A PVO-based Reversible Data Hiding Scheme Using Variable Difference

Rong Li¹, Xiang-yang Li²*, An JIANG¹ and Yan XIONG²

¹School of Computer Science, Guangdong Agriculture Industry Business Polytechnic, Guangzhou, 510507, China
²Institute of Computer Application, Guangdong Agriculture Industry Business Polytechnic, Guangzhou, 510507, China

*Corresponding author

Keywords: Reversible data hiding, Prediction-Error selection, Pixel-Value-Ordering.

Abstract. In order to improve the PSNR of the marked image, an improved high-fidelity reversible data hiding algorithm was proposed based on Pixel-Value-Ordering. There are many different forms of prediction errors used to embed information, each of which leads to different pixel shift rate. One certain appropriate form of prediction errors was selected for embedding data, which can not only provide adequate embedding capacity, but also increase the PSNR of marked image. The algorithm can extract the secret information and restore the original image without any damage. Experimental results show that the proposed algorithm compared with the original PVO algorithm can effectively improve the visual quality of the marked image.

Introduction

Image data hiding is usually divided into reversible data hiding (RDH) and irreversible data hiding. The main feature of reversible data hiding is reversibility, that is, while extracting hidden information, the original image can be restored without any deformation, and the reversible data hiding is also known as lossless data hiding. The performance of the reversible data hiding algorithm is usually evaluated in three aspects: the embedding capacity (EC); the visual quality of the image after hiding the information; the complexity of the algorithm. Now, data hiding has been used for a variety of applications such as authentication, ownership protection, annotation and secret communication.

At present, the reversible data hiding algorithm has been extensively researched and developed. Difference expansion (DE) [1, 2] has received extensive attention, which can provide a suitable embedding capacity by adjusting the threshold value. Histogram shifting (HS) [3, 4] is another method focusing on large data embedding capacity and high quality of the marked image. Prediction error expansion (PEE) [5-7] is a reversible data hiding method, which improves the performance of the algorithm by introducing predictive mechanisms. Li et al. [8] proposed a novel RDH scheme based on pixel value ordering (PVO), which is effective in obtaining higher image fidelity. Later, many other methods [9-12] based on PVO have been proposed. Ou et al. proposed a new prediction strategy named PVO-k [9] and a pixel-based PVO (PPVO) method [10], these methods take full advantage of the pixels in the smooth area that are completely ignored by the PVO method, to get bigger the embedding capacity than PVO method. Peng et al. [11] extended an improved reversible data hiding scheme based on pixel value ordering (IPVO), in which a new prediction error has been computed and new histogram modification strategy has been utilized, the blocks which the largest pixel value equals to the second largest pixel value can be exploited to embed data, and these blocks are not utilized in Li et al.’s [8] method, resulting that this method exploits more image redundancy space and outperforms better performance than Li et al.’s work. Subitha P et al. [12] proposed novel reversible pixel-value-ordering technique, which contains two improvement strategies: Novel Difference Computation (DC) and Novel Histogram Shifting (HS), this technique significantly improves the EC along with PSNR. Jung et al. [13] proposed an RDH method based on sorting and prediction for three pixels in pixel blocks, which provides high embedding capacity and good image quality.
In order to acquire better visual quality of marked image, the less modification on an original image is expected. After grouping and sorting image pixel, we study carefully the distribution of the difference between the maximum pixel value and the adjacent pixel value or the minimum pixel value and the adjacent pixel value, and then choose an appropriate form to embed the information. An improved data hiding scheme based on PVO is proposed that has higher-fidelity RDH with a very low embedding distortion. The method can effectively improve the quality of the cover image under the same embedding capacity, which leads to the better visual performance.

Related Work
Li et al. [8] proposed a data hiding scheme based on pixel-value-ordering and predication-error expansion. The main process of the method is as follows: firstly, dividing an image into non-overlapping blocks that consists a certain number of pixels, such as four pixels \( x_1, x_2, x_3 \text{ and } x_4 \) as shown in Figure 1(a); secondly, pixels in a block are sorted in a descending order and are denoted as \( x'_1, x'_2, x'_3 \text{ and } x'_4 \) as shown in Figure 1(b); thirdly, calculating the largest pixel value \( x'_4 \) using the following formula (1).

\[
x'_4 = \begin{cases} 
    x'_4 & \text{if } x'_4 - x'_3 = 0 \\
    x'_4 + m & \text{if } x'_4 - x'_3 = 1 \\
    x'_4 + 1 & \text{otherwise}
\end{cases} \quad (1)
\]

when \( x'_4 - x'_3 = 0 \), the result of \( x'_4 \) would be unchanged, here \( x'_4 = x'_4 \); when \( x'_4 - x'_3 = 1 \), the secret message bit \( m \) is embedded into \( x'_4 \) and would become to \( x''_4 \), here \( x''_4 = x'_4 + m \); otherwise, the result of \( x'_4 \) would be added 1, all process as show in Figure 1(c).

![Figure 1. Example of data hiding process based on PVO scheme](image)

The recovery process is the reverse of its embedding process. You can use following formula (2-1) and (2-2) extract the secret message bit from a marked block and recover the original pixel.

\[
m = \begin{cases} 
    0 & \text{if } x''_4 - x'_3 = 0 \\
    1 & \text{if } x''_4 - x'_3 = 1 \\
    \text{null} & \text{otherwise}
\end{cases} \quad (2-1)
\]

\[
x'_4 = \begin{cases} 
    x'_4 & \text{if } x'_4 - x'_3 \in \{0, 1\} \\
    x'_4 - 1 & \text{otherwise}
\end{cases} \quad (2-2)
\]

Li et al.’s scheme is based on the concept that for a natural image the pixel values in a block are similar and the information is embedded when the difference is 1.

Proposed Method
In this section, we improved the scheme based on the PVO, and introduced the data embedding process and extraction process.

Embedding Process
We embed secret messages into a cover image and obtains a marked image. The embedding process is presented as following.

1. For a cover image \( I \) with \( m \times n \) pixels, a bit string \( M = m_0 m_1 \cdots m_{r-1} \) is the secret message
with \( r \) bits, which be embedded into the image, where \( m_j \in \{0,1\} \) and \( 0 \leq j \leq r - 1 \), the length of \( M \) is denoted as \( l_p \).

(2) The cover image \( I \) is divided into some non-overlapped blocks, which contains four neighboring pixels denoted by \( x_i, x_{i+1}, x_{i+2}, x_{i+3} \) respectively in block \( i \), here \( 0 \leq x_i, x_{i+1}, x_{i+2}, x_{i+3} \leq 255 \).

(3) For a block \( i \), the pixels are sorted in a descending order to get \( x_i', x_{i+1}', x_{i+2}', x_{i+3}' \). \( x_{i+3}' \) is the largest pixel value and \( x_i' \) is the smallest pixel value.

(4) For a block \( i \), calculating the difference between the largest and neighbor pixel values, and denoted as \( e1_i \) in block \( i \) that is derived from \( e1_i = x_{i+3}' - x_{i+2}' \). In the similar way, calculating the difference between the smallest and neighbor pixel values, and denoted as \( e2_i \) in block \( i \) that is derived from \( e2_i = x_{i+1}' - x_i' \).

(5) Search the number of various difference value, use \( n_0, n_1, \ldots, n_{t1} \) express the number of difference value \( e1_i \) is 0,1,\ldots,t1 and use \( m_0, m_1, \ldots, m_{t2} \) express the number of difference value \( e2_i \) is 0,1,\ldots,t2 respectively, compare \( n_t (0 \leq i \leq t1) \), \( m_k (0 \leq k \leq t2) \) and \( r \), choose \( n_t, m_k \) noted as \( n^* \) and \( m^* \) respectively that are able to meet the following objective function as shown in the formula (3).

\[
\begin{align*}
(n^* + m^*) &= \max(n_t + m_k) \\
(n^* + m^*) &\geq l_p
\end{align*}
\]

Symbol \( l_p \) is the information length.

(6) Next, the maximum pixels value \( x_{i+3}' \) is modified to get the marked pixel values \( x_{i+3}^* \) by taking

\[
x_{i+3}^* = \begin{cases} 
  x_{i+3}' & \text{if } x_{i+3}' - x_{i+2}' < n^* \\
  x_{i+3}' + m & \text{if } x_{i+3}' - x_{i+2}' = n^* \\
  x_{i+3}' + 1 & \text{if } x_{i+3}' - x_{i+2}' > n^*
\end{cases}
\]

The minimum pixels value \( x_i' \) is modified to get the marked pixel values \( x_i^* \) by taking

\[
x_i^* = \begin{cases} 
  x_i' & \text{if } x_{i+1}' - x_i' < m^* \\
  x_i' - m_{j+1} & \text{if } x_{i+1}' - x_i' = m^* \\
  x_i' - 1 & \text{if } x_{i+1}' - x_i' > m^*
\end{cases}
\]

Here \( m_j \in \{0,1\} \) is a secret bit, the prediction error \( x_{i+3}' - x_{i+2}' = n^* \) or \( x_{i+1}' - x_i' = m^* \) are used to embedded information, so the capacity is the number of pixels with \( n^* \) and \( m^* \).

(7) The complexity of block \( i \) is measured by the difference \( x_{i+2}' - x_{i+1}' \). A block is taken as a flat one and is chosen to embed secret data when its complexity is less than a predefined threshold T. It is reasonable that in a natural image most image blocks are smooth, so we would obtain more space for embedding a message than without prediction. All information be embedded by using formula (2) and (3), we can obtain the marked image \( I' \).

(8) For a block \( i \), pixel change may cause overflow/underflow, when \( x_{i+3}' = 255 \) and \( x_{i+3}' - x_{i+2}' \geq n^* \) or \( x_i' = 0 \) and \( x_i' - x_{i+1}' = m^* \). In this case, we set \( h1_i = 1 \) when \( x_{i+3}' = 255 \) and \( x_{i+3}' - x_{i+2}' \geq n^* \). At the same time, set \( h2_i = 1 \) when \( x_i' = 0 \) and \( x_i' - x_{i+1}' = m^* \), otherwise set \( h1_i = 0 \) and \( h2_i = 0 \). Let \( H = h1_1 h2_1 \cdots h1_t h2_t \) be the required overhead information for extracting the embedded messages. The length of bitmap \( H \) is \( 2t \), \( h1_j \in \{0,1\} \), \( 0 \leq j \leq t \) and \( h2_j \in \{0,1\} \), \( 0 \leq j \leq t \). Because only a few “1” in the overhead information, we can use lossless compression to significantly reduce the size of overhead information and denote compressed location map as \( H_{clm} \) and it's length as \( l_{clm} \).

(9) Finally, embed supplementary information \( SI \) and \( H_{clm} \) to the least significant bits (LSB) of first \( l_{st} \) + \( l_{clm} \) image pixels. Symbol \( L_{st} \) is the supplementary information length. Information SI include threshold T (8 bits), value \( n^* \) (8 bits) and \( m^* \) (8 bits), end location p (\( \lfloor log_2 N \rfloor \) bits) and length of the compression bitmap \( l_{clm} \) (\( \lfloor log_2 N \rfloor \) bits), so length of information SI is \( L_{st} = 24 + \).
2⌈\log_2 N⌉ bits, where \(N\) is the total number of image pixels and \([\ . \ ]\) is the ceiling function.

**Extraction Process**

The corresponding data extraction process is detailed as follows:

1. Read LSB of first \(24 + 2⌈\log_2 N⌉\) pixels of marked image to get the auxiliary information, that including \(T, n^*, m^*, p\) and \(l_{ctm}\), then read LSB of next \(l_{ctm}\) pixels to get the compressed location map \(H_{ctm}\) and decompress the location map.

2. Divide marked image \(I'\) into some blocks that combine four pixels as it was divided by the encoder in the embedding process.

3. For each block \(i\), sort the pixel values \(y_i, y_{i+1}, y_{i+2}, y_{i+3}\) in a descending order and marked as \(y'_i, y'_{i+1}, y'_{i+2}, y'_{i+3}\).

4. For each block \(i\), when \(y'_{i+2} - y'_{i+1} < T\) and \(h_i = 0\), calculate the difference \(e_1_i = y'_{i+3} - y'_{i+2}\) and \(e_2_i = y'_{i+1} - y'_i\) respectively.

5. For \(e_1_i\), we can extract the secret information as formula (6). For \(e_2_i\), we can extract the secret information as formula (7)

\[
\begin{align*}
   m_i &= \begin{cases} 
      0 & \text{if } e_1_i = n^* \\
      1 & \text{if } e_1_i = n^* + 1 \\
      \text{null otherwise} 
   \end{cases} 
   \quad (6) \\
   m_{i+1} &= \begin{cases} 
      0 & \text{if } e_2_i = m^* \\
      1 & \text{if } e_2_i = m^* + 1 \\
      \text{null otherwise} 
   \end{cases} 
   \quad (7) 
\end{align*}
\]

At the same time, restore pixel \(y'_{i+3}\) and \(y'_i\) as follows formula (8) and (9) and mark as \(y^*_{i+3}\) and \(y^*_i\), respectively.

\[
\begin{align*}
   y^*_{i+3} &= \begin{cases} 
      y'_{i+3} & \text{if } e_1_i \leq n^* \\
      y'_{i+3} - 1 & \text{if } e_1_i \geq n^* + 1 
   \end{cases} 
   \quad (8) \\
   y^*_i &= \begin{cases} 
      y'_i & \text{if } e_2_i \leq n^* \\
      y'_i + 1 & \text{if } e_2_i \geq n^* + 1 
   \end{cases} 
   \quad (9) 
\end{align*}
\]

6. According to the block index, the message bits extracted from blocks are rearranged, so that the secret information is obtained and the original cover image \(I\) is restored lossless.

**Example of Embedding and Extracting Procedures**

Now we illustrate the proposed scheme, the example image is a gray-level one with pixel value between 0 and 255. Divide the cover image into non-overlapped blocks with \(2 \times 2\) pixels and choose 6 blocks of them as examples. Hypothesis \(n^* = 2\) and \(m^* = 3\) and the secret message \(M = 1011\), the specific information reference table 1.

<table>
<thead>
<tr>
<th>block</th>
<th>(x_1)</th>
<th>(x_{i+1})</th>
<th>(x_{i+2})</th>
<th>(x_{i+3})</th>
<th>(x'_i)</th>
<th>(x'_{i+1})</th>
<th>(x'_{i+2})</th>
<th>(x'_{i+3})</th>
<th>(x'_i)</th>
<th>(x'_{i+1})</th>
<th>(x'_{i+2})</th>
<th>(x'_{i+3})</th>
<th>smooth or not</th>
<th>Min bit embedding</th>
<th>Max bit embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>144</td>
<td>148</td>
<td>147</td>
<td>146</td>
<td>144</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>143</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>Yes</td>
<td>1</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>148</td>
<td>143</td>
<td>140</td>
<td>145</td>
<td>140</td>
<td>143</td>
<td>145</td>
<td>148</td>
<td>139</td>
<td>143</td>
<td>145</td>
<td>149</td>
<td>Yes</td>
<td>left shift 1 bit</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>168</td>
<td>170</td>
<td>169</td>
<td>171</td>
<td>168</td>
<td>169</td>
<td>170</td>
<td>171</td>
<td>168</td>
<td>169</td>
<td>170</td>
<td>171</td>
<td>Yes</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>4</td>
<td>92</td>
<td>97</td>
<td>90</td>
<td>94</td>
<td>90</td>
<td>92</td>
<td>94</td>
<td>97</td>
<td>89</td>
<td>92</td>
<td>94</td>
<td>98</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>78</td>
<td>82</td>
<td>73</td>
<td>86</td>
<td>73</td>
<td>78</td>
<td>82</td>
<td>86</td>
<td>72</td>
<td>78</td>
<td>82</td>
<td>87</td>
<td>Yes</td>
<td>left shift 1 bit</td>
<td>right shift 1 bit</td>
</tr>
<tr>
<td>6</td>
<td>101</td>
<td>98</td>
<td>68</td>
<td>46</td>
<td>46</td>
<td>68</td>
<td>98</td>
<td>101</td>
<td>46</td>
<td>68</td>
<td>98</td>
<td>101</td>
<td>No</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>
Table 2. An example of the extracting procedure.

<table>
<thead>
<tr>
<th>block</th>
<th>$x_i$</th>
<th>$x_{i+1}$</th>
<th>$x_{i+2}$</th>
<th>$x_{i+3}$</th>
<th>$x_i'$</th>
<th>$x_{i+1}'$</th>
<th>$x_{i+2}'$</th>
<th>$x_{i+3}'$</th>
<th>smooth or not</th>
<th>extracting information from Min bit</th>
<th>extracting information from Max bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>143</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>143</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>Yes</td>
<td>1</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>139</td>
<td>143</td>
<td>145</td>
<td>149</td>
<td>139</td>
<td>143</td>
<td>145</td>
<td>149</td>
<td>Yes</td>
<td>right shift 1 bit</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>168</td>
<td>169</td>
<td>170</td>
<td>169</td>
<td>171</td>
<td>168</td>
<td>169</td>
<td>171</td>
<td>Yes</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>4</td>
<td>89</td>
<td>92</td>
<td>94</td>
<td>89</td>
<td>92</td>
<td>94</td>
<td>90</td>
<td>92</td>
<td>Yes</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>72</td>
<td>78</td>
<td>82</td>
<td>78</td>
<td>72</td>
<td>78</td>
<td>82</td>
<td>73</td>
<td>Yes</td>
<td>right shift 1 bit</td>
<td>left shift 1 bit</td>
</tr>
<tr>
<td>6</td>
<td>46</td>
<td>68</td>
<td>98</td>
<td>101</td>
<td>46</td>
<td>68</td>
<td>98</td>
<td>101</td>
<td>No</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

In the table 1, the column “smooth or not” Used to indicate whether the block is smooth area, the column “Min bit embedding” and “Max bit embedding” denotes message bits to be embedded or other operations were carried out. According to the proposed scheme, block 1 which embed $m_0 = 1$ into the minimum pixel and embed nothing in the maximum pixel, block 2 which shift 1 bit to left in the minimum pixel and embed $m_1 = 0$ into the maximum pixel and, the pixels in block 3 are remained unchanged, $m_2 = 1$ and $m_3 = 1$ are embedded into minimum and maximum pixel of block 4 respectively, block 5 is Executed shift operation and block 6 have no change because they don’t meet the smooth condition. Based on the above-mentioned process, we can simplicity complete the embedding process.

The extracting procedure is the reverse of embedding procedure. As described in Table 1, the information extraction process is shown in Table 2.

Experiment Results

In this section, to evaluate the performance of proposed scheme, a comparison test between the proposed algorithm and other recently proposed algorithm is carried out using the test images as shown in Figure 2.

![Test images](image)

Figure 2. Test images.

Each image may be divided into $512 \times 512 \div 4 = 65536$ non-overlapping blocks, and the embedding capacity of an image may be up to $65536 \times 2$ bits. The proposed method only modifies each pixel value at most by 1. For a 256-gray-level image with $m \times n$ pixels, the image quality or the similarity between original image and secret image can be evaluated by PSNR.

We changed EC with a step size. Figure 3 shows the performance comparison of our method with other three recently proposed methods that are PEE, HS and Li’s [8] schema. The results clearly show that our method has a higher PSNR for test images in the majority cases.

Summary

In this paper, an improved scheme based on generalized PVO-based embedding was proposed. In some cases, the demand for maximum EC is not large, we not only chose the maximum pixel value or minimum pixel value in a block for data embedding when the difference is 1, more extensively, considering the performance of cover image, other blocks that difference is bigger than 1 are utilized. Under meeting the demand of embedding capacity, the larger pixel difference is chosen to embed
information. Experimental results show that the proposed method can achieve a higher PSNR under the same condition of EC comparing to Li’s scheme, it is a better RDH scheme which has low distortion and satisfied visual quality.

Figure 3. Performance comparison between our method and other three methods.

Acknowledgments
This research work is supported by National Natural Science Research Project of China (No. 17BQNS01007).

References


