediction is computed by using following formula

$$d = x_{\pi(4)} - x_{\pi(3)}.$$

Case 2.1: If $d > 1$ then increase $x_{\pi(4)}$ one unit and no secret data is hidden.

Case 2.2: If $d = 1$ and $(\pi(4) > \pi(3))$ then the secret data b is embedded as the formula below

$$x'_{\pi(4)} = x_{\pi(1)} + b.$$

Otherwise

$$x'_{\pi(4)} = x_{\pi(1)} + 1.$$

Case 2.3: If $d = 0$ then the secret data is embedded as the formula below

$$x'_{\pi(4)} = x_{\pi(4)} + b.$$

Afterward, we get $(x'_{\pi(1)}, x'_{\pi(2)}, x'_{\pi(3)}, x'_{\pi(4)})$ without changing the order in which $x'_{\pi(3)} = x_{\pi(3)}$.

3.3.2. Extracting and restoring in a block

Firstly sorting I'_i to get array denoted by $(x'_{\pi(1)}, x'_{\pi(2)}, x'_{\pi(3)}, x'_{\pi(4)})$. Next, we restore the secret data and the host image as below.

Case 1: Restoring secret data from $(x'_{\pi(1)}, x'_{\pi(3)})$, $(x'_{\pi(2)}, x'_{\pi(3)})$. Restoring the secret data from $(x'_{\pi(1)}, x'_{\pi(3)})$ is similar to restoring from array $(x'_{\pi(2)}, x'_{\pi(3)})$. Restoring the secret

data from $(x'_{\pi(1)}, x'_{\pi(3)})$ is described below. Firstly, error prediction is computed by using the following formula

$$d' = x'_{\pi(1)} - x'_{\pi(3)}.$$

Case 1.1: If $d' < -2$ then the original image is restored as the following formula

$$x_{\pi(1)} = x'_{\pi(1)} + 1.$$

Case 1.2: If $d' = -2$ and $\pi(1) > \pi(3)$ then the secret data b equals to 1, the host image is extracted as formula below

$$x_{\pi(1)} = x'_{\pi(1)} + 1.$$

Otherwise, no secret data is restored and the original image is computed

$$x_{\pi(1)} = x'_{\pi(1)} + 1.$$

Case 1.3: If $d' = -1$ and $(\pi(1) > \pi(3))$ then secret data b equals to 0, the host image is extracted as the formula below

$$x_{\pi(1)} = x'_{\pi(1)}.$$

If $\pi(1) < \pi(3)$ then secret data b equals to 1, the host image is extracted as the formula below

$$x_{\pi(1)} = x'_{\pi(1)} + 1.$$

Case 1.4: If $d' = -1$ then secret data b equals to 0, the host image is extracted as the formula below

$$x_{\pi(1)} = x'_{\pi(1)}.$$

Case 2: Restoring secret data from $(x'_{\pi(3)},\ x'_{\pi(4)})$. Error prediction is computed by using the following formula:

$$d' = x'_{\pi(4)} - x'_{\pi(3)}.$$

Case 2.1: If $d' > 2$ then the original pixel is gotten by decreasing $x'_{\pi(4)}$ one unit and no secret data is hidden.

Case 2.2: If $d' = 2$: If $\pi'(4) > \pi'(3)$ then secret data bit is 1 otherwise we get bit 0. The original pixel is extracted for $d' = 2$ as the formula below

$$x_{\pi(4)} = x'_{\pi(4)} - 1.$$

Case 2.3: If $\pi'(4) > \pi'(3)$ then secret data bit is 1 and original pixel is gotten by decreasing $x'_{\pi(4)}$ one unit. Otherwise, the secret data bit is 0 and original pixel equals to $x'_{\pi(4)}$ $(x_{\pi(4)} = x'_{\pi(4)})$.

Case 2.4: If $d' = 0$ we extract bit 0 and the original pixel is restored as the formula below

$$x'_{\pi(4)} = x_{\pi(4)}.$$

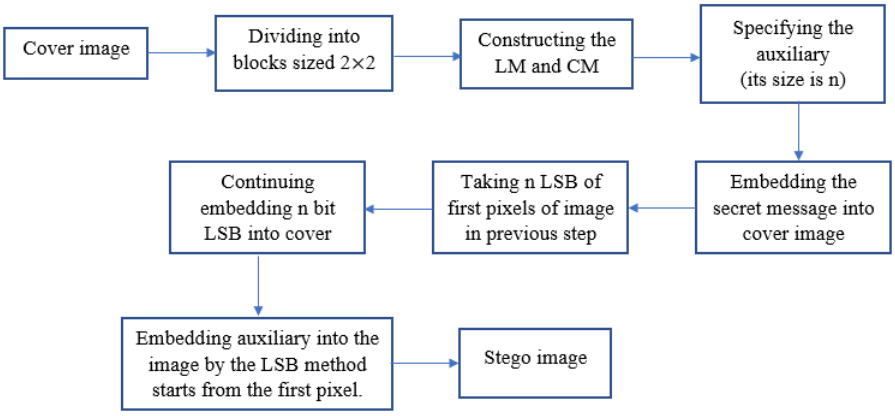

Figure 4: The flowchart of the embedding procedure

3.4. Embedding procedure

This section describes the embedding procedure in detail. Figure 4 shows the process of the embedding procedure.

Input: cover image I, payload B, threshold θ.

Output: stego image I'.

Step 1: Image partition.

We use 2 bottom rows and 2 right columns for context prediction. The remaining original image (called I) is used for embedding as follows.

The cover image I is divided into blocks sized 2×2 from top to bottom and left to right with 4 pixels denoted by $I_i = (x_1, x_2, x_3, x_4)$.

Step 2: Location map construction: Location map LM is built and compressed in a lossless way to get a compressed map denoted by CM based on Subsection 3.2.

Step 3: Specifies the to-be-embed data and auxiliary information: The auxiliary for extracting secret data and restoring original image is in Table 1. Denote $extraLSB = LSBof(16 + 24 + 24 + size(CM))$ $(64 + size(CM))$ pixels which are used for embedding started-block position information, threshold, $size(CM)$ and CM by LSB method. ($size(CM)$ is size of CM). So, the embedded data includes B and $extraLSB$: $P = B + extraLSB$. Note that, the $extraLSB$ must be embedded before B.

Table 1: The auxiliary information

	Content	size
1	The started-block position for embedding (SB) $extraLSB$ (see below)	16
2	The size of CM	24
3	The threshold	24
4	The compress location map	size(CM)

Step 4: Data embedding.

Embedding array of bits P into I by browsing each bock I_i as below:

If $LM(I_i) = 1$, ignore this block, the stego block is the same original block.

If $LM(I_i) = 0$ and I_i is smooth ($(F(I_i) \leq \theta$, θ is computed in Subsection 3.1) then embedding bit by bit from B into I_i blocks according to embedding algorithm in Subsection 3.3.1. Note that save the position of the block which embed the end of B (plus 1 for started-block position for embedding $extraLSB$).

To get $extraLSB$ by taking $(64+size(CM))$ LSB of the first pixels of the image, continue embedding $extraLSB$ into the remaining block until the end of P or all blocks are visited.

Step 5: Embedding all of the auxiliary information starts from the first pixel by the LSB method to get the stego image.

3.5. Recovering procedure

This section describes the extracting procedure in detail. Figure 5 shows the process of the extracting procedure.

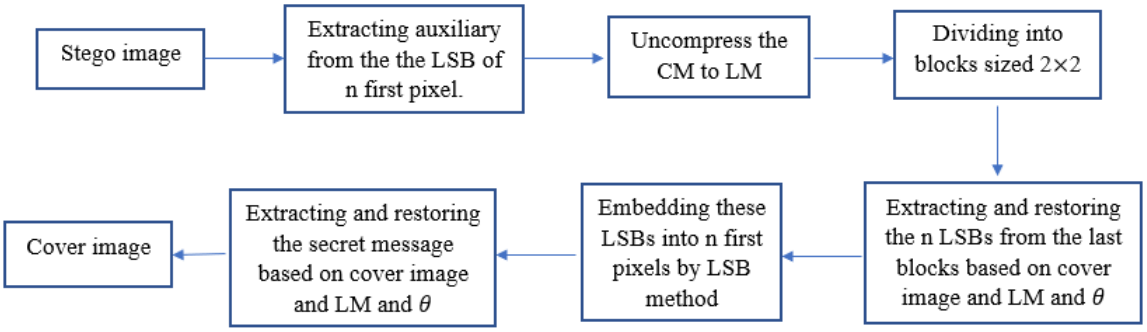

Figure 5: The flowchart of the extracting procedure

Input: Stego image I'.

Output: Cover image I, payload B.

The extracting procedure is an inversion of the embedding process, the details of extracting are shown below.

Step 1: Extracting the auxiliary information from LSB of $(64 + size(CM))$ first pixels to get started-block position information, threshold θ and CM.

Uncompress CM to get LM.

Step 2: Extract secret data and restore the cover image.

Browsing each block I'_i of stego image I' (Note that 2 bottom rows and 2 right columns for context prediction). Extracting the embedded data P and restoring I according to the following algorithm:

First, The stego image is divided into blocks of 4 pixels like the embedding process. From the bottom to top, right to left, with each block I_i, there is a marked value $LM(I'_i)$ in LM.

Visiting block by block, considering correspondingly $LM(I'_i)$ of it.

If $LM(I'_i) = 1$ or $F(I'_i) > \theta$, $x_i = x'_i$ and no secret data is restored.

If $LM(I'_i) = 0$ and I_i is smooth $(F(I'_i) \leq \theta)$ then extra and restore as follow sequence:

Extracting $extraLSB$ first from the bottom block to the SB block according to extracting and restoring algorithm in Subsection 3.3.2. Embedding all $extraLBS$ bits into $(64 + size(CM))$ first pixels of image by LSB method.

Table 2: Comparison in term of the max capacity

Image	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	Average
PVO	10947	7296	8947	21447	48389	24192	32108	13211	20817
IPVO	16553	35799	12584	25149	89366	26735	38713	13730	32329
PVOK	15317	9076	11995	24298	76737	27414	38952	1420	25651
TPES	17748	43129	13948	27061	109374	28975	44476	14441	37394
Proposed	18708	43941	14247	28098	112401	29386	45264	15369	38427

Continuing extracting secret bits and restoring the cover blocks.

At the end of this Step, we get B and I which contain all of restoring blocks.

from I, get the cover image.

Note that, we can completely fix $x_{\pi(2)}$ to replace $x_{\pi(3)}$ in the above reversible data hiding algorithm to investigate embedding for a given image. The fixed point for embedding is higher, then the algorithm will select it for the embedding schema.

4. EXPERIMENTAL RESULTS

We use the images sized 512×512 as in Figure 6 to compare the embedding capacity and image quality of our proposed methods with the related schemes. The maximum embedding capacity is shown in Table 2, the image quality in the case of maximum embedding is expressed in Table 3.

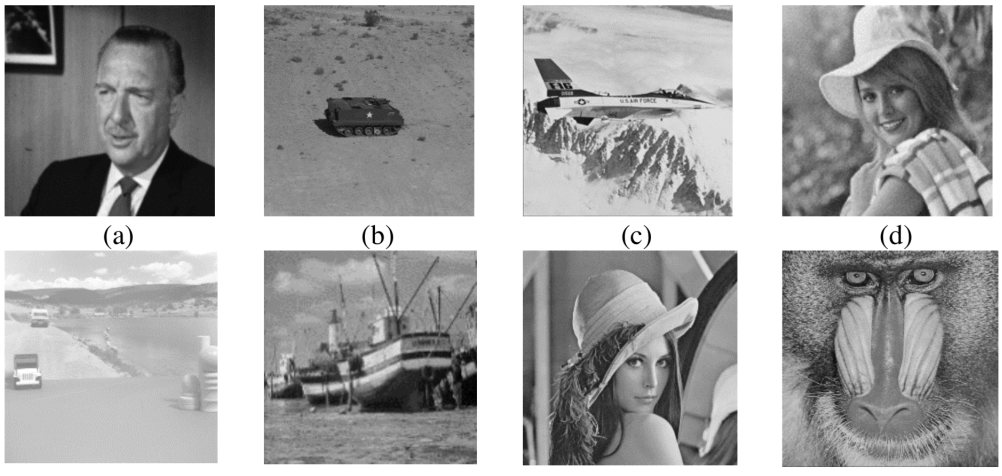

Figure 6: Standard test images

Table 2 shows that the proposed method embeds more 17610, 6098, 12776 , 1033 bits than PVO, IPVO, PVOK and TPES methods respectively. Embedding ability of the proposed is highest in comparison with related works. That means, selecting block size with 2×2 obtains maximal embedding capacity. It is easy to understand because there are more created prediction errors to embed data.In particular, the embedding technique of the proposed

Table 3: PSNR of the watermarked images in maximal embedding case

Image	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	Average
PVO	54.22	52.51	53.14	51.12	55.80	52.15	52.62	51.62	52.90
IPVO	52.49	54.67	52.40	51.58	52.99	51.61	51.84	51.37	52.37
PVOK	51.30	50.24	50.24	51.24	51.35	51.23	51.17	51.15	50.99
TPES	56.42	54.24	47.47	50.51	49.07	50.00	50.546	50.339	51.07
Proposed	50.36	49.86	50.07	49.71	50.87	49.72	53.896	54.136	51.08

method makes it possible to embed 3 bits in a block instead of 2 bits like the previous methods.

Next, we use PSNR to evaluate the imperceptibility of the proposed scheme and several widely PVO based reversible data hiding in maximal embedding case. PSNR is calculated by using the following Equation (8)

$$PSNR = 10 \log_{10} \frac{255^2}{MSE}, \tag{8}$$

where the mean square error (MSE) between the original and embedded image is defined as

$$MSE = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (I(i,j) - I'(i,j))^2, \tag{9}$$

where $M \times N$ is the size of the image, $I(i,j)$ and $I'(i,j)$ are the pixel value at position (i,j) of the host image and the embedded image, respectively.

The experimental result is show in Table 3. The higher PSNR is, the better quality image is. Table 3 shows that, although the number of embedding bits of the proposed scheme is more 17610, 6098, 12776 ,1033 bits than the PVO, IPVO, PVOK and TPES schemes respectively, the image quality of the proposed schema is almost equivalent to the related schemes.

Next, we test the image quality of schemes when embedding the same payload. The test results are shown in the following Figure 7.

The smaller the number of embedded bits, the better the image quality of the proposed scheme is compared to the related schemes. In general, the proposed scheme gains the best image quality in most cases.

5. CONCLUSION

In this paper, we present novel reversible data hiding methods based on pixel value ordering. By dividing the host image into 4-pixel subblocks to create 3 error prediction pairs to embed 3 bits per block. As a result, the embedding capacity of the proposed schemes is higher than the original scheme. In this article, we also use a simple method to evaluate the flatness of blocks. Secret data will be embedded into flatter blocks to improve image quality therefore the proposed scheme not only has higher embedding capacity but also gains better imperceptibility. Experimental results show that the proposed scheme is much higher performance than the related works in terms of both embedding ability and image quality.

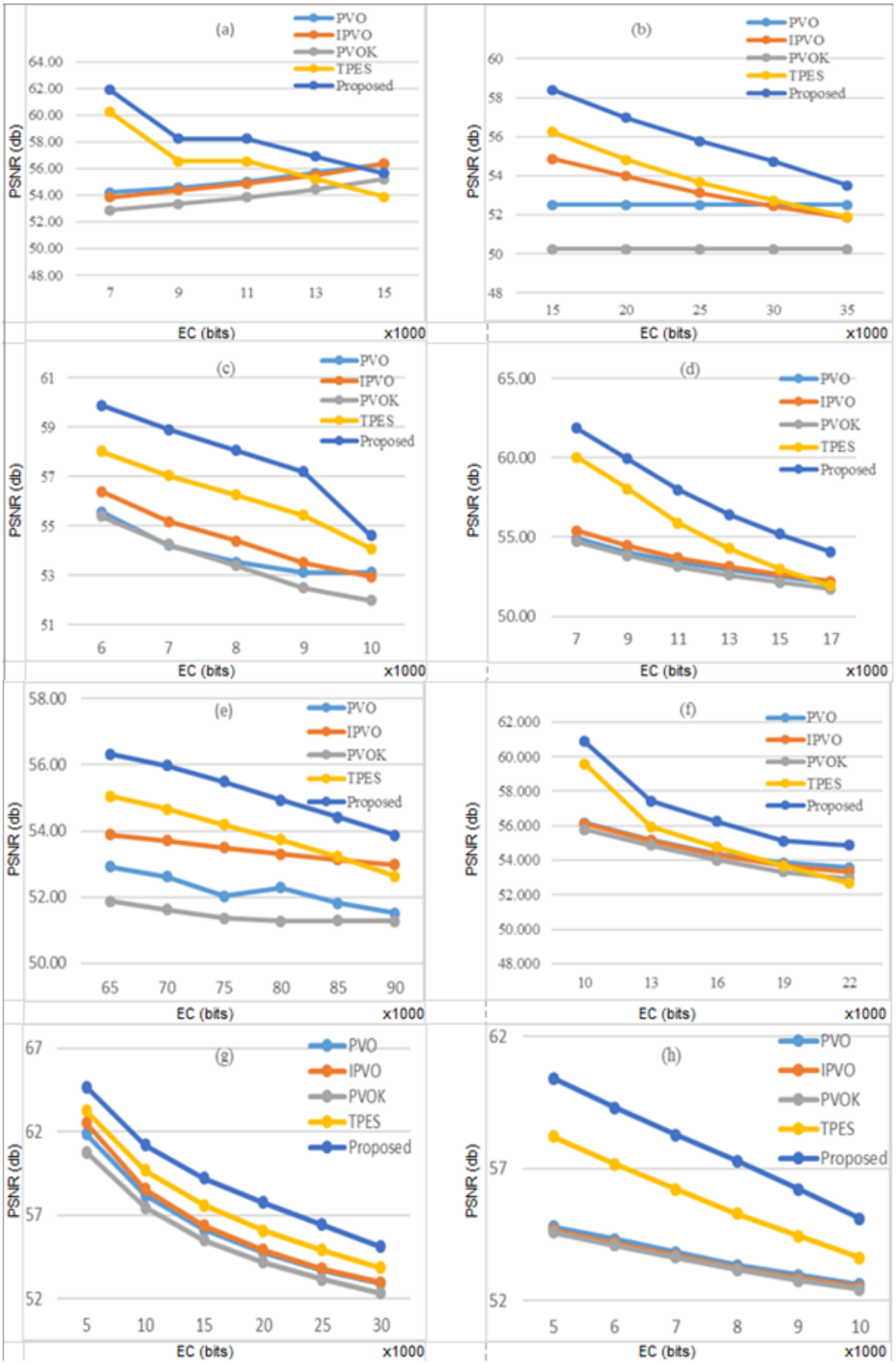

Figure 7: Comparisons in terms of quality image between methods

ACKNOWLEDGEMENTS

This research is funded by University of Transport and Communications (UTC) under grant number T2022-CN-005.

REFERENCES

[1] A.M.Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Transactions on Image Processing*, vol. 13, no. 8, pp. 1147–1156, 2004.

[2] Y. Z. R. N. B. Ou, X. Li, "Reversible data hiding using invariant pixel-value ordering and prediction-error expansion," *Signal Process Image Commun*, vol. 29, no. 7, pp. 760–772, 2014.

[3] P. V. A. Cao Thi Luyen, "An efficient reversible watermarking method and its application in public key fragile watermarking," *Applied Mathematical Sciences*, vol. 11, no. 30, pp. 1481 — 1494, 2017.

[4] M. U. Celik, G. Sharma, and A. M. Tekalp, "Lossless watermarking for image authentication: a new framework and an implementation," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 1042–1049, 2006.

[5] D. Coltuc, "Improved embedding for prediction-based reversible watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 873–882, 2011.

[6] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*. Morgan Kaufmann, 2007.

[7] D.Coltuc, "Low distortion transform for reversible watermarking," *IEEE Signal Process*, vol. 21, no. 1, pp. 412–417, 2012.

[8] J. R. D.M. Thodi, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721–730, 2007.

[9] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding—new paradigm in digital watermarking," *EURASIP Journal on Advances in Signal Processing*, vol. 2002, no. 2, pp. 1–12, 2002.

[10] J.-M. C. C. R. G. Coatrieux, C. Le Guillou, "Reversible watermarking for knowledge digest embedding and reliability control in medical images," *EEE Trans. Inf Technol. Biomed.*, vol. 13, no. 2, p. 158–165, 2009.

[11] L. Q. Hoa, C. T. Luyen, N. K. Sao, P. Van At *et al.*, "An improved reversible watermarking based on pixel value ordering and prediction error expansion," in *Asian Conference on Intelligent Information and Database Systems*. Springer, 2020, pp. 582–592.

[12] W. Hong, "An efficient prediction-and-shifting embedding technique for high quality reversible data hiding," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, pp. 1–12, 2010.

[13] W. Hong, T.-S. Chen, and C.-W. Shiu, "Reversible data hiding for high quality images using modification of prediction errors," *Journal of Systems and Software*, vol. 82, no. 11, pp. 1833–1842, 2009.

[14] J. H. H.T. Wu, "Reversible image watermarking on prediction errors efficient histogram modification," *Signal Processl*, vol. 92, no. 12, pp. 3000–3009, 2012.

[15] ——, "General framework to histogram shifting based reversible data hiding," *Signal Processl*, vol. 22, no. 6, pp. 2181–2191, 2013.

[16] P. C.-H. L. K.L. Chung, Y.H. Huang, "Reversible data hiding-based approach for intra-frame error concealment in h.264/avc," *IEEE Trans. Circuits Syst. Video Technol*, vol. 20, no. 11, p. 1643–1647, 2010.

[17] 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, no. 1, pp. 65–77, 2018.

[18] 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 Processing*, vol. 93, no. 1, pp. 198–205, 2013.

[19] Z. Pan and E. Gao, "Reversible data hiding based on novel embedding structure pvo and adaptive block-merging strategy," *Multimedia Tools and Applications*, vol. 78, no. 18, pp. 26 047–26 071, 2019.

[20] F. Peng, X. Li, and B. Yang, "Improved pvo-based reversible data hiding," *Digital Signal Processing*, vol. 25, pp. 255–265, 2014.

[21] X. Qu and H. J. Kim, "Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding," *Signal Processing*, vol. 111, pp. 249–260, 2015.

[22] Y. H. S.K. Lee, Y.H. Suh, "Reversible image authentication based on watermarking," in *2006 IEEE International Conference on Multimedia and Expo.* IEEE, 2006, pp. 1321—-1324.

[23] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890–896, 2003.

[24] J. N.-S. S. Y. S. V. Sachnev, H.J. Kim, "Reversible watermarking algorithm using sorting and prediction," *IEEE Trans.Circuits Syst Video Technol*, vol. 19, no. 7, pp. 989–999, 2009.

[25] J. C. W. Hong, T.-S. Chen, "Reversible data hiding using delaunay triangulation and selective embedment," *Signal Processing*, vol. 308, pp. 140–154, 2015.

[26] X. L. N. Y. W. Zhang, X. Hu, "Recursive histogram modification establishing equivalency between reversible data hiding and lossless data compression," *IEEE Trans. Image Process*, vol. 22, no. 6, pp. 2775–2785, 2013.

[27] Y. Y. D. T. X. L. X. Gao, L. An, "Lossless data embedding using generalized statistical quantity histogram," *IEEE Trans. Circuits Syst Video Technol*, vol. 20, no. 8, pp. 1061–1070, 2006.

[28] T. Z. X. Li, B. Yang, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Trans.Image Process*, vol. 20, no. 12, pp. 3524–3533, 2011.

[29] B. Y. Z. G. X. Wang, X. Li, "Efficient generalized integer transform for reversible watermarking," *IEEE Signal Process. Lett*, vol. 17, no. 6, pp. 567–570, 2010.

[30] J. L. Y. Hu, H.K. Lee, "De-based reversible data hiding with improved overflow location map," *IEEE Trans. Circuits Syst. Video Technol*, vol. 19, no. 2, pp. 250–260, 2009.

[31] C. L. Y.Y. Tsai, D.S. Tsai, "Reversible data hiding scheme based on neighboring pixel differences," *Digit Signal Process*, vol. 23, no. 3, pp. 919–927, 2013.

[32] N. A. W. S. Z. Ni, Y.Q. Shi, "Reversible data hiding," *IEEE Trans.Circuits Syst Video Technol*, vol. 16, no. 3, pp. 354–362, 2007.

Received on January 17, 2022
Accepted on May 15, 2022