Adaptive Pixel Value Grouping for Protecting Secret Data in Public Computer Networks

Hendro E. Prabowo and Tohari Ahmad
Department of Informatics, Institut Teknologi Sepuluh Nopember, Kampus ITS, Surabaya, 60111, Indonesia
Email: hendro16@mhs.if.its.ac.id, tohari@if.its.ac.id

Abstract—Rapid development of information and communication technology has increased the need of protecting sensitive data from illegal access during their transmission. One of popular methods to secure such data is by embedding them in another file such as a digital image, which is called as data hiding. This involves processing values of pixels that causes image distortion. This noise has become one of common problems in data hiding, along with the capacity of the secret data, which can be carried by the image for example. Some methods have been introduced in order to alleviate these two problems. Nevertheless, this is still challenging. In this paper, we propose a new data hiding mechanism, which is developed based on the difference between neighboring pixels to construct pixel value grouping, called adaptive pixel value grouping (APVG). This scheme selects pixel group head which is then compares it with other pixel to get difference value. This difference determines how the pixels in the image are grouped. This resulted group, if any, is used for further embedding process. The experimental result shows that APVG scheme is able to achieve a better quality, which is depicted in PSNR value, than the existing method. This proposed method achieves 65–72dB, 54–61dB and 44–55dB of PSNR for 1kb, 10kb and 100kb payload (secret data), respectively.

Index Terms—Data hiding, data security, protecting data, information security, secret data

I. INTRODUCTION

The need of integrity and confidentiality of data has increased along with the significant development of the information technology. Data must be protected before being sent to other parties in a network without being worried about illegal access of unauthorized users. In order to achieve this secure condition, some methods have been introduced, such as data hiding. In the implementation level, this can be combined with other securing methods, for example cryptography. That combination step are taken with aim of providing more information protection in terms of confidentiality and reliability [1].

Data hiding itself is a process to hide a secret message or data in another media called carrier or cover such as digital image. This process involves mathematic algorithms to modify pixels which results to stego object [2]-[4]. However, there must not be significant change to the pixels, which may cause noises. On the other hand, the embedding algorithm should provide a high embedding capacity [5]-[7]. These two factors have been common problems in developing a data hiding algorithm. Many data hiding methods have been developed, for example Different Expansion (DE) [8] and Histogram-based embedding [2]. The DE method, which is proposed by Tian [8], takes advantages of difference between neighboring pixels. Here, the different value is added with secret data, which has been converted to binary. This algorithm is able to hide secret data 0.5 bit per pixel (bpp) on average. Because of this relatively low performance, some research is performed to explore some aspects of DE, for example generalization of integer transform [9][10], reduction of location map [11], [12], prediction error expansion [13]-[15], pixel selection embedding [16] and pixel value ordering [17], [18].

Different from DE, histogram-based method which is proposed by Rad et al [2], employs histogram to embed data in an image. In this method, the histogram is the basis of their proposed Adaptive Group Modification (AGM). Additionally, this technique is carried out by minimizing the shifting of the histogram and maximizing the elements of hiding. There are more bits which can be carried by the cover that is shown by the experimental results. In certain cover images, it achieves 4.3 dB of quality improvement. Overall, this method has proven to be a reversible technique.

Recently, histogram-based data hiding has been widely developed to get optimal embedding result. In [19]-[21], some research explores histogram of an image. It is found that the difference of adjacent pixels whose value is same, is likely to have high capacity. Another development of histogram-based method is proposed in [22]-[24] by using sub-images (image pixel blocks) to build histogram and to get difference between adjacent pixels. This method is able to generate a high embedding capacity with low distortion level.

In [25], Pixel Value Grouping (PVG) approach is used to develop histogram-based embedding. In this case, PVG groups some pixels in a sub-image based on its specified embedding level (EL). This level (threshold) acts as a basis for deciding whether a pixel should join to the corresponding group or not. By using this method, possible distortion can be reduced, or even lower than other histogram-based methods. Nevertheless, this
method has some issues. First, if the pixel block has higher difference than EL, then the pixel cannot be grouped in the block, so it is useless. Actually, this pixel may be put in another block. Second, the use of the reference value to be the basis of creating pixel blocks. This has made the quality of the stego image depending on the reference value. In facts, there is no criterion what the reference should be used to get a good stego image.

In order to solve that problem, in this paper we propose a method which is able to categorize pixels, called adaptive pixel value grouping (APVG). This grouping is performed according to the value of EL, which can control the level of distortion. The average of each pixel block is used to determine the difference that accommodates the binary data. This is intended to raise the quality of stego image because the change of pixel is not more than twice of EL.

This paper is structured as follows. Section 2 describes PVG, a method which we intend to improve. Section 3 depicts the proposed method (APVG). The experimental results of the proposed method is provided in Section 4. Finally, this paper is summarized in Section 5.

II. PIXEL VALUE GROUPING

Pixel Value Grouping (PVG) [25] is developed based on the blocks of pixels with specified size, for example 4×4, as previously described. Firstly, pixel in the block is sorted in ascending mode whose result is then classified into 3 types: \( P_{L}, P_{R} \) and \( P_{D} \), where \( P_{L} \) and \( P_{R} \) refer to the pixels whose value is close to the smallest and the highest reference value, respectively; while \( P_{D} \) refers to pixels which do not include in \( P_{L}, P_{R} \).

In the next process, \( P_{L} \) and \( P_{R} \) check every pixel value in the block; it is from the highest to the lowest in the case of \( P_{R} \) and from the lowest to the highest in the case of \( P_{L} \). This checking process stops if the checked pixel exceeds the threshold or the pixel is a member of \( P_{L} \) or \( P_{R} \). Here, the difference is defined as the difference between the pixel being checked and the reference value; while the threshold is EL, which has been determined at the beginning.

A. Data Embedding

The data embedding process starts with processing all pixels in a block. The steps are as follows:

1) Dividing image pixel into block: Let \( I \) be the cover image; it can be divided into \( u \times v \) blocks, where \( u \) and \( v \) is the number of row and column of \( I \), respectively.

2) Block classification: Each block resulted from step 1 is classified according to the difference which has been previously described. From the classification process we have \( P_{L}, P_{R} \) and \( P_{D} \) groups.

3) Difference generation: This is to determine real or virtual pixel, which is used as the reference to calculate the difference between pixels. The procedure to obtain a real or virtual pixel is in [25].

4) Embedding data: Before the secret message is hidden in pixels, the area \([b_{EL+1}, b_{2\times EL+1}]\) and \([b_{-2\times EL-1}, b_{-EL-1}]\) of histogram have to be cleared. Here, \( b \) is the block of pixels being used. By considering \( P_{L} \) and \( P_{R} \), the embedding is performed as in (1), where \( P_{w}^w \) is the pixel of either \( P_{L} \) or \( P_{R} \) (denoted as \( P_{w}^w \)) which has been embedded by the message \( w : P_{ref1} \) is the pixel reference of \( P_{L} \).

\[
P_{w}^w = P_{w} + (P_{w} - P_{ref1}) - w
\]  

(1)

This formula is used when the difference between pixel in \( P_{L} \) and the reference of \( P_{L} \) is more than or equal to \(-EL\) and less than 0. If this condition does not meet, then (2) is used. For this, the difference between the pixel in \( P_{L} \) and the respective reference must be 0.

\[
P_{w} = P_{w} - w
\]  

(2)

In the case that those in \( P_{R} \) meet the requirements, and the difference between them and the respective reference is less than or equal to EL, the condition in (3) is applied. Here, \( P_{ref2} \) is the pixel reference of \( P_{R} \).

\[
P_{w}^w = P_{w} + (P_{w} - P_{ref2}) + w
\]  

(3)

Otherwise, the embedding process is carried out by using (4). Nevertheless, the difference between pixels in \( P_{R} \) and the reference is 0.

\[
P_{w} = P_{w} + w
\]  

(4)

B. Data Extraction and Image Reconstruction

After the stego image \( I' \) is divided into blocks of \( u \times v \), the extraction is firstly performed by classifying those blocks according some criteria. Once the type of the blocks has been determined, then the reference of \( P_{R} \) and \( P_{L} \) are specified by using (5) and (6), respectively. In this case, \( P_{v-ref} \) represents virtual pixel reference, which is used when \( P_{R} \) or \( P_{L} \) is full; \( P_{w}^{\sigma(k)} \) and \( P_{w}^{\sigma(\ell)} \) represent the last pixel of \( P_{R} \) or \( P_{L} \) of the respective stego image.

\[
P_{v-ref} = P_{w}^{\sigma(\ell)} - 2 \times (EL + 1)
\]  

(5)

\[
P_{v-ref} = P_{w}^{\sigma(\ell)} + 2 \times (EL + 1)
\]  

(6)

Once those references have been found, the secret message in \( P_{L} \) can be obtained by using (7).

\[
w = \begin{cases} 
1, & \text{if } P_{w}^{\sigma(k)} - P_{ref1} \in [-2 \times EL - 1, -2(EL - 1), \ldots, -3] \\ 0, & \text{if } P_{w}^{\sigma(k)} - P_{ref1} \in [-2 \times EL - 2(EL - 1), \ldots, -2] \end{cases}
\]  

(7)

In case \( P_{L} \) does not have full member, the formula in (8) is employed.

\[
w = \begin{cases} 
1, & \text{if } P_{w}^{\sigma(k)} - P_{ref1} = -1 \\ 0, & \text{if } P_{w}^{\sigma(k)} - P_{ref1} = 0 \end{cases}
\]  

(8)

For \( P_{R} \), the secret is extracted by using (9) and (10) for full and not full groups, respectively.
Recovering the cover is carried out after the secret message has been fully extracted. The original pixel is generated by implementing (11).

\[
w = \begin{cases} 
1, & \text{if } P_{\sigma(k)}^w - P_{ref2} = 1 \\
0, & \text{if } P_{\sigma(k)}^w - P_{ref2} = 0 
\end{cases}
\]  

(9)

(10)

C. Preventing Overflow or Underflow

In some cases, the resulted stego pixel is more than 255 or less than 0, which respectively called overflow and underflow. In [25], this condition is prevented by using a location map for marking the blocks. The location map is set to 1 if a block contains either overflow or underflow pixel. Otherwise, the location map is set to 0. In the embedding, both overflow and underflow pixels are not used.

III. RESEARCH METHOD

A. Adaptive Pixel Value Grouping

Adaptive pixel value grouping (APVG) is an improvement of previous PVG. In PVG, a pixel is directly grouped to a block, which has been defined; while in APVG, the difference between the pixel group head \( (PG_h) \) and its horizontal neighboring pixel is utilized. For this purpose, a cover image is scanned horizontally to have a set of pixel \( \{ p_1, p_2, p_3, ..., p_n \} \). In addition, the value of \( EL \) acts as the threshold for grouping the pixels. An illustration of this process is presented in Fig. 1.

It is shown in Fig. 1 that the scheme starts by taking an \( i^{th} \) pixel of the carrier \( (P_i) \) which is to be the pixel group head \( (PG_h) \). So, at the beginning, \( P_i = PG_h \). The difference \( d \) between \( PG_h \) and its next neighboring pixel \( (P_{i+1}) \) is calculated by implementing (12). If this \( d \) meets (13), then the respective pixel \( (P_{i+1}) \) has become pixel group body \( (PG_b) \) and grouped with \( PG_h \).

![Fig. 1. The scheme of adaptive pixel value grouping](https://example.com/fig1.png)

©2018 Journal of Communications
This process is repeated until the difference between $PG_n$ and $P_{i+n}$ is more than $EL$, where $n$ is the sequence of the pixel after $P_1$ is set as $PG_1$. This means that (13) is not fulfilled. In this condition, APVG calculates the number of pixels that grouped with $PG_n$. If the number of pixels ($PG$ size) is more than 1, then the last joint pixel ($P_{i+n}$) is set as Pixel Group Tail ($PG_t$). The pixel next to $PG_t$ is set as $PG_n$ of the new group and grouping process re-start. This condition is shown in (14). If a pixel group ($PG$) has more than 1 member, then this respective group has at least $PG_n$ and $PG_t$.

$$d = |P_{i+1} - PG_n|$$  \hspace{1cm} (12)

$$PG_n = P_{i+n}, \quad \text{if} \ d \leq EL$$  \hspace{1cm} (13)

If (14) is not fulfilled, then $PG_n$ is set as non-pixel value grouping which is not used for embedding. If this condition happens, then APVG has a pixel next to $PG_n$ which acts as a new $PG_n$; and the pixel grouping process is re-started.

### B. Data Embedding

As in other data hiding methods, the secret message is converted to binary digits before being embedded to the cover. The overall process of this embedding is depicted in Fig. 2 whose detail steps are explained as follows.

1) **Pixel Grouping**: pixels in the cover image are grouped. Different from PVG that the size of blocks is fixedly specified, in APVG this size is adaptive depending on the cover being used as shown in Fig. 3. Therefore, the size may differ from one to other blocks. Here, this size is determined by the EL and the length of the secret message to be hidden.
2) Data Embedding: the embedding uses the average of the pixels in a block, as shown in (15) and (16) as an improvement of (1), where \( P_{\text{avg}} \) is the average of pixels group; \( P_i \) is \( i^{th} \) pixel in a group and \( n \) is the amount of pixels in the respective group. \( P_i^m \) is the \( i^{th} \) new pixel containing message \( m \).

\[
P_{\text{avg}} = \left[ \frac{\sum P_i}{n} \right]
\]

(15)

\[
P_i^m = P_i + (P_i - P_{\text{avg}}) + m
\]

(16)

In the case that \( (P_i - P_{\text{avg}}) = 0 \), we can use (17) to embed message.

\[
P_i^m = P_i + m
\]

(17)

3) Location Map: In this APVG, the location map (LM) is implemented for tagging the status of a pixel in a group. \( PG_r \) and \( PG_b \) are set as 1, while \( PG_c \) is 0 in the location map. Furthermore, LM is also used for recognizing the last group. That is, if a group does not have \( PG_r \) or LM has only value of 1, then we can conclude that it is the last group.

4) Generate Stego Image: The stego image is constructed according to the new pixels, which have been obtained by using (16) or (17). In general, this embedding process can be illustrated in an example, which is depicted in Fig. 4. Let \( EL = 2 \), and the message is 101001010111011001100000000010100011111. Fig. 4(a) depicts the effect of pixel grouping; while Fig. 4(b) shows the difference between the average of pixel values \( (P_{\text{avg}}) \) and \( i^{th} \) pixel \( (P_i) \) in the relevant pixel group. The bits in Fig. 4(c) is to generate new pixels in Fig. 4(d). The location map before and after embedding process in Figs. 4(e) and 4(f), differs in the last block of the image (bottom-right pixel). Before embedding, the last \( PG \) has 110 \( (PG_{16}, PG_{15}, \text{and } PG_{14}) \) but it has become only 11 after embedding. This is because the message to be embedded is less than the number of pixels in the last block.

5) Group Type: Different from [25], we classify the groups of pixels into two categories: full and not-full groups. Full group means that the group contains \( PG_r \), \( PG_b \) and \( PG_c \); while not-full contains only \( PG_r \) and \( PG_b \).

Fig. 4. An example of data embedding process

C. Data Extraction and Image Reconstruction

In the extraction process, there are three data are required: stego image, location map and average value. The latest value is the average of pixel values of each group. It is assumed that this value is stored in the stego file, such that the receiver does not have to recalculate it.

The secret data extraction and the pixel reconstruction are carried out by using (18) and (19) respectively. All
pixels which are generated from this step reconstruct the original cover image.

\[ m = P_t^m + P_{avg} \mod 2 \]  \hspace{1cm} (18)

\[ P_t = \frac{P_t^m + P_{avg} - m}{2} \]  \hspace{1cm} (19)

D. Preventing Overflow and Underflow

Similar to [25], the overflow and underflow problems are avoided by using the location map. Both conditions are represented by 0 in the map; so, it can be used for a sign that the respective location is not further processed, each pixel in that group is set as non-pixel value grouping.

![Image](image.png)

Fig. 5. Comparison of before embedding, after embedding and after reconstruction: (a) Cover images; (b) Stego image. (c) Reconstructed images

### Table I. Amount of Reconstructed Pixels with Various Embedding Levels (EL) and Message Sizes

<table>
<thead>
<tr>
<th>Image</th>
<th>Message Size (kb)</th>
<th>Amount of reconstructed pixels (%)</th>
<th>Embedding Level (EL)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Lena</td>
<td>1</td>
<td>0.99729</td>
<td>0.99711</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.9738</td>
<td>0.97229</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.73872</td>
<td>0.72304</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.99741</td>
<td>0.99733</td>
</tr>
<tr>
<td>Baboon</td>
<td>10</td>
<td>0.97401</td>
<td>0.973</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.9737</td>
<td>0.9718</td>
</tr>
<tr>
<td>Boat</td>
<td>10</td>
<td>0.974</td>
<td>0.97255</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.972687</td>
<td>0.71096</td>
</tr>
<tr>
<td>Elaine</td>
<td>10</td>
<td>0.99761</td>
<td>0.99727</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.9751</td>
<td>0.97281</td>
</tr>
<tr>
<td>Pepper</td>
<td>10</td>
<td>0.97398</td>
<td>0.97226</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.76569</td>
<td>0.72554</td>
</tr>
</tbody>
</table>

### Table II. Comparison of the PSNR Value between the Proposed Method and [22] with Various Embedding Levels (EL) and Message Size

<table>
<thead>
<tr>
<th>Image</th>
<th>Message Size (kb)</th>
<th>PSNR (dB)</th>
<th>EL=1</th>
<th>EL=2</th>
<th>EL=3</th>
<th>EL=4</th>
<th>EL=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>1</td>
<td>71.13</td>
<td>65.26</td>
<td>69.59</td>
<td>63.56</td>
<td>67.59</td>
<td>62.65</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>61.25</td>
<td>55.46</td>
<td>59.57</td>
<td>53.69</td>
<td>57.57</td>
<td>52.58</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>51.31</td>
<td>46.95</td>
<td>49.78</td>
<td>43.52</td>
<td>47.88</td>
<td>42.27</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>71.61</td>
<td>64.34</td>
<td>70.94</td>
<td>61.81</td>
<td>69.34</td>
<td>59.84</td>
</tr>
<tr>
<td>Baboon</td>
<td>10</td>
<td>61.51</td>
<td>54.47</td>
<td>60.75</td>
<td>51.94</td>
<td>58.91</td>
<td>50.04</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>48.33</td>
<td>-</td>
<td>45.21</td>
<td>-</td>
<td>42.79</td>
<td>-</td>
</tr>
<tr>
<td>Boat</td>
<td>10</td>
<td>71.26</td>
<td>65.32</td>
<td>69.67</td>
<td>63.32</td>
<td>68.15</td>
<td>62.07</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>61.35</td>
<td>55.14</td>
<td>59.81</td>
<td>53.22</td>
<td>57.73</td>
<td>51.90</td>
</tr>
<tr>
<td>Elaine</td>
<td>10</td>
<td>61.83</td>
<td>55.31</td>
<td>60.09</td>
<td>53.45</td>
<td>58.11</td>
<td>52.00</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>47.68</td>
<td>50.26</td>
<td>44.46</td>
<td>48.36</td>
<td>41.99</td>
<td>46.77</td>
</tr>
</tbody>
</table>

IV. RESULT AND DISCUSSION

In order to evaluate the method, we generate random data as the secret whose size is 1kb, 10kb, and 100kb. For the carrier, we use those from USC-SIPI image database [26]: Lena, Baboon, Boat, Elaine and Peppers. An example of cover, stego and reconstruction images is shown in Fig. 5, which are in grey level images. It is depicted that overall, those three types of images are relatively similar. That is, it is subjectively hard to find differences between them. In this case, Fig. 5(c) must be exactly same with Fig. 5(a) while Fig. 5(b) may be similar to Fig. 5(a), which means that the proposed algorithm is reversible.
shown that lower EL value results to higher number of and size of the message whose result is in Table III. It is and maximum size of groups based on the value of EL method is better than that of [25].

quality of the stego image generating by the proposed method is 100kb with smaller EL. Nevertheless, the overall proposed method cannot hide the secret data whose size is 100kb with smaller EL. Furthermore, if the number of pixel resulting from grouping process is less than length of message, this respective message cannot be embedded. When the level of EL is 1, 2 or 3, the embedding is unsuccessful, which means that no pixel recovery can be done.

A. Message Extraction and Restoring Pixels

Experimental results show that all secret data can be fully extracted. In more details, this stage has been able to recover all messages. This means that this proposed algorithm is reversible.

In terms of image reconstruction, the proposed algorithm has various levels, as depicted in Table I. It is shown that, as predicted, raising the message size leads to reducing the successfulness level. Similarly, increasing the value of EL also decreases the reconstruction results. In addition, these two factors: message size and EL value, affect the embedding and its recovering as provided in Table II. When the level of EL is 1, 2 or 3, the embedding is unsuccessful, which means that no pixel recovery can be done.

B. Quality of Stego Image

In order to evaluate the quality of the stego image, we use PSNR value, similar to other research. This measurement is provided in Table II, which shows the PSNR of our proposed method and that of [25], which is the main reference for the comparison purpose.

In Table II, we find that in certain condition, the proposed method cannot hide the secret data whose size is 100kb with smaller EL. Nevertheless, the overall quality of the stego image generating by the proposed method is better than that of [25].

For further analysis, we count the number of minimum and maximum size of groups based on the value of EL and size of the message whose result is in Table III. It is shown that lower EL value results to higher number of groups. Moreover, most of these groups are relatively small. This condition leads to low variation value of pixels within the group. As a result, the difference between the value of a pixel in the group and the average value of the respective group is also small. On the contrary, higher variation value between those values leads to higher difference. This condition has an effect on the quality of the resulted stego image. In the case of APVG, the change of the pixel after the embedding is always less than twice of the respective EL.

V. CONCLUSION

In this research, a method of data hiding has been proposed. It groups the pixels in an image adaptively, which does not need to specify the size fixedly. This means that the method dynamically adjust the size of pixels, according to the characteristic of the respective image. Additionally, the size of the message and the threshold value influence the performance.

The results of the experiment show that this method is able to generate higher stego image quality than the fixed one. It can be inferred that this technique has lower distortion. This condition results to better quality of the stego image than the existing method.

REFERENCES


