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

Manuscript received November 7, 2017; revised February 5, 2018.
This work was supported by Kemenristekdikti of Republic of Indonesia and ITS.
Corresponding author email: tohari@if.its.ac.id.
doi:10.12720/jcm.13.6.325-332

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_O, 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_0 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 l 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 l, 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_0 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 $[b_{-2\times EL+1}, b_{-EL-1}]$ and $[b_{EL+1}, b_{2\times EL+1}]$ 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 $P^w_{\sigma(k)}$ is the pixel of either P_L or P_R (denoted as $P_{\sigma(k)}$) which has been embedded by the message w ; P_{ref1} is the pixel reference of P_L.

$$P^w_{\sigma(k)} = P_{\sigma(k)} + \left(P_{\sigma(k)} - P_{ref1}\right) - w \quad (1)$$

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.

$$P^w_{\sigma(k)} = P_{\sigma(k)} - w \quad (2)$$

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.

$$P^w_{\sigma(k)} = P_{\sigma(k)} + \left(P_{\sigma(k)} - P_{ref2}\right) + w \quad (3)$$

Otherwise, the embedding process is carried out by using (4). Nevertheless, the difference between pixels in P_R and the reference is 0.

$$P^w_{\sigma(k)} = P_{\sigma(k)} + w \quad (4)$$

B. Data Extraction and Image Reconstruction

After the stego image l' 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; $P^w_{\sigma(n)}$ and $P^w_{\sigma(1)}$ represent the last pixel of P_R or P_L of the respective stego image.

$$P_{v-ref} = P^w_{\sigma(1)} - 2 \times (EL + 1) \quad (5)$$

$$P_{v-ref} = P^w_{\sigma(1)} + 2 \times (EL + 1) \quad (6)$$

Once those references have been found, the secret message in P_L can be obtained by using (7).

$$w = \begin{cases} 1, & if\ P^w_{\sigma(k)} - P_{ref1} \in \{-2 \times EL - 1, -2(EL-1), \dots, -3\} \\ 0, & if\ P^w_{\sigma(k)} - P_{ref1} \in \{-2 \times EL, -2(EL-1), \dots, -2\} \end{cases} \quad (7)$$

In case P_L does not have full member, the formula in (8) is employed.

$$w = \begin{cases} 1, & if\ P^w_{\sigma(k)} - P_{ref1} = -1 \\ 0, & if\ P^w_{\sigma(k)} - P_{ref1} = 0 \end{cases} \quad (8)$$

For P_R, the secret is extracted by using (9) and (10) for full and not full groups, respectively.

$$w = \begin{cases} 1, & if\ P^{w}_{\sigma(k)} - P_{ref2} \in \{2 \times EL + 1, 2(EL-1) + 1, \dots, 3\} \\ 0, & if\ P^{w}_{\sigma(k)} - P_{ref2} \in \{2 \times EL, 2(EL-1), \dots, 2\} \end{cases} \quad (9)$$

$$w = \begin{cases} 1, & if\ P^{w}_{\sigma(k)} - P_{ref2} = 1 \\ 0, & if\ P^{w}_{\sigma(k)} - P_{ref2} = 0 \end{cases} \quad (10)$$

Recovering the cover is carried out after the secret message has been fully extracted. The original pixel is generated by implementing (11).

$$P_{\sigma(k)} = \begin{cases} P_{ref2} + \left\lfloor \frac{P^{w}_{\sigma(k)} - P_{ref2}}{2} \right\rfloor, & if\ 0 \le P^{w}_{\sigma(k)} - P_{ref2} \le 2 \times EL + 1 \\ P^{w}_{\sigma(k)} - (EL+1), & if\ P^{w}_{\sigma(k)} - P_{ref2} > 2 \times EL + 1 \\ P_{ref1} + \left\lfloor \frac{P^{w}_{\sigma(k)} - P_{ref1} + 1}{2} \right\rfloor, & if\ -2 \times EL - 1 \le P^{w}_{\sigma(k)} - P_{ref1} \le 0 \\ P^{w}_{\sigma(k)} + (EL+1), & if\ P^{w}_{\sigma(k)} - P_{ref1} < -2 \times EL - 1 \end{cases} \quad (11)$$

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 $\{p_1, p_2, p_3, \dots, p_n\}$. 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.

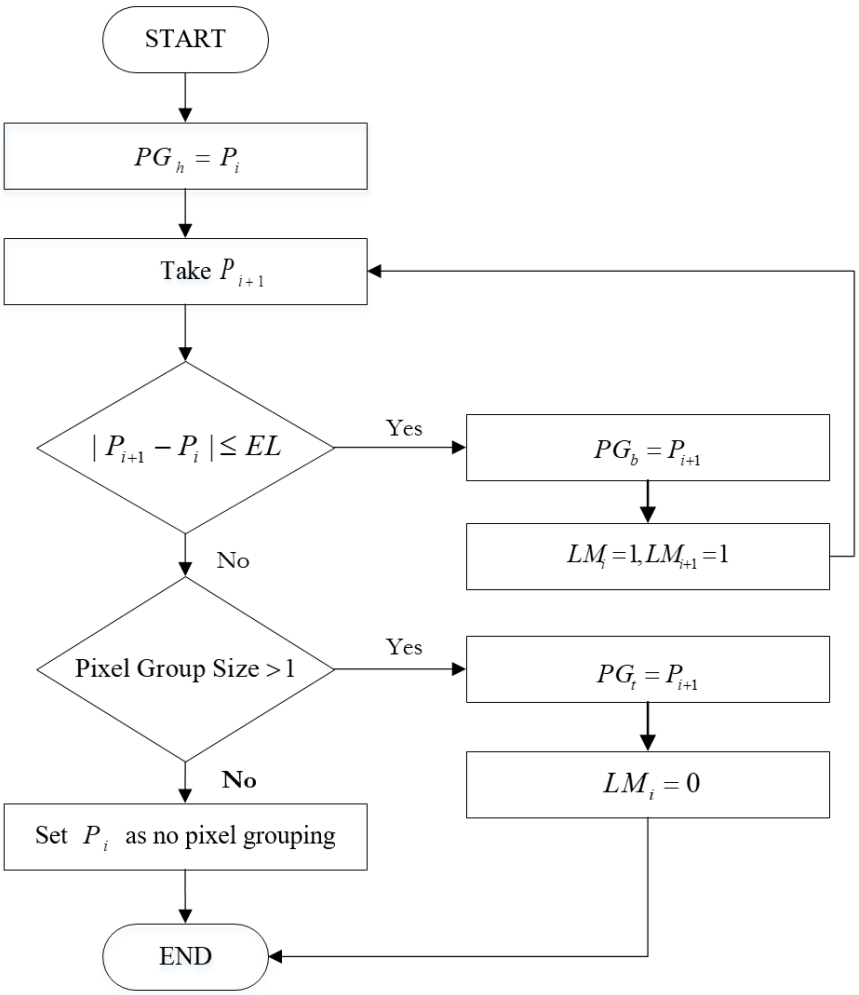


Fig. 1. The scheme of adaptive pixel value grouping

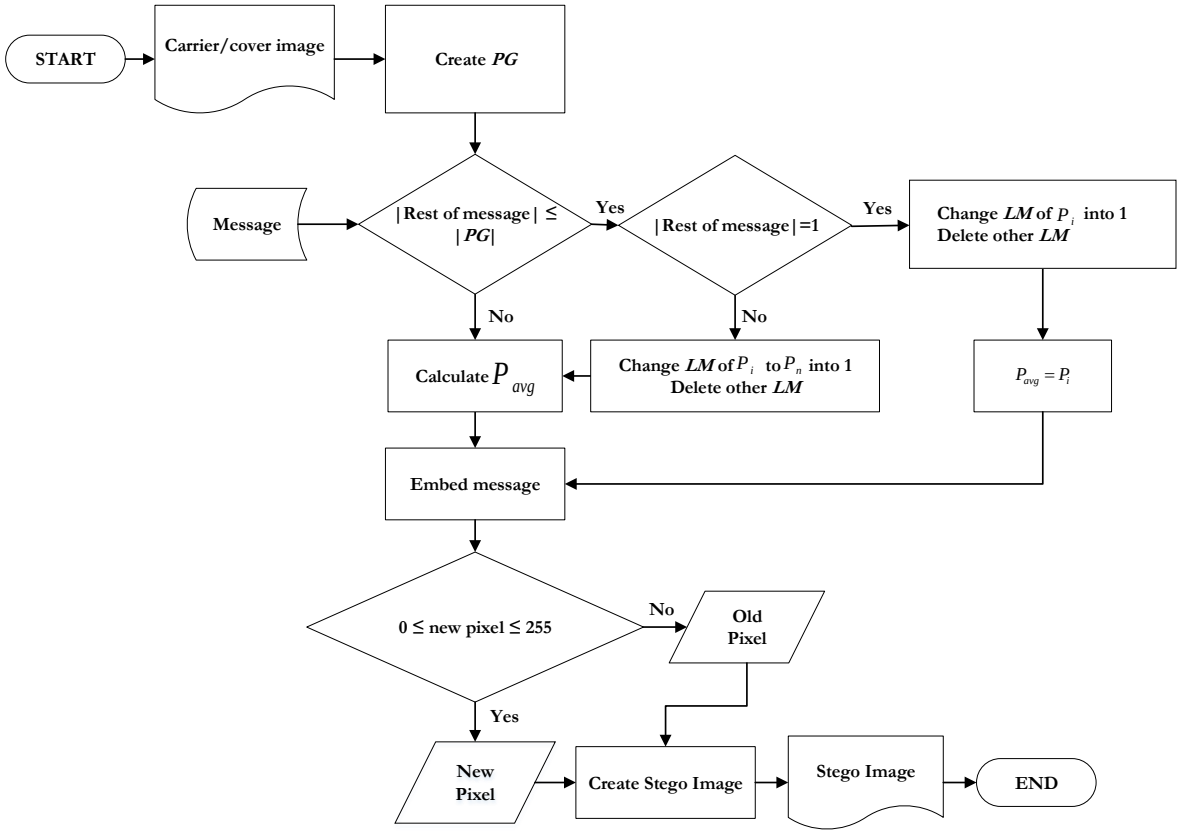
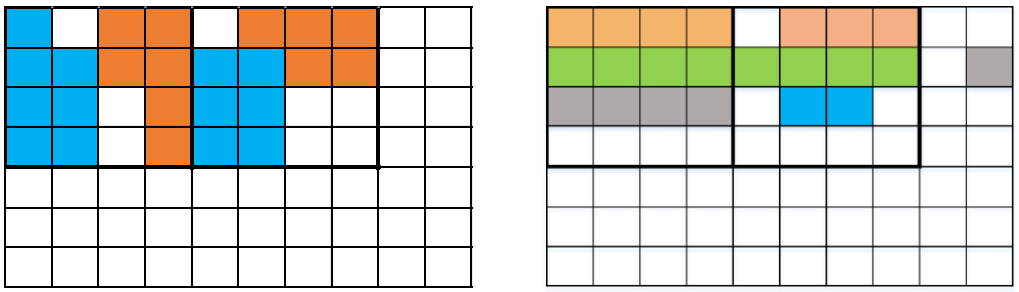


Fig. 2. Data embedding process



(a) (b)

Fig. 3. Pixel grouping: (a) Pixel Value Grouping (PVG), (b) Adaptive Pixel Value Grouping (APVG)

$$d = |P_{i+1} - PG_h| \quad (12)$$

$$PG_b = P_{i+n}, \quad if\ d \leq EL \quad (13)$$

This process is repeated until the difference between PG_h and P_{i+n} is more than EL, where n is the sequence of the pixel after P_i is set as PG_h. This means that (13) is not fulfilled. In this condition, APVG calculates the number of pixels that grouped with PG_h. If the number of pixels ($PG\ size$) is more than 1, then the last joint pixel (P_{i+l}) is set as Pixel Group Tail (PG_t). The pixel next to PG_t is set as PG_h 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_h and PG_t.

$$PG_t = P_{i+l}, \quad if\ d > EL\ \&\ PG\ size > 1 \quad (14)$$

If (14) is not fulfilled, then PG_h 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_h which acts as a new PG_h; 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_{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.

$$P_{avg} = \left\lfloor \frac{\sum_{i=1}^{n} P_i}{n} \right\rfloor \quad (15)$$

$$P_i^m = P_i + \left(P_i - P_{avg}\right) + m \quad (16)$$

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

$$P_i^m = P_i + m \quad (17)$$

3) Location Map: In this APVG, the location map (LM) is implemented for tagging the status of a pixel in a group. PG_h and PG_b are set as 1, while PG_t 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_t 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 be 2, and the message is 10100101011101100110000000010100011111. Fig. 4(a) depicts the effect of pixel grouping; while Fig. 4(b) shows the difference between the average of pixel values (P_{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_h, PG_b, and PG_t) 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_h, PG_b and PG_t; while not-full contains only PG_h and PG_b.

162	162	162	161	162	157	163
162	162	162	161	162	157	163
155	155	158	158	159	160	163
155	155	157	158	155	154	155
156	156	156	160	156	155	163
164	164	158	155	151	159	159
160	160	163	158	160	162	159

(a)

1	1	1	0	1		1
0	0	0	-1	0		
0	0	0	0	1	2	
0	0	2		0	-1	0
1	1	1		1	0	0
1	1			1	-1	-1
0	0	3		0	2	-1

(b)

1	0	1	0	0		1
0	1	0	1	1		
1	0	1	1	0	0	
1	1	0		0	0	0
0	0	0		0	0	1
0	1			0	0	0
1	1	1		1	1	

(c)

164	163	164	161	163	157	165
162	163	162	161	163	157	163
156	155	159	159	160	162	163
156	156	153	158	155	153	155
157	157	157	160	157	155	165
165	165	158	155	162	160	160
162	162	164	158	161	165	159

(d)

1	1	1	1	0		1
1	1	1	1	0		
1	0	1	1	1	0	
1	1	0		1	1	1
1	1	0		1	0	1
1	0			1	1	1
1	1	0		1	1	0

(e)

1	1	1	1	0		1
1	1	1	1	0		
1	0	1	1	1	0	
1	1	0		1	1	1
1	1	0		1	0	1
1	0			1	1	1
1	1	0		1	1	

(f)

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.

$$m = P_i^m + P_{avg} \bmod 2 \quad (18)$$

$$P_i = \frac{P_i^m + P_{avg} - m}{2} \quad (19)$$

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.

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.

(a) (b) (c)

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

Image	Message Size (kb)	Amount of reconstructed pixels (%) Embedding Level (EL)				
		1	2	3	4	5
Lena	1	0.99729	0.99711	0.99698	0.99629	0.99698
	10	0.9738	0.97229	0.97055	0.96893	0.968
	100	0.73872	0.72304	0.7085	0.69123	0.67975
Baboon	1	0.99741	0.99733	0.9973	0.99705	0.99689
	10	0.97401	0.973	0.9722	0.96991	0.96838
	100	-	-	-	0.69643	0.68255
Boat	1	0.9973	0.99718	0.99701	0.99685	0.9677
	10	0.974	0.97255	0.97082	0.96882	0.96749
	100	-	0.72687	0.71096	0.69246	0.6797
Elaine	1	0.99761	0.99727	0.99714	0.9969	0.99673
	10	0.9751	0.97281	0.97123	0.9694	0.96828
	100	-	0.72796	0.71492	0.69526	0.6811
Pepper	1	0.99725	0.99718	0.99697	0.99697	0.99686
	10	0.97398	0.97226	0.97083	0.96901	0.96814
	100	0.76569	0.72554	0.71001	0.6917	0.68006

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

Image	Message Size (kb)	PSNR (dB)									
		EL=1		EL=2		EL=3		EL=4		EL=5	
		Prop.	[22]	Prop.	[22]	Prop.	[22]	Prop.	[22]	Prop.	[22]
Lena	1	71.13	65.26	69.59	63.56	67.59	62.65	66.62	61.74	65.83	61.63
	10	61.25	55.46	59.57	53.69	57.57	52.58	56.06	51.90	54.89	51.64
	100	51.31	46.95	49.78	43.52	47.88	42.27	46.33	41.34	45.00	40.66
Baboon	1	71.61	64.34	70.94	61.81	69.34	59.84	68.59	58.26	67.11	57.05
	10	61.51	54.47	60.75	51.94	58.91	50.04	57.51	48.56	56.00	47.38
	100	-	48.33	-	45.21	-	42.79	47.07	40.81	45.60	39.12
Boat	1	71.26	65.32	69.67	63.32	68.15	62.07	66.51	61.20	65.27	60.73
	10	61.35	55.14	59.81	53.22	57.73	51.90	56.00	50.94	54.63	50.25
	100	-	47.61	50.06	44.29	48.07	41.77	46.46	40.48	44.99	39.56
Elaine	1	72.32	65.70	70.54	63.77	68.64	61.83	67.40	60.81	65.62	60.57
	10	61.83	55.31	60.09	53.45	58.11	52.00	56.83	50.92	55.21	50.92
	100	-	47.68	50.26	44.46	48.36	41.99	46.77	40.20	45.22	39.18

	1	70.89	65.07	63.74	63.04	62.38	61.31	67.14	59.74	66.11	58.58
Pepper	10	61.27	55.19	59.85	53.26	57.85	51.82	56.27	50.66	54.88	49.79
	100	-	47.40	49.96	43.94	47.93	41.78	46.29	40.74	44.89	39.96

TABLE III. NUMBER OF GROUPS (MIN=MINIMUM, MAX=MAXIMUM, QTY=QUANTITY) WITH VARIOUS MESSAGE SIZES AND EMBEDDING LEVELS (EL)

Image	EL	Number of Groups											
		Message size = 1 Kb				Message size = 10 Kb				Message size = 100 Kb			
		Min	Qty	Max	Qty	Min	Qty	Max	Qty	Min	Qty	Max	Qty
Lena	1	2	268	5	27	2	2491	14	1	2	28168	16	2
	3	2	92	16	2	2	875	31	1	2	13998	37	1
	5	2	43	60	2	2	376	71	1	2	8209	78	1
Baboon	1	2	467	4	2	2	4276	8	1	2	18585	19	2
	3	2	448	5	1	2	3738	8	2	2	29210	53	1
	5	2	434	4	2	2	3312	9	3	2	24524	26	1
Boat	1	2	282	7	1	2	3171	8	1	2	24414	18	1
	3	2	138	21	1	2	1407	37	1	2	18641	57	1
	5	2	51	46	1	2	585	63	3	2	12834	63	1
Elaine	1	2	167	10	10	2	2595	38	1	2	26230	17	1
	3	2	151	25	3	2	1765	27	1	2	23570	33	1
	5	2	94	25	1	2	1259	14	1	2	16363	58	2
Pepper	1	2	252	15	1	2	3050	15	1	2	28401	15	2
	3	2	84	66	1	2	1529	66	1	2	16289	66	1
	5	2	47	86	1	2	892	86	1	2	9253	86	1

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 Baboon image with 100kb message size. 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.

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

[1] M. H. A. Al-Hooti, S. Djanali, and T. Ahmad, "Audio data hiding based on sample value modification using modulus function," *Journal of Information Processing System*, vol. 12, no. 3, pp. 525-537, 2016.

[2] R. M. Rad, K. S. Wong, and J. Guo, "Reversible data hiding by adaptive group modification on histogram of prediction errors," *Signal Processing*, vol. 125, pp. 315-328, August 2016.

[3] M. H. A. Al Huti, T. Ahmad, and S. Djanali, "Increasing the capacity of the secret data using DE pixels blocks and adjusted RDE-based on grayscale images," in *Proc. International Conference on Information and Communication Technology and Systems*, Surabaya, 2015, pp. 225-230.

[4] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 2016–2027, 2015.

[5] M. B. Andra, T. Ahmad, and T. Usagawa, "Medical record protection with improved GRDE data hiding method on audio files," *Engineering Letters*, vol. 25, no. 2, pp. 112-124, 2017.

[6] J. C. Chang, Y. Z. Lu, and H. L. Wu, "A separable reversible data hiding scheme for encrypted JPEG bitstreams," *Signal Processing*, vol. 144, pp. 135-143, 2017.

[7] C. V. Kumar and V. Natarajan, "Hybrid local prediction error-based difference expansion reversible watermarking for medical images," *Computers and Electrical Engineering*, vol. 53, pp. 333-345, 2016.

[8] J. Tian, "Reversible Data Embedding Using A Difference Expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 890-896, August 2003.

[9] X. Gui, X. Li, and B. Yang, "A novel integer transform for efficient reversible watermarking," in *Proc. 21st International Conference on Pattern Recognition*, Tsukuba, 2012, pp. 947-950.

[10] F. Peng, X. Li, and B. Yang, "Adaptive reversible data hiding scheme based on integer transform," *Signal Processing*, vol. 92, pp. 54-62, January 2012.

[11] N. Chen, C. Su, C. Shih, and Y. Chen, "Reversible watermarking for medical images using histogram shifting with location map reduction," in *Proc. IEEE International Conference on Industrial Technology*, Taipei, 2016, pp 792-797.

[12] M. Liu, H. S. Seah, C. Zhu, W. Lin, and F. Tian, "Reducing location map in prediction-based difference expansion for reversible image data embedding," *Signal Processing*, vol. 92, pp. 819-828, March 2012.

[13] B. Ou, X. Li, and J. Wang, "High-fidelity reversible data hiding based on pixel-value-ordering and pairwise prediction-error expansion," *Journal of Visual Communication and Image Representation*, vol. 39, pp. 12–23, August 2016.

[14] B. Ou, X. Li, Y. Zhao, R. Ni, and Y. Shi, "Pairwise Prediction-Error Expansion for Efficient Reversible Data Hiding," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5010–5021, 2013.

[15] D. Ioan-Catalin and D. Coltuc, "Local-Prediction-Based difference *expansion* reversible watermarking," *IEEE Transaction on Image Processing*, vol. 23, pp. 1779-1790, April 2014.

[16] X. Li, B. Yang, and T. Zeng, "Efficient Reversible *Watermarking* Based on Adaptive Prediction-Error Expansion and Pixel Selection," *IEEE Transaction on Image Processing*, vol. 20, pp. 3524-3533, December 2011.

[17] X. Li, J. Li, B. Li, and B. Yang, "High-Fidelity reversible data hiding *scheme* based on pixel-value-ordering and prediction-error-expansion," *Signal Processing*, vol. 93, pp. 198-205, January 2013.

[18] X. Wang, J. Ding, and Q. Pei, "A novel reversible image data hiding scheme based on pixel value ordering and dynamic pixel block partition," *Information Sciences*, vol. 310, pp. 16-35, July 2015.

[19] M. Li and Y. Li, "Histogram shifting in encrypted images with public key cryptosystem for reversible data hiding," *Signal Processing*, vol. 130, pp. 190-196, January 2017.

[20] H. Chen, J. Ni, W. Hong, and T. Chen, "Reversible data hiding with contrast enhancement using adaptive histogram shifting and pixel value ordering," *Signal Processing: Image Communication*, vol. 46, pp. 1-16, 2016.

[21] Z. Zhao, H. Luo, Z. M. Lu, and J. S. Pan, "Reversible Data Hiding Based on Multilevel Histogram Modification and Sequential Recovery," *AEU - International Journal of Electronics and Communications*, vol. 65, pp. 814-826, October 2011.

[22] W. He, G. Xiong, K. Zhou, and J. Cai, "Reversible data hiding based on multilevel histogram modification and pixel value grouping," *Journal of Visual Communication and Image Representation*, vol. 40B, pp. 459-469, October 2016.

[23] J. Wang, J. Ni, X. Zhang, and Y. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Transactions on Cybernetics*, vol. 47, no. 2, pp. 315–326, 2017.

[24] Z. Pan, S. Hu, X. Ma, and L. Wang, "Reversible data hiding based on local histogram shifting with multilayer embedding," *Journal of Visual Communication and Image Representation*, vol. 31, pp. 64-74, August 2015.

[25] W. He, G. Xiong, K. Zhou, and J. Cai, "Reversible data hiding based on multilevel histogram modification and pixel value grouping," *Journal of Visual Communication and Image Representation*, vol. 40, pp. 459-469, October 2016.

[26] USC-SIPI (1977). *The USC-SIPI Image Database* [Online]. Available: http://sipi.usc.edu/database

