International Journal of Applied Engineering Research ISSN 0973-4562 Volume 14, Number 4 (2019) pp. 997-1005
 https://dx.doi.org/10.37622/IJAER/14.4.2019.997-1005

Pixel Value Ordering with Prediction Error Expansion Based High Fidelity Reversible Data Hiding Scheme

Alok Haldar
Department of Computer Science,
Kharagpur College,West Midnapur,
West Bengal, India.
Satyajit De
Department of Computer Science,
Maheshtala College, Budge Budge Trunck Road,
Kolkata-700141, West Bengal, India.
Biswapati Jana
Department of Computer Science,
Vidyasagar University, West Midnapur,
West Bengal, India.

Abstract
This paper introduced a reversible data hiding method based on pixel value ordering with the prediction-error expansion technique and the average value of end pixels'. A host image is first segmented into non-overlapping sub-blocks of three pixels and ordered them as ascending order. For each sub-block maximum pixel value and the minimum pixel value is predicted by the middle pixel value and also the middle pixel value is predicted by the average of the maximum and minimum pixel values. Then by using prediction-error expansions, we can embed secret bits into maximum pixel and minimum pixel and also by using the average value of these two pixels we can embed secret bit into the middle pixel of every sub-block. All secret bits can be recovered and restored the cover image completely from watermarked image. Experimental result of this scheme demonstrates that the embedding capacity and average PSNR value is larger than another pixel value ordering and prediction error expansion based approach for relatively smooth images. Also, the visual quality of the obtained marked image is better than other Pixel Value Ordering and Prediction Error Expansion based method.

Keywords: Pixel-value ordering, Reversible data hiding, Prediction-error expansion, Average pixel value.

INTRODUCTION

Nowadays, data hiding has been found in different application such as authentication, ownership protection, and secret communication. The reversible data hiding(RDH) scheme is proposed to recover the original content from the marked one without any distortion. Here, reversible data hiding methods are the primary technique of lossless compression [1-6], difference expansion (DE) [7-10], histogram shifting (HS) [11-17], prediction- error expansion (PEE) [18-33], etc. Among them, an interesting research part is to achieve high-level image fidelity my modification of each pixel by at most 1. Recently, Li et al.[31] proposed a novel RDH based on pixel-value-ordering (PVO). For this method, the pixels in a block are sorted in ascending order to get (p_1....p_n). Then, the maximum p_n is predicted by p_{n-1}. Finally, the pixel with prediction-error of p_n-p_{n-1}=1 is embedded with one data bit. Besides in [31], by also considering the minimum p1, i.e. predict p1 using p2, two-bit can be embedded into a block at the same time. The experimental results reported in [31] show that the prediction using sorted pixel values is more accurate than the previous methods. The marked image fidelity can be significantly improved by [31]. Huang et al. applied the utilization rate and histogram shift to a high bit-depth of volume structure on medical images [12]. Ni et al. proposed a new lossless data hiding based on a histogram modification, where the zero or minimum points of the image histogram are utilized [17]. Thodi and Rodriguez proposed a reversible data hiding method using prediction-error expansion. Hu et al. proposed an improved reversible data hiding by reducing the overflow location map. Li et al. Proposed an improvement by using adaptive embedding and pixel selection. Lee et al. proposed a reversible data hiding scheme that is free of location map and a corresponding predictive value is derived from the average of its adjacency pixels to make little bit predictive errors.

Li et al. proposed a reversible data hiding scheme [1] using pixel-value-ordering and prediction-error expansion. After sorting in ascending order of every non-overlapped sub-block of equal sizes, the second maximum or minimum pixel value was used to predict the maximum pixel or minimum pixel value respectively. Then by applying prediction-error expansion technique secret data was embedded. Best result was achieved in this technique by using 2x2 sub-blocks i.e. 4 pixels' sub-block. But for this method, maximum pixels are not used to improve the embedding capacity as well as the image quality for every sub-block and this improvement is done by Jung's method.

Jung's proposed a reversible data hiding scheme [2] using pixel-value-ordering and prediction-error expansion. To improve Li et al's scheme Jung's divide the cover image into three pixels non-overlapped sub-blocks. For each sub-block, sorted the pixels in ascending order and then the second largest pixel value was used to predict the maximum pixel value. Then to embed secret data prediction error expansion was applied into it. As a result, high embedding capacity and

good image quality occurs than Li et al's method. But still there is a space available to improve the embedding capacity as well as the image quality for every sub-block and this improvement is done by the proposed method.

In this paper to improve Jung's method, a new reversible data hiding scheme is proposed for three pixels' sub-blocks based on pixel value ordering with prediction error expansion with an average value of the maximum pixel and minimum pixel values. Three pixels of each sub-blocks are ordered in ascending order and the prediction errors are calculated. One is between the maximum and second maximum pixel value and the other between the second maximum and minimum pixel value. Depending on these prediction errors it is tested that the sub-blocks can embed two/one secret bits. Then by applying pixel-error expansion, two/one secret data is hidden into the sub-blocks. Now by depending on the average pixel value of maximum and minimum pixels another one secret bit is embedded into middle pixel and then all secret bits can be fetched and restore the cover image from watermarked image completely. As a result, embedding capacity and image quality becomes high.

RELATED WORK

Li et al.'s Scheme

Pixel value ordering with Pixel error expansion based reversible data hiding scheme have to be used by Li et al. [1].

The Cover image is divided into non-overlapped equal-sized sub blocks to embed secret bits. In every sub block, sort the pixel values in ascending order and then it uses the second largest pixel value to predict the largest pixel or second smallest pixel value to predict the smallest pixel. Performance of this paper becomes best (For embedding capacity and PSNR value) by considering 2 × 2 size blocks.

Suppose (a_i, a_{i+1}, a_{i+2},a_{i+3}) be a sub-block of sorted pixels of order 2 × 2. At first the prediction error is calculated as $err_2=a_{i+3} - a_{i+2}$ and it is modified to err_2'. This modified error err_2' is used to embed secret bits at the largest pixel.

$$err_2' = \begin{cases} err_2 & \text{if } err_2 = 0 \\ err_2 + p_2 & \text{if } err_2 = 1 \\ err_2 + 1 & \text{if } err_2 > 1 \end{cases} \quad (1)$$

Here the value of the secret bit p_2 is either 0 or 1.

To store a secret bit into the largest pixel a_{i+3} it is changed to a'_{i+3}.

$$a'_{i+3} = \begin{cases} a_{i+3} & \text{if } err_2 = 0 \\ a_{i+3} + p_2 & \text{if } err_2 = 1 \\ a_{i+3} + 1 & \text{if } err_2 > 1 \end{cases} \quad (2)$$

To embed secret bit into smallest pixel a_i calculate the prediction error as $err_1=a_i-a_{i+1}$ and modified it to err_1' as follows.

$$err_1' = \begin{cases} err_1 & \text{if } err_1 = 0 \\ err_1 - p_1 & \text{if } err_1 = -1 \\ err_1 - 1 & \text{if } err_1 < -1 \end{cases} \quad (3)$$

Here the value of the secret bit p_1 is either 0 or 1.

Now change the smallest pixel a_i into a_i' to store secret bit p_1 as follows

$$a_i' = \begin{cases} a_i & \text{if } err_1 = 0 \\ a_i - p_1 & \text{if } err_1 = -1 \\ a_i - 1 & \text{if } err_1 < -1 \end{cases} \quad (4)$$

So, a watermarked block (a'_i, a_{i+1}, a_{i+2}, a'_{i+3}) is generated.

Extraction of this method is the reverse of embedding method. First calculate the prediction error $err_2'=a'_{i+3} - a_{i+2}$ to extract bit from the largest pixel.

If $err_2'=0$ then no secret bit exists at maximum pixel. If $1 \le err_2' \le 2$ then a secret bit exists and it is $p_2=err_2' - 1$ and the original pixel is recovered as $a_{i+3}= a'_{i+3} - p_2$. If $err_2' >2$ then no secret bit exists and the largest pixel is recovered as $a_{i+3}= a'_{i+3} - 1$.

Similarly, to extract secret bit from smallest pixel the prediction error $err_1'=a'_i-a_{i+1}$ is calculated. If $err_1'=0$ then the smallest pixel does not contain any secret bit. If $-2 \le err_1' \le -1$ then the minimum pixel contains a secret bit and it is $p_1= |err_1'| - 1$ and the original smallest pixel is restored as $a_i= a'_i + p_1$.

If $err_1' < -2$ then no secret bit exists and the original minimum pixel is recovered as $a_i= a'_i + 1$.

Jung's method improves the embedding capacity as well as the image quality than this method [2].

Jung's Scheme

Jung introduces another PVO and PEE based reversible data embedding scheme to embed more secret bits with good PSNR value [2]. It divides the cover image into three pixels non-overlapped sub-blocks. Like Li et. al.'s scheme, Jung's sorted every sub-block and using second largest pixel predicts the maximum pixel to embed data.

Consider a three pixels sorted sub block as (a_i, a_{i+1}, a_{i+2}). Now the prediction error $err=a_{i+2} - a_{i+1}$ is calculated and it is modified to err' using the secret bit p.

$$err' = \begin{cases} err & \text{if } err = 0 \\ err + p & \text{if } err = 1 \\ err + 1 & \text{if } err > 1 \end{cases} \quad (5)$$

The secret bit p is embedded into the largest pixel a_{i+2} by changing it to a'_{i+2} as follows

$$a'_{i+2} = \begin{cases} a_{i+2} & \text{if } err = 0 \\ a_{i+2} + p & \text{if } err = 1 \\ a_{i+2} + 1 & \text{if } err > 1 \end{cases} \quad (6)$$

So, the generated watermarked sub-block is (a_i, a_{i+1}, a'_{i+2}).

To extract the secret bit from largest pixel the prediction error is calculated as $err'=a'_{i+2} - a_{i+1}$. Secret bit does not exist into the sub-block if $err'=0$. This Sub-block contains a secret bit into maximum pixel if $1 \le err' \le 2$ and the fetched secret bit is $p = err' - 1$ and the covered image pixel is restored as $a_{i+2}= a'_{i+2} - p$.

Sub-block does not contain any secret bit for $err' > 2$ and the original pixel is restored as $a_{i+2}= a'_{i+2} - 1$.

Since no bits are embedded into minimum pixel and second

minimum pixel therefore still there two spaces are available to improve the embedding capacity and image quality of every sub-block. To improve EC, image quality and average PSNR than previously discussed method a better reversible data hiding scheme is proposed in this paper.

Motivation and Objective

Our main aim is to design a reversible data hiding scheme to hide more secret bits maintaining high PSNR with better image quality using PVO and PEE based technique into the cover image.

Proposed Scheme

In this paper we establish a better reversible watermarking scheme to embed secret bits into all three pixels for the three-pixel non-overlapped sub blocks based on pixel-value ordering with prediction error expansion technique and on average value of smallest and largest pixels. The algorithm for embedding procedures is as follows.

EMBEDDING PROCEDURE

Algorithm for data embedding

The detailed algorithm for embedding secret data is described as follows:

Input: We take a cover image of order x × y and a secret bits' sequence to embed into this image.

Output: A watermarked image of order x × y

Step 1: Input a cover image matrix of order x × y.

Step 2: Now divided this cover image into three pixels non-overlapping sub-blocks.

Step 3: Choose an i^{th} sub-block containing the pixel values as (m_i, m_{i+1}, m_{i+2}).

Step 4: After sorting the pixels values (m_i,m_{i+1},m_{i+2}) in ascending order we get (a_i,a_{i+1},a_{i+2}) (i.e. $a_i \leq a_{i+1} \leq a_{i+2}$).

Step 5: calculate the prediction errors as

$err_2 = a_{i+2} - a_{i+1}$

$err_1 = a_i - a_{i+1}$

Step 6:

[To indicate overflow or underflow or nonexistence of secret bits into sub-blocks we set the location map value to 0 i.e. lp(i)=0. Location map value set to 1 i.e. lp(i)=1 to indicate that secret bits may exists at the largest pixel (a_{i+2}) or at smallest pixel (a_i) or both into the sub-block. Again we set the location map value to 2 i.e. lp(i)=2 when secret bits may exists at the largest pixel (a_{i+2}) or at the smallest pixel (a_i) or both and at the middle pixel (a_{i+1}) where the average value (between smallest and highest pixels of sub-block) is calculated using floor function$(\lfloor . \rfloor)$ in the i^{th} sub-block. Location map value, lp(i)=3, represents that secret bits may exists at all three pixels but the average value is taken using ceiling function$(\lceil . \rceil)$ in the i^{th} sub-block.]

We check the embedding capability of the i^{th} sub-block using the following conditions

i) $a_i = 0$ or $a_{i+2}=255$ [condition for underflow or overflow]

ii) $err_2 \neq 1$ and $err_1 \neq -1$

Sub-blocks have no capability to embed secret data for the above conditions and we skip the sub-blocks with setting lp(i)=0. Set i=i+1 and continue from step 3 otherwise go to step 7.

Step7: Prediction error err_2 is changed to err_2' to insert the secret bit p_2 as follows

$$err_2' = \begin{cases} err_2 & \text{if } err_2 = 0 \\ err_2 + p_2 & \text{if } err_2 = 1 \\ err_2 + 1 & \text{if } err_2 > 1 \end{cases} \quad (7)$$

Step 8: Largest pixel a_{i+2} is changed to a'_{i+2} to store the secret bit p_2 as follows

$$a'_{i+2} = \begin{cases} a_{i+2} & \text{if } err_2 = 0 \\ a_{i+2} + p_2 & \text{if } err_2 = 1 \\ a_{i+2} + 1 & \text{if } err_2 > 1 \end{cases} \quad (8)$$

Step 9: Prediction error err_1 is changed to err_1' to insert secret bit p_1 as follows

$$err_1' = \begin{cases} err_1 & \text{if } err_1 = 0 \\ err_1 - p_1 & \text{if } err_1 = -1 \\ err_1 - 1 & \text{if } err_1 < -1 \end{cases} \quad (9)$$

Step 10: Smallest pixel a_i is changed to a'_i to store a secret bit p_1 as follows

$$a_i' = \begin{cases} a_i & \text{if } err_1 = 0 \\ a_i - p_1 & \text{if } err_1 = -1 \\ a_i - 1 & \text{if } err_1 < -1 \end{cases} \quad (10)$$

Step 11: Predict the middle pixel a_{i+1} by the average value of the changed pixels a'_i and a'_{i+2} to embed the secret bit p_0 and the calculated average value is

$$\text{average} = \left\lfloor \frac{a_i' + a_{i+2}'}{2} \right\rfloor.$$

If average $\neq a_{i+1}$ then go to next step otherwise the middle pixel a_{i+1} is changed to a'_{i+1} to store the secret bit p_0 as follows.

$$a'_{i+1} = \begin{cases} a_{i+1} + p_0 & \text{if } a_{i+1} + p_0 < a'_{i+2} \\ a_{i+1} - p_0 & \text{if } a_{i+1} - p_0 > a'_i \end{cases} \quad (11)$$

If a_{i+1} is unchanged then jump to Step13 otherwise the watermarked i^{th} sub-block becomes (a'_i, a'_{i+1}, a'_{i+2}) and set the value 2 to the location map i.e. lp(i)=2 and continue from Step 14.

Step 12: Another average value is calculated as

$$\text{average} = \left\lceil \frac{a_i' + a_{i+2}'}{2} \right\rceil$$

If average $\neq a_{i+1}$ then continue from step-13 otherwise the middle pixel a_{i+1} is changed to a'_{i+1} to store the secret bit p_0 as follows.

$$a'_{i+1} = \begin{cases} a_{i+1} + p_0 & \text{if } a_{i+1} + p_0 < a'_{i+2} \\ a_{i+1} - p_0 & \text{if } a_{i+1} - p_0 > a'_i \end{cases} \quad (6)$$

If a_{i+1} is effected then the watermarked i^{th} sub-block becomes (a'_i, a'_{i+1}, a'_{i+2}) and set the location map value to 3 i.e. lp(i)=3 and go to Step 14.

Step 13: So, we get the watermarked i^{th} sub-block as (a'_i, a_{i+1}, a'_{i+2}) and set lp(i)=1.

Step 14: The resultant pixel values of watermarked i^{th} sub-

block is stored into its original position and we get (m'_i, m'_{i+1}, m'_{i+2}).

Step 15: Set i=i+1 and continue from step 3 until all data bits are embedded.

Step 16: [Storing of compressed location map with auxiliary information]

Using arithmetic coding location map values is losslessly compressed. Suppose the length of the compressed location map is 'lplen' and the sub-block number containing last secret bit is 'blno'. To store the compressed location map with auxiliary information (i.e. last embedding block number 'blno' and length of the compressed location map 'lplen') we have to replace R=2*$\lceil(\log_2 T)\rceil$ + lplen [where T=total number of pixels in cover image] LSB of image pixels starting from the beginning. For that before replacing we store the least significant bits of R pixels (starting from beginning image pixel) into the remaining non watermarked sub-blocks starting from the block no (blno + 1) using the same procedure as above. Finally, we get a watermarked image.

Numerical examples for data embedding

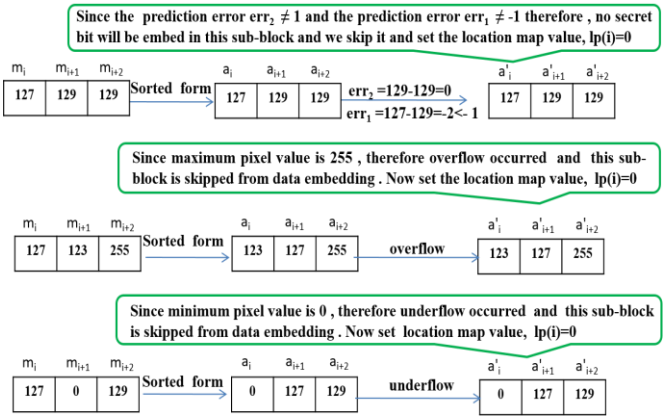

Fig 1: Example of sub-blocks which are not capable to embed secret bits.

In the above examples (i.e. in Fig 1) of sub-blocks since the prediction error $err_1 \neq -1$ or $err_2 \neq 1$ or since the maximum pixel value is 255 i.e. overflow occurred and also underflow occurred therefore these sub-blocks have no capability to embed secret bits, hence skip these blocks from embedding.

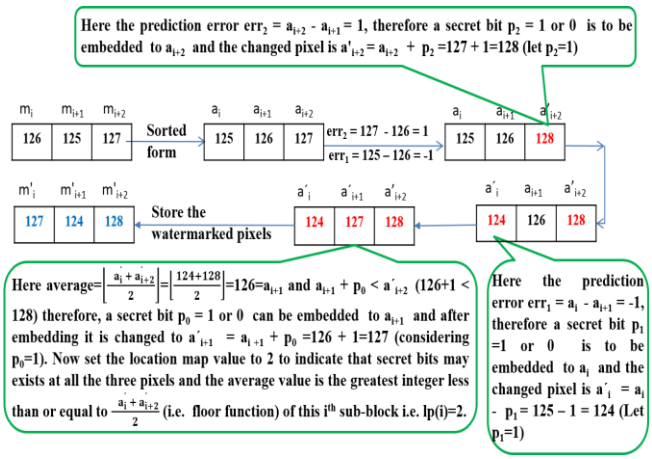

Fig 2: Example of a sub-block which is capable to embed three secret bits into maximum pixel, second maximum pixel and middle pixel with location map value 2.

In Fig 2, consider a i^{th} sub-block of pixel values m_i=126, m_{i+1}= 125, m_{i+2}=127. After sorting them in ascending order we get a_i=125, a_{i+1}=126 and a_{i+2}=127. Calculate the prediction errors $err_2=a_{i+2} - a_{i+1}$=127-126=1 and $err_1=a_i - a_{i+1}$=125–126 = -1. Suppose we want to store secret bits p_1=1 and p_2=1. Since err_2 =1, therefore it is changed to $err_2'=err_2+p_2$ =1+1 =2 and the maximum pixel embed the bit p_2 and it is changed to $a'_{i+2}=a_{i+2} + p_2$=127+1=128. Similarly, since err_1= -1 therefore a_i embed the bit p_1=1 and a_i is changed to $a'_1 = a_i - p_1$ =125 - 1 =124.

Again calculate average=$\left\lfloor \frac{a'_i + a'_{i+2}}{2} \right\rfloor = \left\lfloor \frac{124+128}{2} \right\rfloor$=126=$a_{i+1}$ and $a_{i+1} + p_0 < a'_{i+2}$ (126+1 < 128) therefore, a secret bit p_0 = 1 or 0 can be embedded to a_{i+1} and after embedding it is changed to $a'_{i+1} = a_{i+1} + p_0$ =126 + 1=127 (considering p_0=1). Now set the location map value to 2 to indicate that secret bits may exists at all the three pixels and value of average is the greatest integer less than or equal to $\frac{a'_i + a'_{i+2}}{2}$ (using floor function) of this i^{th} sub-block i.e. lap(i)=2. So, the secret bits sequence embedded here is $p_2p_1p_0$=111.

So, we get the watermarked sub-block as (a'_i, a_{i+1}', a_{i+2}') = (124,127,128) and they are stored as (m'_i, m'_{i+1}, m'_{i+2}) = (127,124,128). The data extraction for this watermarked sub-block is shown in Fig 4.

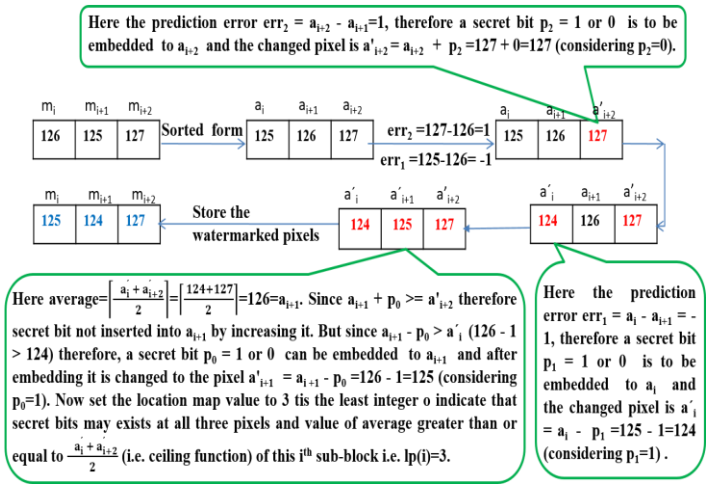

Fig 3: Example of a sub-block which is capable to embed three secret bits into maximum pixel, second maximum pixel and middle pixel with location map value is 3.

The data extraction for the above watermarked sub-block is shown in Fig 5.

Others embedding and extracting situations are as obvious as previous.

Data Extraction Procedure

A watermarked image is divided into three pixels non-overlapping sub-blocks. In Extracting method, we extract the secret bits and restore the pixel values in its original form as follows.

Algorithm for data extraction and image recovery

Input: A watermarked image of order x × y.

Output: Extraction of hided secret bits and fully recover the cover image.

Step 1: First we read M=2*$\lceil(\log_2 T)\rceil$ number of least significant bits of pixels starting from beginning of the watermarked image to get the auxiliary information [i.e. last embedding block number blno and length of the compressed location lplen]. Then read next lplen number of lease significant bits of pixels to get the value of compressed location map and then decompressing it to get the array of location map values "lp()".

Step 2: Starting from pixel number (M + lplen + 1), we divide the watermarked image into 3 pixels non-overlapped sub-blocks. Then extract R=M+ lplen hided secret bits from the sub-block number (blno + 1) to the last block using the following data extraction and image restoration procedure. Replace the least significant bits of first R pixels by the sequence of R extracted bits. Again by applying the following procedure retrieve the watermark bit sequence and restore the pixels in its original form.

Step 3: Let a generalized i^{th} sub-block contains the pixel values as (m'_i, m'_{i+1}, m'_{i+2}). After sorting them in ascending order we obtain (a'_i,,a'_{i+1},a'_{i+2}). [i.e. $a'_i \le a'_{i+1} \le a'_{i+2}$].

[Here no secret bit exists if location map value, lp(i)=0 for i^{th} sub-block. Secret bits may exist either at a'_{i+2} or at a'_i or both if lp(i)=1. lp(i)=2, indicates that secret bits may exist either at a'_{i+2} or at a'_i or both and at the middle pixel a'_{i+1}. In this case average value of a'_{i+2} and a'_i is find out using floor function$(\lfloor . \rfloor)$. Also location map value lp(i)=3 implies that, for the i^{th} sub-block, hided secret bits may exist either at a'_i or at a'_{i+2} or both and at a'_{i+1}, where average value of a'_{i+2} and a'_i is calculated using ceiling function$(\lceil . \rceil)$.]

Step 4: If lp(i)=0 then secret bits does not exist in that sub-block and pixel values are unchanged. Therefore restore the pixel values (a'_i,, a'_{i+1}, a'_{i+2}) into its exact positions and we get the cover image sub-block as (m_i, m_{i+1}, m_{i+2}). Set i=i+1 and continue from step 3.

otherwise continue from step 5.

[end of if]

Step 5: If lp(i)=1, secret bits may exist at both side pixels in the sorted sub-block but not at the middle.

a) To find out secret bit from maximum pixel we calculate the prediction error $err_2' = a'_{i+2} - a'_{i+1}$. Now using the following procedure, we extract the secret bit and restored the pixel with its original value.

For $err_2'=0$, a'_{i+2} becomes unchanged because no secret bit is there and the original pixel value is $a_{i+2}=a'_{i+2}$. If $err_2' \ge 1$ and $err_2' \le 2$, secret bit is fetched by the calculation $p_2=err_2'-1$ and the original pixel value is restored as $a_{i+2}=a'_{i+2} - p_2$. Also for $err_2' > 2$, secret bit is not existing at a'_{i+2} but this pixel is shifted in data embedding time therefore its original pixel value is $a_{i+2} = a'_{i+2} - 1$

b) To find out secret bit from minimum pixel we calculate the prediction error $err_1' = a'_i - a'_{i+1}$. Secret bit extraction and image restoration from minimum pixel a'_i as follows.

For $err_1'=0$, no secret bits are there and a'_i becomes unchanged and the original pixel value is $a_i=a'_i$. For $-2 \le err_1' \le -1$, secret bit exists in minimum pixel it is founded as $p_1=|err_1'| - 1$ and the original pixel value is restored as $a_i=a'_i + p_1$. Also If $err_1' > -2$, no secret is present at a'_i and this pixel is shifted in data embedding time therefore its original value is $a_i=a'_i+1$

So, the required fetched bit sequence is $p_2\ p_1$ (if they exists) and the recovered cover pixels of i^{th} sub-block is (a_i, a'_{i+1}, a_{i+2}). Now go to step 8.

[end of outer if of step 5]

Step 6: If lp(i)=2 then

a) To fetch the secret bit from middle pixel a'_{i+1} and restored it in original form calculate the average value as average = $\left\lfloor \frac{a'_i + a'_{i+2}}{2} \right\rfloor$.

If a'_{i+1} - average = 0, secret bit is there and the secret bit is $p_0=0$ and the recovered cover pixel value is $a_{i+1}=a'_{i+1}$. For s'_{i+1} – average > 0, secret bit is present there and it is $p_0=1$ and the recovered cover pixel value is $a_{i+1}=a'_{i+1} - 1$. Again if a'_{i+1} – average < 0, secret bit is there and the secret bit is $p_0=1$ and the recovered cover pixel value is $a_{i+1}=a'_{i+1} + 1$

b) Applying the same procedure as Step-5(a) we extract the secret bit say p_2 from maximum pixel and restore it in original form.

c) Applying the same procedure as Step-5(b) we extract the secret bit say p_1 from minimum pixel and restore it in original form.

So, the required fetched bit sequence is $p_2\ p_1 p_0$ (if they exists) and the recovered cover pixels of i^{th} sub-block is (a_i,,a_{i+1}, a_{i+2}). Now go to step 8.

[end of outer if of step 6]

Step 7: If lp(i)=3 then

a) To fetch the secret bit from middle pixel a'_{i+1} and restored it in original form calculate the average value as average = $\left\lfloor \frac{a'_i + a'_{i+2}}{2} \right\rfloor$.

Applying the same procedure as Step-6(a) we extract the secret bit say p_0 from the middle pixel and restore it in original form.

b) Applying the same procedure as Step-5(a) we extract the secret bit say p_2 from the maximum pixel and restore it in original form.

c) Applying the same procedure as Step-5(b) we extract the secret bit say p_1 from the minimum pixel and restore it in original form.

So, the required fetched bit sequence is $p_2\ p_1 p_0$ (if they exists) and the recovered cover pixels of i^{th} sub-block is (a_i,,a_{i+1}, a_{i+2}).

[end of outer if of step 7]

Step 8: Store the recovered cover pixels of i^{th} sub-block into its original positions (m_i, m_{i+1}, m_{i+2}). Set i=i+1 and the above process continued until all sub-blocks are completed.

Step 9: Applying the above procedure ultimately we fetched all hided secret bits from the watermarked image and the cover image restored completely.

Numerical examples of data extraction

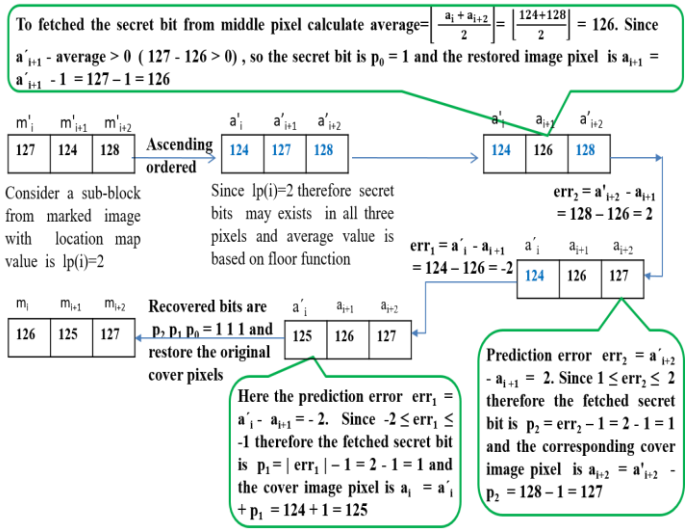

Fig 4: Example of watermarked sub-blocks whose location map value is 2 and three secret bits are fetched from middle pixel depending on average value and from maximum pixel and minimum pixels depending on prediction error and restore the cover pixels for that sub-block.

In Fig 4 a watermarked i^{th} sub-block is considered for data extraction and image recovery. Here the pixel values are m'_i=127, m'_{i+1}= 124, m'_{i+2}=128 and location map value is lp(i)=2.

Sorted these pixels in ascending order we get a'_i=124, a'_{i+1}=127 and a'_{i+2}=128.

For lp(i)=2 secret bits may exist in all three pixels and average value is calculated based on floor function.

We recovered secret bit from middle pixel at first and restored the middle pixel value in its original form. Using this original middle pixel value, we fetched the secret bit (if exists) from maximum pixel and minimum pixel and restored the pixel values in their original form. The required bit sequence is the fetched bit from maximum pixel followed by fetched bit from minimum pixel followed by fetched bit from middle pixel.

Calculate average = $\left\lfloor \frac{a'_i + a'_{i+2}}{2} \right\rfloor = \left\lfloor \frac{124+128}{2} \right\rfloor$ = 126. Since a'_{i+1} – average > 0 (127 - 126 > 0), so, the secret bit is p_0 = 1 and the restored image pixel is $a_{i+1} = a'_{i+1} - 1 = 127 – 1 = 126$.

Now calculate the prediction error $err_2=a'_{i+2} – a_{i+1}=128 – 126 = 2$. Since $err_2=2$ therefore the fetched secret bit $p_2=err_2 -1=2 – 1 = 1$ and the cover pixel is restored as $a_{i+2}=a'_{i+2} – p_2=128 – 1 = 127$. Again calculate the prediction error $err_1=a'_i - a'_{i+1} = 124 – 126 = -2$. Since $err_1=-2$ ($-2 \le err_1 \le -1$) therefore the required secret bit is find out as $p_1=| err_1 | - 1 = 2 - 1=1$ and the cover pixel is restored as $a_i = a'_i + p_1 = 124 + 1 = 125$. So, the required recovered bit sequence is $p_2p_1p_0$=111and the cover pixel sub-block becomes (a_i, a_{i+1}, a_{i+2}) = (125,126,127) and they are stored into their original positon as (m_i, m_{i+1}, m_{i+2}) = (126,125,127).

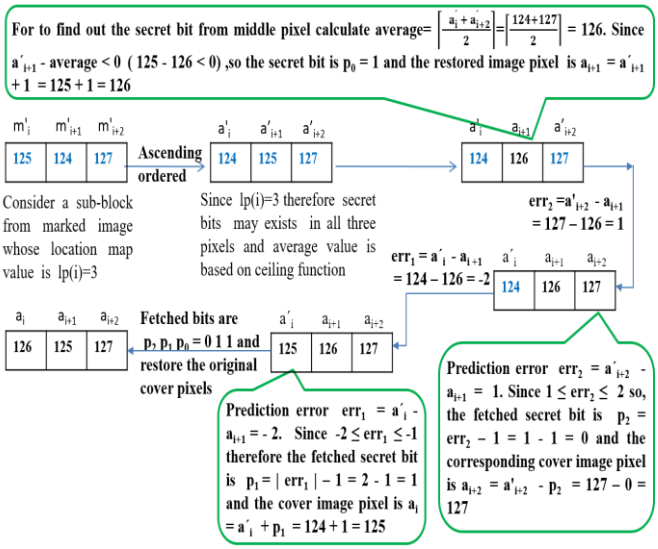

Fig 5: Example of watermarked sub-block whose location map value is 3 and three secret bits are fetched from middle pixel depending on average value and from the maximum pixel and minimum pixel depending on prediction error and restore the cover pixels for that sub-block.

EXPERIMENTAL RESULTS AND COMPARISONS

Watermarked image quality factors i.e. the embedding rate (ER), Peak signal to noise ratio(PSNR) and universal image quality index Q are generated using the formulas presented in [2].

Experimental results of this proposed method is implemented by using MATLAB. Experiment is done by using 512×512 grey scale cover images taking from USC-SIPT image database and researchget.net. In MATLAB secret bits are randomly generated by using the function randi([0,1],1,1).

In the proposed method each 3 pixels' sub-block could contain maximum three bits. Therefore, the total number of secret bits could be embedded in a cover image of order r × c is (r × c) ×$\frac{3}{3}$=(r × c)

So for a 512 × 512 grey scale image, maximum secret bits could be embedded in the proposed scheme is(512 × 512) × $\frac{3}{3}$= 262144 bits.

In the proposed scheme if we neglect a few sub-blocks that contain auxiliary information and location map, then for each other 3-pixels sub-blocks all three pixels will be modified to embed secret bits.

Therefore, in the proposed scheme the embedding capacity is higher than the Jung's method and Li et al.'s method.

Also theoretically the PSNR of the marked image versus its cover image is at least $10\log_{10}\frac{255^2}{MSE} \ge 10\log_{10}\frac{255^2 \times 3}{3}$= 48.13dB.

But practically in the proposed scheme since we skipped all the 3-pixels sub-blocks which have no capability to embed at least one secret bit (i.e. we avoid unnecessary expansions of pixels) therefore the average PSNR value is higher than Li et al.'s method and Jung's method.

Images are used in the proposed scheme as follows.

Image used for the experiment

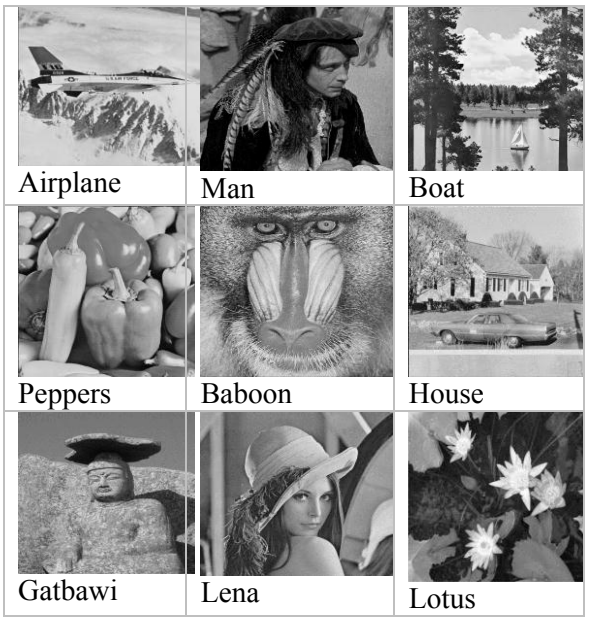

Fig 6: All are grayscale images of size 512 × 512 taken from USC-SIPI image database for the experiment.

Comparison Table of experimental results of the proposed scheme with the existing scheme

The results of proposed scheme compared with Li et al.'s method [1] and Jung's method [2] using nine standards 512×512 sized grey scale images are shown in Table 1 and Table 2 as follows.

Table 1: Comparisons of experimental results (Embedding capacity and embedding rate) of the proposed method with other methods [1,2]

	Li et al.'s method [1]		Jung's method [2]		Proposed method	
Cover Image	EC	ER	EC	ER	EC	ER
Airplane	12260	0.047	16098	0.061	58415	0.2228
Man	8275	0.032	9533	0.036	27824	0.1061
Boat	7043	0.027	8450	0.032	40592	0.1548
Lena	9799	0.037	11048	0.042	48533	0.1851
Peppers	8918	0.034	10805	0.041	37887	0.1445
Baboon	3839	0.015	4508	0.017	12989	0.0495
House	9910	0.038	13194	0.050	40829	0.1558
Lotus	11423	0.044	12685	0.048	59541	0.2271
Gatbawi	7105	0.027	9361	0.036	39703	0.1515
Average	8730	0.033	10631	0.040	40701	0.1552

On the experiment, compressed location map bits with auxiliary information are not stored into the watermarked image.

On average the embedding capacity of Li et al.'s method is 8730 bits and Jung's method is 10631 bits, but in the proposed method it is 40701. This shows that the proposed method hides 30070 bits more on average than Jung's method and 31971 bits more than Li et al.'s method. Also, the average value of ER is 0.1149 more than Jung's method and 0.1218 more than Li et al.'s method.

Table 2: Comparisons of experimental results (PSNR and Quality Index) of the proposed method with other methods [1,2]

	Li et al.'s method [1]		Jung's method [2]		Proposed method	
Cover Image	PSNR	Q	PSNR	Q	PSNR	Q
Airplane	55.82	0.9889	56.85	0.9910	55.73	1.000
Man	45.32	0.9965	45.40	0.9971	57.47	1.000
Boat	55.91	0.9976	57.05	0.9983	56.89	1.000
Lena	55.56	0.9955	57.09	0.9968	55.70	1.000
Peppers	55.66	0.9965	57.30	0.9976	56.21	1.000
Baboon	56.04	0.9995	56.97	0.9996	60.15	1.000
House	55.95	0.9852	57.07	0.9887	57.11	1.000
Lotus	41.98	0.9935	42.58	0.9949	56.22	1.000
Gatbawi	34.24	0.9888	31.84	0.9909	57.65	1.000
Average	50.72	0.9936	51.31	0.9950	57.01	1.000

EC: Embedded capacity, ER: Embedded rate, PSNR: Peak signal to noise ratio, Q: Image quality index

On the other hand, the average PSNR of Li et al.'s method is 50.72 dB and that of Jung's method is 51.31 dB, but in the proposed method it is 57.01dB. So, the proposed method improves the average PSNR 5.7 dB more than Jung's method and 6.29 dB more than Li et al.'s method.

From the above table, it is also shown that the average quality index Q is better than the Li et al.'s and Jung's method.

Comparison of Graphs

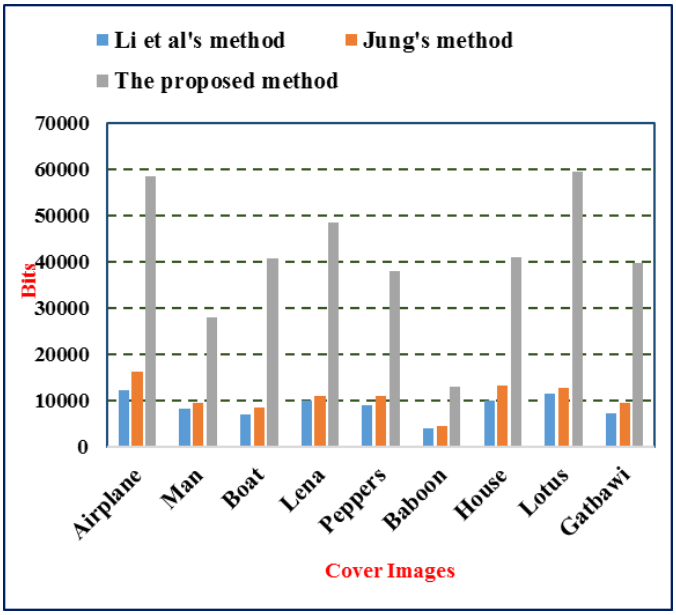

Fig 7: Graph to show a comparison of embedding bits of the proposed scheme with the existing scheme

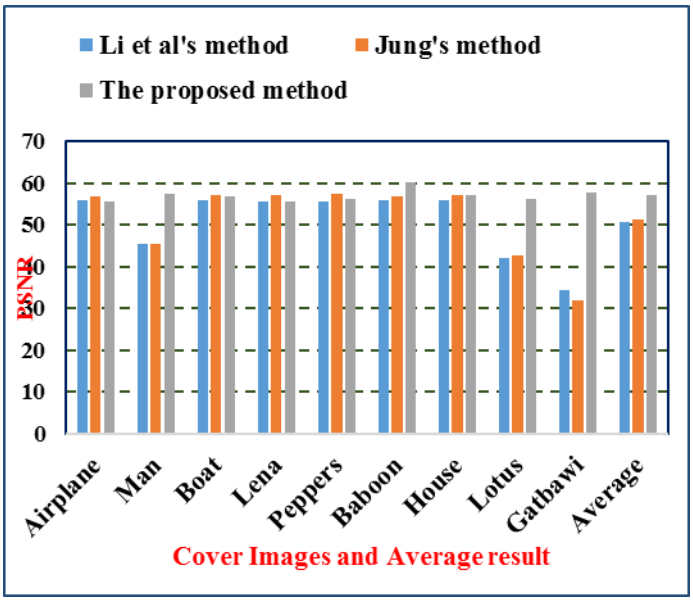

Fig 8: Graph to show a comparison of PSNR of proposed scheme with existing schemes

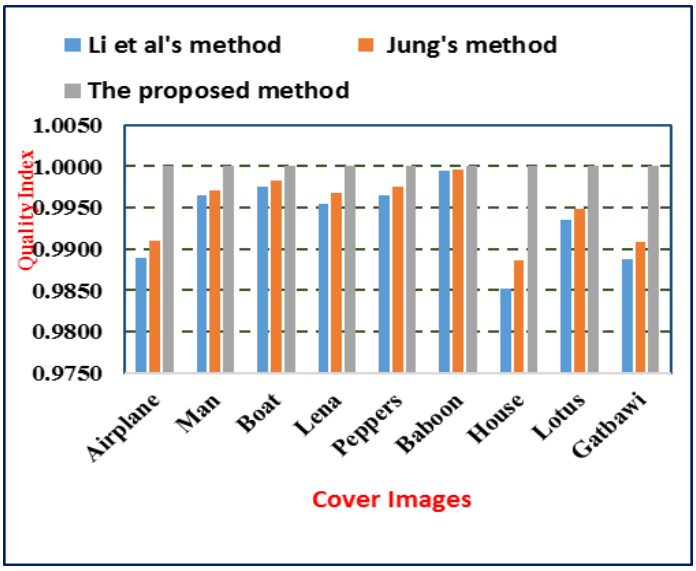

Fig 9: Graph to show a comparison of the visual quality index between proposed schemes with existing schemes.

Conclusion

Here, we proposed a new Pixel-value ordering and Prediction error expansion with the average value of pixels based reversible watermarking scheme to improve Li et al.'s method and Jung's method. In this scheme, we calculate the prediction errors on ordered pixels of three-pixel sub-blocks and then skip those blocks which are not capable to embed data. Then we try to embed two secret bits into each remaining sub-block depending on the Prediction error expansion technique and then store another one secret bit into middle pixel depending on the average value of end pixels. In this way, we get a watermarked image and finally from the watermarked image we can fetch the secret bits and restore the cover image completely. Experimental results of the proposed scheme show that the embedding capacity, average PSNR, and quality index is better than Li et al.'s method and Jung's method. In the future, one can try to improve embedding capacity with improved visual quality in a different domain using PVO based PEE technique.

References

[1] Li, X., Li, J., Li, B. and Yang, B. Hsiao, J.Y. High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion. *Signal Processing 93*(2013) 198-205.

[2] Jung, K.H. Data Hiding Scheme with Reversibility Based on Neighboring Pixel Value Grouping and Ordering. *Jour of Adv Research in Dynamical & Control Systems*10 (2)(2018)227-233.

[3] J. Fridrich, M. Goljan, R. Du, Lossless data embedding - new paradigm in digital watermarking, EURASIP Journal on Applied Signal Processing 2002 (2) (2002) 185–196.

[4] M.U. Celik, G. Sharma, A.M. Tekalp, E. Saber, Lossless generalized- LSB data embedding, IEEE Transactions on Image Processing 14 (2) (2005) 253–266.

[5] Tai, W.L., Yeh, C.M. and Chang, C.C. Reversible data hiding based on histogram modification of pixel differences. *IEEE Transactions on Circuits and Systems for Video Technology* **19** (2009) 906-910.

[6] Tian, J. Reversible data embedding using a difference expansion. *IEEE Transactions on Circuits and Systems for Video Technology* **13** (2003) 890-896.

[7] Alattar, A.M. Reversible watermark using the difference expansion of a generalized integer transform. *IEEE Transactions on Image Processing* **13** (2004) 1147-1156.

[8] Al-Qershi, O.M. and Khoo, B.E. Two-dimensional difference expansion (2D-DE) scheme with a characteristics-based threshold. *Signal Processing* **93** (1) (2013) 154-162.

[9] F. Peng, X. Li, B. Yang, Adaptive reversible data hiding scheme based on integer transform, Signal Processing 92 (1) (2012) 54–62.

[10] Luo, H., Yu, F.X., Chen, H., Huang, Z.L., Li, H. and Wang, P.H. Reversible datahiding based on block median preservation. *Information Sciences* **181** (2011) 308-328.

[11] Zhao, Z., Luo, H., Lu, Z.M. and Pan, J.S. Reversible data hiding based on multilevel histogram modification and sequential recovery. *AEU - International Journal of Electronics and Communications* **65** (2011) 814-826.

[12] Huang, L.C., Tseng, L.Y. and Hwang, M.S. A reversible data hiding method by histogram shifting in high quality medical images. *The Journal of Systems and Software* **86** (2013) 716-727

[13] M. Liu, H.S. Seah, C. Zhu, W. Lin, F. Tian, Reducing location map in prediction-based difference expansion for reversible image data

embedding, Signal Processing 92 (3) (2012) 819–828.

[14] Y. Hu, H.K. Lee, J. Li, DE-based reversible data hiding with improved overflow location map, IEEE Transactions on Circuits and Systems for Video Technology 19 (2) (2009) 250–260.

[15] W. Hong, T.S. Chen, C.W. Shiu, Reversible data hiding for high quality images using modification of prediction errors, Journal of Systems and Software 82 (11) (2009) 1833–1842.

[16] C.C. Lin, N.L. Hsueh, A lossless data hiding scheme based on three- pixel block differences, Pattern Recognition 41 (4) (2008) 1415–1425.

[17] W.L. Tai, C.M. Yeh, C.C. Chang, Reversible data hiding based on histogram modification of pixel differences, IEEE Transactions on Circuits and Systems for Video Technology 19 (6) (2009) 906–910.

[18] P. Tsai, Y.C. Hu, H.L. Yeh, Reversible image hiding scheme using predictive coding and histogram shifting, Signal Processing 89 (6) (2009) 1129–1143.