Reversible Data Hiding Based on Histogram by Using Gradient Adjacent Prediction

Y.-C. Lee,
Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan.
E-mail: 1qaz5222@gmail.com

H.-H. Chen,
Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan.
E-mail: texasflamedh@gmail.com

C.-Y. Chen
Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan.
E-mail: cychen07@nuk.edu.tw

Abstract
In this paper, we use gradient adjacent prediction to present a new data hiding method based on histogram. Our method can increase the amount of hidden data, and keep higher PSNR of stego-image. According to experiment results, our method increases the amount of hidden data by 12.5% on the average as compared with Jang et al.’s method.

Keywords: gradient adjacent prediction, histogram, reversible data hiding.

1. Introduction
Thanks to the flourish of the computer hardware and the Internet, the advanced technologies and their applications advert more and more rapidly. To avoid the interception of sensitive information by a malicious user during transmission, data hiding has attracted significantly much attention [1-7]. The philosophy of data hiding is to embed the secret data into a cover image and to generate quality stego-image with PSNR higher than 40dB. Since the stego-image with PSNR higher than 40dB is imperceptible to human’s recognition, the malicious user will not be able to recognize whether the stego-image is embedded with the secret data or not. Only the specific user can extract the secret data from the stego-image.

Two major concerns of data hiding method are the quality of and the amount of secret data embedded in a stego-image. Reversible data hiding method can be categorized as three classes, which are histogram-based methods, Difference Expansion [9], and VQ-Compressed Domain strategies [10]. The histogram-based methods are widely-adopted due to their efficiency and economic computation complexity. In this paper we propose a method derived from the well-known gradient adjacent prediction (GAP) to improve the accuracy of prediction. According to experiment
results, our method can increase the amount of hidden secret data, and keep higher PSNR of stego-image.

The paper is organized as follows. In Section 2, we review related work. Section 3 presents our method. We give the experiment result in Section 4. The last section concludes this paper.

2. Related Work

In this section, we review the algorithm of Jheng et al.’s method [8].

At first, we give the required definitions in this paper. The size of cover image is $512 \times 512$. Pixel $(i, j)$ in the cover image has a pixel value $P_{(i,j)} \in \{0 \sim 255\}$. The surrounding pixels of pixel $(i, j)$, $P_a, P_b, P_c, P_d, P_e, P_f, P_g, P_h, P_i, P_j, P_k$, are shown as Figure 1.

![Figure 1. The surrounding pixels of pixel $(i, j)$](image)

Let $4$-neighbor set and oblique set of pixel $(i, j)$ be $\alpha_{(i,j)}$ and $\beta_{(i,j)}$, respectively. We then have

$$\alpha_{(i,j)} = \{(x, y)| x, y \in [0, 511], (i-x)^2 + (j-y)^2 = 1\} \text{ and}$$

$$\beta_{(i,j)} = \{(x, y)| x, y \in [0, 511], (i-x)^2 + (j-y)^2 = \sqrt{2}\}.$$

We give the example of $(i, j)=(1,1)$ and get

$$\alpha_{(1,1)} = \{(0,1), (2,1), (1,0), (1,2)\}, \beta_{(1,1)} = \{(0,0), (0,2), (2,0), (2,2)\}.$$

The prediction of gradient around pixel $(i, j)$, $S_{(i,j)}$, is computed by

$$S_{(i,j)} = (D^h_{(i,j)} + D^e_{(i,j)}) - (D^l_{(i,j)} + D^f_{(i,j)}),$$

where

$$D^h_{(i,j)} = |P_i - P_n| + |P_g - P_1| + |P_a - P_4|, D^e_{(i,j)} = |P_g - P_m| + |P_a - P_3| + |P_d - P_7|,$$

$$D^l_{(i,j)} = |P_f - P_m| + |P_a - P_5| + |P_b - P_9|, \text{ and } D^f_{(i,j)} = |P_d - P_a| + |P_h - P_n| + |P_e - P_i|.$$

We use $S_{(i,j)}$ to compute $D_{(i,j)}$ according to the gradient status as shown in Table 1, which is based on the previous work [8]. In Table 1, $|\alpha_{(i,j)}|$ and $|\beta_{(i,j)}|$ are the total numbers of elements in a
set of $\alpha_{(i,j)}$ and $\beta_{(i,j)}$, respectively. We further compute

\[ D'_{(i,j)} = P_{(i,j)} - D_{(i,j)} \]

The previously calculated prediction error are shifted to obtain new prediction error $D^*_{(i,j)}$.

### Table 1. The weights of Gradient types

<table>
<thead>
<tr>
<th>Gradient Type</th>
<th>$S_{(i,j)}$</th>
<th>$D_{(i,j)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong Diagonal Similarity</td>
<td>(80, ∞]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.2}{\sum \beta_{(i,j)} \times 0.8}$</td>
</tr>
<tr>
<td>Diagnot Similarity</td>
<td>(32, 80]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.3}{\sum \beta_{(i,j)} \times 0.7}$</td>
</tr>
<tr>
<td>Weak Diagonal Similarity</td>
<td>(8, 32]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.4}{\sum \beta_{(i,j)} \times 0.6}$</td>
</tr>
<tr>
<td>Uniform Similarity</td>
<td>[8, -8]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.5}{\sum \beta_{(i,j)} \times 0.5}$</td>
</tr>
<tr>
<td>Weak 4-Neighbor Similarity</td>
<td>(-8, -32]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.6}{\sum \beta_{(i,j)} \times 0.4}$</td>
</tr>
<tr>
<td>4-Neighbor Similarity</td>
<td>(-32, -80]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.7}{\sum \beta_{(i,j)} \times 0.3}$</td>
</tr>
<tr>
<td>Strong 4-Neighbor Similarity</td>
<td>(-80, -∞]</td>
<td>$\frac{\sum \alpha_{(i,j)} \times 0.8}{\sum \beta_{(i,j)} \times 0.2}$</td>
</tr>
</tbody>
</table>

Let $M$ be the secret data to be embedded and $M_n$ be n-th bit of $M$. We further assume that the sum of peaks of histogram are higher than the length of secret data $M$.

The prediction error $D'_{(i,j)}$ of pixel $(i, j)$ is determined as follows. First we look up Table 1 according to $S_{(i,j)}$, to calculate the prediction of gray value $D_{(i,j)}$. The difference between the predicted value and the actual value of pixel $(i, j)$ is then calculated, and the result is assigned to $D'_{(i,j)}$. Once the histogram of prediction errors is accumulated, two highest peaks are selected and denoted as $H_{1x}$ and $H_{2x}$. The peaks with zero accumulation closest to these two peaks are then selected and denoted as $Z_{1x}$ and $Z_{2x}$. Afterwards, the secret data are embedded using the obtained results.

We use Figure 2 to show how to compute $D'_{(i,j)}$.

<table>
<thead>
<tr>
<th></th>
<th>106</th>
<th>108</th>
<th>98</th>
<th>96</th>
<th>91</th>
</tr>
</thead>
<tbody>
<tr>
<td>108</td>
<td>107</td>
<td>99</td>
<td>98</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>110</td>
<td>103</td>
<td>100</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>108</td>
<td>106</td>
<td>94</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>109</td>
<td>107</td>
<td>105</td>
<td>99</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 2. Cover image**
According to the pixel value, $P_{(i,j)}=103$, we calculate $D_{(i,j)}^h=9$, $D_{(i,j)}^v=6$, $D_{(i,j)}^l=12$, $D_{(i,j)}^r=21$ and $S_{(i,j)}=-18$. According to $S_{(i,j)}=-18$, we can figure out that specified pixel is classified to Weak 4-Neighbor. We then compute

$$D_{(i,j)} = (\sum_{\alpha} (\frac{\alpha_{(i,j)}}{|\alpha_{(i,j)}|}) \times 0.6 + \sum_{\beta} (\frac{\beta_{(i,j)}}{|\beta_{(i,j)}|}) \times 0.4) = 0.$$ 

3. Our Method

In this section we propose our method. We use a 512 x 512 image to illustrate our method. In order to get more accurate prediction value, we greatly enhance the pixel position of the surrounding pixels from $P_a$ to $P_z$ in Figure 3, instead of Figure 1.

![Figure 3. The new surrounding pixels of pixel (i, j)](image)

Thus, the new prediction of gradient around pixel $(i, j)$ is computed by

$$S_{(i,j)} = (D_{(i,j)}^h + D_{(i,j)}^v) - (D_{(i,j)}^l + D_{(i,j)}^r),$$

where

$$D_{(i,j)}^h = \left\{ |P_x - P_a| + |P_x - P_b| + |P_x - P_c| + |P_x - P_d| \right\}/4,$$

$$D_{(i,j)}^v = \left\{ |P_y - P_a| + |P_y - P_b| + |P_y - P_c| + |P_y - P_d| \right\}/4,$$

$$D_{(i,j)}^l = \left\{ |P_x - P_e| + |P_x - P_f| + |P_x - P_g| + |P_x - P_h| \right\}/4,$$

$$D_{(i,j)}^r = \left\{ |P_y - P_e| + |P_y - P_f| + |P_y - P_g| + |P_y - P_h| \right\}/4,$$

and

$$D_{(i,j)}^l = \left\{ |P_x - P_e| + |P_x - P_f| + |P_x - P_g| + |P_x - P_h| \right\}/6.$$

Here, $D_{(i,j)}^h$, $D_{(i,j)}^v$, $D_{(i,j)}^l$, and $D_{(i,j)}^r$ represent horizontal gradient prediction, vertical gradient prediction, tilted left gradient prediction, and tilted right gradient prediction, respectively. Since the prediction direction and the number of surrounding pixels are different, we introduce the average concept to reduce prediction errors.

Let $\alpha_{(i,j)}$ and $\beta_{(i,j)}$ be the cross and diagonal sets of pixel $(i, j)$ such that

$$\alpha_{(i,j)} = \{(x, y) | x, y \in [0,511], (i-x)^2 + (j-y)^2 = 1\} \quad \text{and} \quad \beta_{(i,j)} = \{(x, y) | x, y \in [0,511], (i-x)^2 + (j-y)^2 = \sqrt{2}\}.$$
We use $S_{(i,j)}$ to compute $D_{(i,j)}$ according to the gradient status as shown in Table 2. Note that
Table 2 shows the experimental weights of gradient types which are adapted to new surrounding
pixels in Figure 3.

### Table 2. New weights of Gradient types

<table>
<thead>
<tr>
<th>Gradient Type</th>
<th>$S_{(i,j)}$</th>
<th>$D_{(i,j)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong Diagonal Similarity</td>
<td>$(12, \infty]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.6 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.8$</td>
</tr>
<tr>
<td>Diagonal Similarity</td>
<td>$(8, 12]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.8 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.8$</td>
</tr>
<tr>
<td>Weak Diagonal Similarity</td>
<td>$(4, 8]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.5 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.5$</td>
</tr>
<tr>
<td>Uniform Similarity</td>
<td>$[4, -4]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.3 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.3$</td>
</tr>
<tr>
<td>Weak 4-Neighbor Similarity</td>
<td>$(-4, -8]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.7 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.7$</td>
</tr>
<tr>
<td>4-Neighbor Similarity</td>
<td>$(-8, -12]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.8 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.8$</td>
</tr>
<tr>
<td>Strong 4-Neighbor Similarity</td>
<td>$(-12, -\infty]$</td>
<td>$\sum \frac{\alpha_{(i,j)}}{\beta_{(i,j)}} \times 1.9 + \sum \frac{\beta_{(i,j)}}{\beta_{(i,j)}} \times 0.9$</td>
</tr>
</tbody>
</table>

Furthermore, we can calculate the prediction errors as

$$D_{(i,j)}' = P_{(i,j)} - D_{(i,j)}.$$  \hspace{1cm} (2-1)

To hide the secret data, we generate the space near the peaks by shifting the prediction error

$$D_{(i,j)}' = \begin{cases} 
D_{(i,j)}' - 1, & \text{if } Z_{1x} < D_{(i,j)}' < H_{1x} \\
D_{(i,j)}' + 1, & \text{if } H_{2x} < D_{(i,j)}' < Z_{2x} 
\end{cases}$$  \hspace{1cm} (2-2)

If $D_{(i,j)}'$ is equal to the peaks, it denote that we can embed the secret data $M_{n}$.

$$D_{(i,j)}'' = \begin{cases} 
D_{(i,j)}', & \text{if } D_{(i,j)}' = H_{1x} \text{ or } H_{12} \text{ and if } M_n = 0 \\
D_{(i,j)}' - 1, & \text{if } D_{(i,j)}' = H_{11} \text{ and if } M_n = 1 \\
D_{(i,j)}' + 1, & \text{if } D_{(i,j)}' = H_{12} \text{ and if } M_n = 1 
\end{cases}$$  \hspace{1cm} (2-3)

We can get two pairs of zero and peak points $(H_{1x}, Z_{1x})$ and $(H_{2x}, Z_{2x})$, where $Z_{1x} < H_{1x} < H_{2x} < Z_{2x}$. Note that $x \in \{1, 2, 3, 4\}$ is the part of segments we progress in cover image. We further use
\[ P'_{(i,j)} = D'_{(i,j)} + D_{(i,j)}. \tag{2-4} \]

to produce the stego-image.

Next, we extract secret data from the stego-image. First, we calculate the prediction errors in stego-image by

\[ D''_{(i,j)} = P_{(i,j)} - D_{(i,j)}. \tag{2-5} \]

From \( D''_{(i,j)} \), \( H_{1x} \), and \( H_{2x} \), we extract secret \( M_n \) by

\[
M_n = \begin{cases} 
1, & \text{if } D''_{(i,j)} = H_{2x} + 1 \text{ or } H_{1x} - 1 \\
0, & \text{if } D''_{(i,j)} = H_{2x} \text{ or } H_{1x} 
\end{cases}
\tag{2-6} \]

To reverse the stego-image to cover image, we shift the prediction errors back by

\[
D'_{(i,j)} = \begin{cases} 
D''_{(i,j)} + 1, & \text{if } D''_{(i,j)} = H_{2x} + 1 \\
D''_{(i,j)} - 1, & \text{if } D''_{(i,j)} = H_{1x} - 1 \\
D''_{(i,j)}, & \text{if } D''_{(i,j)} = H_{2x} \text{ or } H_{1x}
\end{cases}
\tag{2-7} \]

We then shift \( D'_{(i,j)} \) back from histogram by

\[
D'_{(i,j)} = \begin{cases} 
D''_{(i,j)} + 1, & \text{if } Z_{1x} < D''_{(i,j)} < H_{1x} \\
D''_{(i,j)} - 1, & \text{if } H_{1x} < D''_{(i,j)} < Z_{2x}
\end{cases}
\tag{2-8} \]

Finally, we discover the cover image by

\[ P_{(i,j)} = D'_{(i,j)} + D_{(i,j)}. \tag{2-9} \]

![Figure 4. Four segments of cover image](image)

**Input:** Cover image, secret data M.

**Output:** Stego-image, the length of secret data \( L \), \( H_{11} \), \( H_{21} \), \( Z_{11} \), \( Z_{21} \), \( H_{12} \), \( H_{22} \), \( Z_{12} \), \( Z_{22} \), \( H_{13} \), \( H_{23} \), \( Z_{13} \), \( Z_{23} \), \( H_{14} \), \( H_{24} \), \( Z_{14} \) and \( Z_{24} \)

**Step 1.** Read and divide the cover image into four segments, 1, 2, 3, 4, as Figure 4.

**Step 2.** Compute all \( D_{(i,j)} \) in segment 1 by Table 2, and predictive errors \( D'_{(i,j)} \) by Formula (2-1).

**Step 3.** Output the numbers of \( D'_{(i,j)} \) in segment 1 in histogram.
Step 4. Find $H_{11}$, $H_{21}$, $Z_{11}$ and $Z_{21}$ where $Z_{11} < H_{11} < H_{21} < Z_{21}$.

Step 5. Shift the predictive errors by Formula (2-2).

Step 6. Shift the predictive errors to embed the secret data by Formula (2-3).

Step 7. Scan segment 1 again and reverse predictive errors into pixel values by Formula (2-4).

Step 8. If the secret data remain to be embedded, we perform the steps 2-7 in segment 2, 3, 4 in sequence.

Step 9. Find $H_{12}$, $H_{22}$, $Z_{12}$, $Z_{22}$, $H_{13}$, $H_{23}$, $Z_{13}$, $Z_{23}$, $H_{14}$, $H_{24}$, $Z_{14}$ and $Z_{24}$ where $Z_{12} < H_{12} < H_{22} < Z_{22}$, $Z_{13} < H_{13} < H_{23} < Z_{23}$, and $Z_{14} < H_{14} < H_{24} < Z_{24}$.

Transport stego-image, $L$, $H_{11}$, $H_{21}$, $Z_{11}$, $Z_{21}$, $H_{12}$, $H_{22}$, $Z_{12}$, $Z_{22}$, $H_{13}$, $H_{23}$, $Z_{13}$, $Z_{23}$, $H_{14}$, $H_{24}$, $Z_{14}$ and $Z_{24}$ to the receiver.

**Figure 5. Our Embedded algorithm**

| Input: Stego-image, $L$, $H_{11}$, $H_{21}$, $Z_{11}$, $Z_{21}$, $H_{12}$, $H_{22}$, $Z_{12}$, $Z_{22}$, $H_{13}$, $H_{23}$, $Z_{13}$, $Z_{23}$, $H_{14}$, $H_{24}$, $Z_{14}$ and $Z_{24}$. |
| Output: Secret data, cover image. |
| Step 1. Read and divide stego-image into four segments as Figure 4, and process these segments 4, 3, 2, 1, in order. |
| Step 2. Output the number of $D''_{(i,j)}$ by Table 2 in segment 4. |
| Step 3. Extract the secret data by Formula (2-6). |
| Step 4. Shift the predictive errors back by Formula (2-7), and Formula (2-8). |
| Step 5. Reverse cover image by Formula (2-9). |
| Step 6. If the secret data remain to be extracted, we perform the steps 2-6 in segments 3, 2, 1 in sequence. |
| Step 7. Extracted all secret data and output secret data M and cover image. |

**Figure 6. Our extracted algorithm**

4. **Experiment Result**

In this section, we compare Jheng et al.’s method with ours by using with 3 grayscale images as Lena, boat, and baboon. We give Table 3 and 4 to show our experiment result. In Table 4, our method increases the amount of hidden secret data by 12.5% on the average as compared with Jheng et al.’s method. At the same time, our method keeps higher PSNR value in Table 4.

**Table 3. Secret data capacity**

<table>
<thead>
<tr>
<th>Test image</th>
<th>Snow forest</th>
<th>Aurora</th>
<th>Sunlight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jheng et al.’s method</td>
<td>87625</td>
<td>147468</td>
<td>186829</td>
</tr>
<tr>
<td>Our method</td>
<td>99101</td>
<td>171001</td>
<td>204715</td>
</tr>
<tr>
<td>The rate of progress</td>
<td>13%</td>
<td>15%</td>
<td>9.5%</td>
</tr>
</tbody>
</table>
Table 4. PSNR

<table>
<thead>
<tr>
<th>Test image</th>
<th>Snow forest</th>
<th>Aurora</th>
<th>Sunlight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jheng et al.’s method</td>
<td>55.17</td>
<td>57.53</td>
<td>59.14</td>
</tr>
<tr>
<td>Our method</td>
<td>56.90</td>
<td>58.34</td>
<td>61.51</td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper we have presented a new data hiding method based on histogram by using GAP. As compared with Jheng et al.’s method our method keeps higher quality and higher capacity. According to experiment results of 3 grayscale images, our method increases the amount of hidden data by 12.5% on the average as compared with Jang et al.’s method.

Acknowledgment

The work was supported by the Research Support Scheme of the National Science Council, NSC NO. 101-2221-E390-034

Reference