International Journal of Innovative
Computing, Information and Control
Volume **9**, Number **2**, February **2013**

ICIC International ©2013 ISSN 1349-4198
pp. **693–704**

REVERSIBLE DATA HIDING BY COMBINING LOCAL AND GLOBAL SEARCH

Xiang Wang[1], Qingqi Pei[2,*], Xinbo Gao[3] and Hui Li[1]

[1]School of Telecommunications Engineering
[2]State Key Laboratory of Integrated Service Networks
[3]School of Electronic Engineering
Xidian University
No. 2, South Taibai Road, Xi'an 710071, P. R. China
wangxiang@xidian.edu.cn; { xbgao; lihui }@mail.xidian.edu.cn
*Corresponding author: qqpei@mail.xidian.edu.cn

Received November 2011; revised March 2012

Abstract. *Finding an appropriate predictor is the first and also the most important step for the histogram shifting based reversible data hiding. Unfortunately, most predictors reported in the literature cannot be directly introduced to the embedding process. To this end, this paper proposes a hybrid predictor for histogram shifting, which not only uses the local information near a pixel, but also utilizes the global information of the whole image. In addition, we design an estimation function which enables the use of sorting. As a result, the embedding performance is significantly improved. The superiority of the proposed data hiding method is finally experimentally verified by comparing with four state-of-the-art methods.*

Keywords: Reversible data hiding, Lossless data hiding, Reversible watermarking, Histogram shifting

1. **Introduction.** Reversible data hiding (also known as reversible watermarking) has the ability of exactly recovering the host image as well as extracting the hidden data. Because of the reversibility, it has been widely applied to some high fidelity applications, e.g., medical and military image protection. So far, several methods have been proposed in the literature.

Fridrich *et al.* [1] used the lossless compression to reversibly hide the data. The embedding space is created by losslessly compressing a special part of the host image. Then, the host is substituted by the compressed image and the hidden data. Celik *et al.* [2] further generalized the lossless compression based method by using LSB replacement. Limited by the compression ratio, the maximum embedding capacity of this kind of method is less than 1 bits per pixel (bpp).

The difference expansion (DE) technique proposed by Tian [3] is a more productive approach. The DE technique expands the difference of two adjacent pixels to carry one bit of data. Owing to the correlation between the neighboring pixels, this difference usually has small magnitude. Thus, the DE method introduces low distortion to the host image. Since two pixels are used to embed one bit of data, the embedding capacity of DE is bounded by 0.5 bpp. To further improve the embedding capacity, Alattar [4] and Wang *et al.* [5] generalized DE technique to blocks with arbitrary size rather than pixel pairs. As a result, the embedding capacity is improved from DE's 0.5 bpp to almost 1 bpp. Chen and Tsai [6] and Hong and Chen [7] also hid data into pixel blocks, but the block size is adaptively adjusted according to the texture information. Notice that, the overflow and underflow problems are inevitable in both Tian's original DE technique and

its extensions. The location map technique is widely adopted to solve the flow problems. However, in order to ensure reversibility, the location map should be embedded together with the data, which usually occupies a huge partition of the embedding space (even the location map is compressed) [8, 9, 10].

Inspired by Tian's DE technique, the histogram shifting technique is introduced to reduce the size of auxiliary information [11, 12, 13, 14, 15, 16]. In the classical histogram shifting model (proposed by Thodi and Rodriguez [11]), the encoder mainly includes two steps: the prediction and the histogram shifting. The result of prediction highly determines the performance of the histogram shifting and the whole data hiding process. For natural images, due to the high correlation, the prediction errors are mostly equal to (or nearly equal to) zero. The statistical histogram of the prediction errors has a sharp curve with the highest peak value at the origin. According to the histogram shifting strategy, the prediction errors around the origin are used to embed the data, while the others are shifted to make free space for embedding. Therefore, the sharper the histogram is, the less distortion the histogram shifting generates.

Sachnev *et al.* [17] introduced the rhombus pattern predictor into histogram shifting. The rhombus pattern predictor is experimentally proved to be more efficient than median edge detector [11]. So, it is more suitable for histogram shifting. In addition, by exploiting Kamstra's sorting idea [18], small prediction errors in the embedding process are preferentially selected to hide the data. Experimental results show that this method achieves significant improvement over Thodi's method.

It seems that one can simply employ a more efficient prediction algorithm to improve the performance of histogram shifting. Although there are numerous superior predictors, most of them cannot be directly introduced to the reversible data hiding process. Two reasons account for this failure. Firstly, the prediction value of each pixel should be identical between the encoder and the decoder to guarantee the reversibility, which is not necessary for the prediction algorithm. Secondly, the steps of the reversible watermark embedding are not independent of each other. The predictor needs to be well integrated with other steps (the histogram shifting, sorting, etc.). In fact, finding an appropriate predictor is an important task for reversible data hiding methods.

In this paper, we design a hybrid predictor for the histogram shifting based reversible data hiding. The proposed predictor robustly exploits the correlation within the whole image, and produces better results than the rhombus pattern predictor and the median edge detector. In addition, a function used to estimate the efficiency of the hybrid predictor is designed, which ensures that the global predictor satisfies the requirement of sorting. We further experimentally indicate that the proposed hybrid predictor produces a sharper histogram than several state-of-the-art works. Therefore, according to the principle of histogram shifting, our predictor is more applicable to the reversible data hiding. A detailed description of the incorporation of the sorting and the proposed predictor is also given.

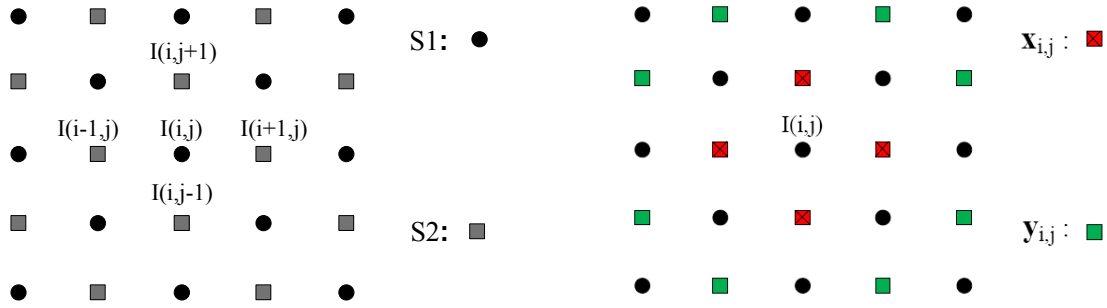
2. **Hybrid Predictor.**

In the hybrid predictor, the local prediction is firstly used to predict the pixels in smooth area, and the global prediction is then employed to refine the prediction results in textural area.

2.1. **Local prediction.**

The rhombus pattern predictor is employed to compute the local prediction value owning to the remarkable prediction performance it produces. Considering a 512×512 sized image I, the pixels of the host image are divided into two sets:

$$I_1 = \{I(x, y) : x \in [1, 512], y \in \{2k - x\%2\}\},$$
$$I_2 = \{I(x, y) : x \in [1, 512], y \in \{2k + x\%2 - 1\}\},$$



(a) Labeling of I_1 and I_2 (b) Labeling of $\mathbf{x}_{i,j}$ and $\mathbf{y}_{i,j}$

FIGURE 1. Labeling of local and global prediction

where $k \in [1, 256]$ and % represents the modulo operator. The black points and the grey boxes in Figure 1(a) denote the pixels in set I_1 and I_2, respectively. The local prediction value of $I(i, j)$ in set I_1 is computed according to its four neighboring pixels. For simplicity, the vector $\mathbf{x}_{i,j}$ is defined to consist of these four pixels (red blocks with cross in Figure 1(b)):

$$\mathbf{x}_{ij} = \{I(i, j-1), I(i, j+1), I(i-1, j), I(i+1, j)\}. \tag{1}$$

Then, $I(i, j)$ is predicted as

$$\dot{I}(i, j) = \lfloor E(\mathbf{x}_{ij}) \rfloor, \tag{2}$$

where $\lfloor \cdot \rfloor$ represents the floor operation. Notice that the reference pixels used to predict $I(i, j)$ belong to set I_2, i.e., belong to a different set from the predicted one.

The texture complexity strongly determines the efficiency of local prediction. In smooth area, Equation (2) achieves an accurate prediction value, whereas, it usually does not work efficiently in textural area. Therefore, the local texture information can be exploited to indicate the efficiency of the local prediction. Regarding a pixel $I(i, j)$, the local texture information is computed as

$$v_{loc}(i, j) = \frac{\sum_{k=1}^{4} |\mu(i, j, k) - \bar{\mu}(i, j)|}{4}, \tag{3}$$

where $\mu(i, j, 1) = |I(i, j-1) - I(i-1, j)|$, $\mu(i, j, 2) = |I(i-1, j) - I(i, j+1)|$, $\mu(i, j, 3) = |I(i, j+1) - I(i+1, j)|$, $\mu(i, j, 4) = |I(i+1, j) - I(i, j-1)|$ and $\bar{\mu}(i, j) = \sum_{k=1}^{4} \mu(i, j, k)/4$. A small $v_{loc}(i, j)$ indicates that the pixel $I(i, j)$ is in smooth area and the prediction value is quite close to its original value, and vice versa.

2.2. **Global prediction.** It is observed that the natural image includes many textural areas which usually repeat themselves at various locations (as shown in Figure 2). Based on this observation, the global prediction is designed to calculate the prediction value in these areas. For simplicity, the vector $\mathbf{y}_{i,j}$ is designed to include the pixels surrounding $I(i, j)$ (green blocks in Figure 1(b)):

$$\begin{aligned} \mathbf{y}_{i,j} = \{&I(i+1, j-2), I(i-1, j-2), I(i+1, j+2), I(i-1, j+2), \\ &I(i-2, j-1), I(i-2, j+1), I(i+2, j-1), I(i+2, j+1)\}. \end{aligned} \tag{4}$$

Clearly, if two pixels have similar contexts (neighborhoods), their values are close to each other. So, the sets $\mathbf{x}$ (Equation (1)) and $\mathbf{y}$ (Equation (4)) are exploited to estimate

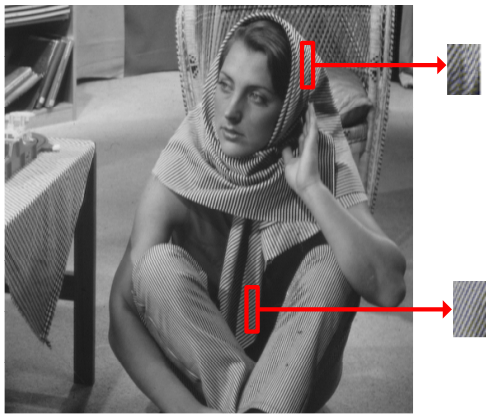


FIGURE 2. Nature image often has many similar textures

the similarity between two pixels, which is defined as

$$Sim(I(i_1,j_1),I(i_2,j_2)) = \frac{1}{8}\|\mathbf{x}_{i_1j_1} - \mathbf{x}_{i_2j_2}\|_{l^2} + \frac{1}{16}\|\mathbf{y}_{i_1j_1} - \mathbf{y}_{i_2j_2}\|_{l^2}. \tag{5}$$

Then all the pixels in the set I_1 are sorted in ascending order in terms of the local variance v_{local}. Suppose the set I_1 after sorting is

$$\hat{I}_1 = \{I(i_1,j_1), I(i_2,j_2), I(i_3,j_3), \ldots, I(i_n,j_n)\}.$$

If $k < l$, we have $v_{loc}(i_k,j_k) \leq v_{loc}(i_l,j_l)$. Then, for a pixel $I(i_k,j_k)$ in set $\hat{I}_1$, its global prediction value is computed as

$$\ddot{I}(i_k,j_k) = \left\lfloor \frac{\sum_{l=1}^{k-1} \omega(I(i_l,j_l),I(i_k,j_k))I(i_l,j_l)}{\sum_{l=1}^{k-1} \omega(I(i_l,j_l),I(i_k,j_k))} \right\rfloor, \tag{6}$$

where $\omega(I(i_l,j_l),I(i_k,j_k)) = e^{-Sim(I(i_l,j_l),I(i_k,j_k))/h}$, and h is the parameter to control the decay of the exponential function. The above equation indicates that only pixels prior to $I(i_k,j_k)$ in the set $\hat{I}_1$ are used to calculate the global prediction value, which ensures that the global prediction value in the encoder remains identical with the one in the decoder.

Equation (6) also indicates that the efficiency of global prediction depends on the context similarity between the predicted pixel and the reference pixels. So, the following equation is defined to estimate the global prediction efficiency:

$$v_{glob}(I(i_k,j_k)) = \frac{\sum_{l=1}^{k-1} \omega(I(i_l,j_l),I(i_k,j_k))Sim(I(i_l,j_l),I(i_k,j_k))}{\sum_{l=1}^{k-1} \omega(I(i_l,j_l),I(i_k,j_k))}.$$

The above equation first calculates the similarities between the current pixel and its previous pixels in set $\hat{I}_1$. Then, the similarities are normalized to indicate the efficiency of the global prediction.

2.3. **Hybrid predictor.** This subsection introduces how to use both the local and global prediction to calculate a prediction value, which is named as hybrid predictor. Specifically, the hybrid prediction value of the pixel $I(i,j)$ is calculated as

- If $v_{loc}(i,j) < v_{glob}(i,j)$, set $\hat{I}(i,j) = \dot{I}(i,j)$ and $v_{hyb}(i,j) = v_{loc}(i,j)$.

• If $v_{loc}(i,j) \geq v_{glob}(i,j)$, the hybrid prediction value and $v_{hyb}(i,j)$ is computed as

$$\widehat{I}(i,j) = \frac{e^{-v_{loc}(i,j)}\dot{I}(i,j) + e^{-v_{glob}(i,j)}\ddot{I}(i,j)}{e^{-v_{loc}(i,j)} + e^{-v_{glob}(i,j)}},$$

and

$$v_{hyb}(i,j) = \frac{e^{-v_{loc}(i,j)}v_{loc}(i,j) + e^{-v_{glob}(i,j)}v_{glob}(i,j)}{e^{-v_{loc}(i,j)} + e^{-v_{glob}(i,j)}}. \tag{7}$$

The global prediction is used only when $v_{loc}(i,j) \geq v_{glob}(i,j)$, for this means that the pixel is probably in the textural area. In this case, the global prediction is used to refine the local prediction results. Furthermore, Equation (7) indicates that the better the global prediction performs than the local prediction, the larger proportion it occupies in hybrid prediction, and vice versa. The weights ($e^{-v_{loc}(i,j)}$ and $e^{-v_{glob}(i,j)}$) are calculated based on $v_{loc}(i,j)$ and $v_{glob}(i,j)$, which are the estimations of the local and global prediction efficiency, respectively. Accordingly, an estimation value v_{hyb} used for sorting (introduced later) is computed to indicate the efficiency of the hybrid predictor.

3. **Encoder and Decoder.**

3.1. **Histogram shifting.** The classical histogram shifting technique in [11] is utilized in this work to hide the data efficiently. For a pixel $I(i,j)$, its prediction error is computed as

$$d(i,j) = I(i,j) - \widehat{I}(i,j).$$

Then, the prediction error $d(i,j)$ of the pixel $I(i,j)$ is shifted or expanded according to two thresholds T_l and T_r as

$$d^w(i,j) = \begin{cases} d(i,j) + T_l, & \text{if } d(i,j) < T_l, \\ 2d(i,j) + w, & \text{if } d(i,j) \in [T_l, T_r], \\ d(i,j) + T_r + 1, & \text{if } d(i,j) > T_r, \end{cases} \tag{8}$$

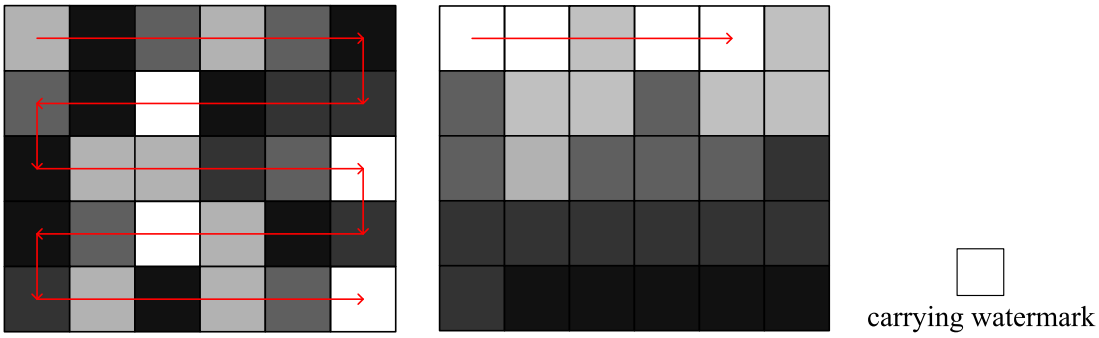
where $w \in \{0,1\}$ is a bit to be hidden. The embedded pixel value is reconstructed as $I^w(i,j) = \widehat{I}(i,j) + d^w(i,j)$. In the decoder, the pixels are recovered as

$$d(i,j) = \begin{cases} d^w(i,j) - T_l, & \text{if } d^w(i,j) < 2T_l, \\ \left\lfloor \frac{d^w(i,j)}{2} \right\rfloor, & \text{if } d^w(\mathbf{x}_k) \in [2T_l, 2T_r + 1], \\ d^w(i,j) - T_r - 1, & \text{if } d^w(i,j) > 2T_r. \end{cases} \tag{9}$$

Evidently, the data can be extracted by reading the LSBs of $d^w(i,j)$.

The overflow/underflow problem is inevitable in the histogram shifting. Since some pixels may cause overflow (more than 255) and underflow (less than 0) problem if changed by Equation (8), they should not be involved in the embedding process. To deal with this problem, a location map is established to distinguish these pixels from the others, and then embedded into the host image together with the watermark.

In this method, Thodi's *DS-HS-FB* technique is used to reduce the size of location map. Specifically, the pixels that can be modified by Equation (8) more than twice (without causing underflow/overflow problem) are not recorded in the location map, since we can distinguish these pixels from the others by testing whether they are still modifiable in the decoder. In addition, the experimental results in [11] illustrate that this kind of pixels usually comprises a considerable proportion in the image. Therefore, in comparison with the traditional location map technique which records all the pixels of the host image, the size of location map decreases. Due to space limitation, this technique is not presented here. The details can be found in [11].



(a) Before sorting (b) After sorting

FIGURE 3. An example to illustrate the effort of sorting

3.2. **Sorting.** The introduction of sorting significantly improves the embedding performance of histogram shifting. However, the sorting requires that embedding data into a pixel should not affect the embedding of other pixels, which leads to the failure of incorporating sorting into many prediction algorithms (e.g., MED and GAP). The hybrid prediction, like the rhombus pattern prediction, divides the pixels of the host image into two parts: I_1 and I_2. When hiding data into I_1, the pixels in I_2 are used to compute the prediction value. Therefore, changing a pixel in I_1 will not affect the embedding process of other pixels. Based on this property, the sorting idea is introduced to the histogram shifting process.

The value v_{hyb} (Equation (7)) is computed to estimate the efficiency of the hybrid predictor. The pixel with small v_{hyb} usually has small prediction error, and therefore should be given high priority during the embedding process. In order to achieve that, the pixels are first sorted in terms of v_{hyb}. Then, according to the data capacity, a part of pixels are selected (in ascending order) to embed the data, and the others are left unchanged. Therefore, after sorting, the pixels introducing small distortion (i.e., with small prediction error) are first used for embedding. Whereas the classical histogram shifting processes the pixels one by one according to their positions in the image, which definitely generates more embedding distortion.

Figure 3 gives an example to demonstrate the effect of sorting. For the sake of clarity, 30 boxes are used to represent the prediction errors of 30 pixels, and the gray level of the box represents the value of the prediction error: the darker the box, the larger the prediction error. Since the distribution of the pixels of nature image (especially the textural image) is not regular, the prediction errors are usually in random distribution before sorting (Figure 3(a)). Suppose 4 bits data are hid into these prediction errors. According to the threshold selection strategy, the data should be embedded into four smallest prediction errors (Figure 3(c)). However, in order to achieve the embedding capacity, the embedding process has to traverse all the 30 pixels. As a result, the encoder shifts the other 26 prediction errors based on Equation (8), which generates unnecessary distortion.

Figure 3(b) indicates the benefits of sorting. After sorting, the prediction errors are almost in ascending order. Notice that since Equation (7) is only an estimation function, we cannot make sure that the prediction errors are perfectly in ascending order. Also considering embedding 4 bits data, it is observed that the first 5 pixels are able to provide enough payloads. In this case, only one pixel (the third one) needs to be shifted. Therefore, the distortion caused by shifting is significantly reduced.

3.3. **Encoder.** A two-pass embedding scheme is designed to embed the data into the host image. In the first pass, half of the data is embedded into the pixels in the set I_1.

In the second pass, the rest data is embedded into I_2. For simplicity, only the embedding process for the preselected parameters (T_l and T_r) is given, but the embedding process can be easily extended to embed a desired amount of data that meets the requirement.

Step-1: Sort all the pixels in the set I_1 in ascending order based on v_{hyb}. Define the sorted set as ${I_1}'$.

Step-2: Establish a location map M, and assume its length is m. Divide the pixels in set ${I_1}'$ into three parts: I_{1_1}', I_{1_2}' and I_{1_3}'. I_{1_3}' includes the last m pixels in set ${I_1}'$, and I_{1_1}' consists of the first m pixels with $d \in [T_l, T_r]$, and the rest pixels constitute I_{1_2}'.

Step-3: Embed the location map M into I_{1_3}' by least significant bit (LSB) replacement, and record the original LSB sequence as L.

Step-4: Embed the sequence L into I_{1_1}' by Equation (8). Then, embed the data into I_{1_2}' according to Equation (8).

Step-5: Repeat *Steps 1-4*, and embed the rest data into set I_2.

Finally, the image carrying the whole data is produced, and the embedding process is finished.

3.4. **Decoder.** The encoder first embeds the data into set I_1, and then into I_2. The decoder, on the contrary, has to first recovery the pixels of I_2 (also extract the data). Then, the data carried by I_1 can be correctly extracted. Only in this way can the consistency between the encoder and the decoder be guaranteed. The decoding process is summarized as follows.

Step-1: Sort all the pixels in the set I_2 in ascending order based on v_{hyb}. Suppose the sorted set is I_2'.

Step-2: Extract the location map M by reading the LSBs of the last m pixels. Based on the location map M, determine the pixels which are modified by encoder (named as *embeddable*).

Step-3: Among all the *embeddable* pixels, we can identify which pixels are carrying watermark by the condition $d^w \in [2T_l, 2T_r + 1]$. Then the sequence L (see *Step-3* of encoder) and the watermark are extracted by reading the LSBs of these pixels.

Step-4: Recover the *embeddable* pixels according to Equation (9), and the last m pixels of ${I_2}'$ by replacing their LSBs with the corresponding values in sequence L.

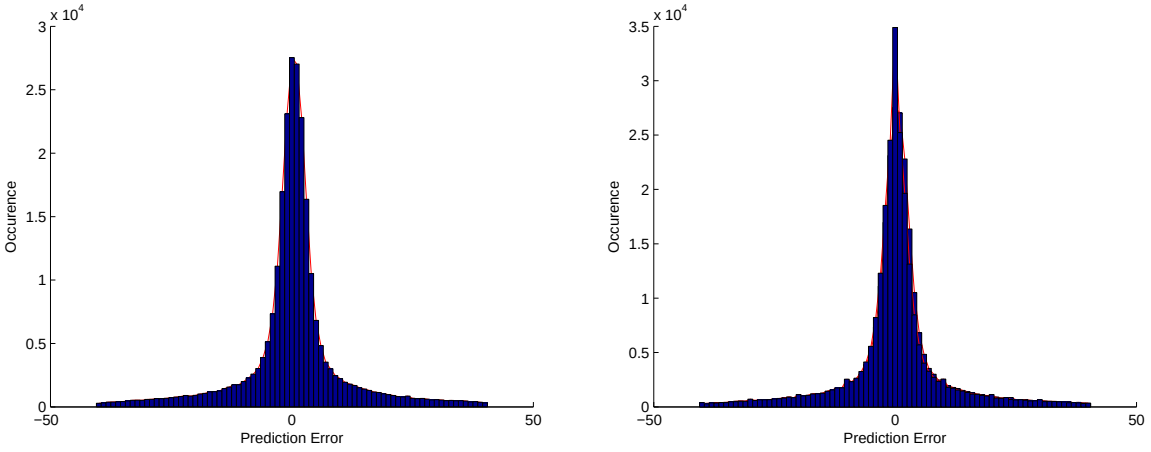
Step-5: Repeat the same extracting and recovering process to set I_1.

Finally, the data is extracted and the original image is restored.

4. **Experimental Results.** The hybrid predictor is first evaluated by comparing with the rhombus pattern predictor. The two predictors are applied on the tested image "Barbara". The prediction errors of all the pixels are collected to construct the statistic histogram. Figure 4 shows that the proposed predictor produces a sharper histogram than the rhombus pattern predictor does. Specifically, the peak values of the rhombus pattern predictor and the proposed one are close to 27000 and 35000, respectively. As aforementioned, the performance of histogram shifting depends on the shape of the histogram. For the same embedding capacity, a sharper histogram shifts fewer pixels and generates less embedding distortion. Therefore, the hybrid predictor is more suitable for the histogram shifting.

The proposed method is compared with four state-of-the-art algorithms (capacity versus distortion) to well demonstrate the performance:

(1) Alattar *et al.*'s method [4];
(2) Hu *et al.*'s method [19];
(3) Tai *et al.*'s method [20];



(a) Rhombus pattern predictor (b) Hybrid predictor

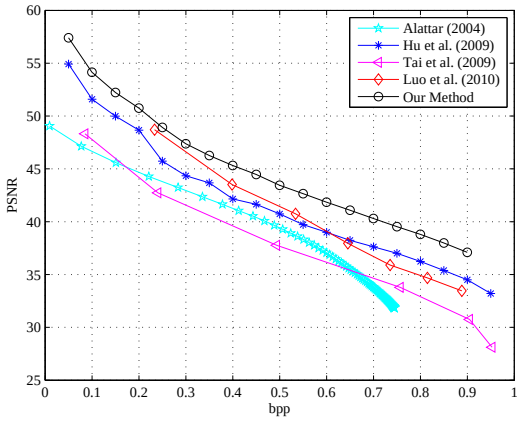
FIGURE 4. Histograms of rhombus pattern predictor and hybrid predictor. The tested image is Barbara.

(4) Luo *et al.*'s method [21].

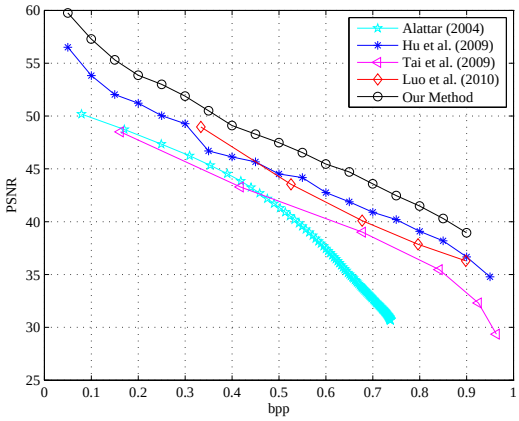
Alattar *et al.*'s method represents the classical integer transform based reversible data hiding methods. The other three are the recently proposed histogram shifting methods. Six standard 512×512 sized gray-scale images downloaded from USC-SIPI are used in the experiments: "Lena", "Baboon", "Barbara", "Airplane" (F-16), "Peppers", "Goldhill". We implement the methods 1-4 by using Matlab, and the arithmetic lossless compression coding is employed to compress the location map. For our method, we vary the embedding capacity from 0.05 bpp to 0.9 bpp with step size 0.05.

The comparison results in Figure 5 indicate that the proposed method performs better than the methods 1-4 at almost all embedding rates. The best performance of the proposed method is obtained on "Barbara". Since "Barbara" has more repeated textures than the other images, the global prediction performs more efficiently, and the advantage is more obvious. Compared with [4]'s method, the maximum improvement appearing at 0.7 bpp is more than 10 dB. Compared with the histogram shifting methods 2-4, the maximum improvement of 6 dB, 10 dB and 5 dB is obtained at 0.9 bpp, 7.5 bpp and 0.9 bpp, respectively. However, Figure 5 also presents that the superiority is not clear on "Baboon". The performances of [21]'s method and the proposed one are very close at small embedding capacity, e.g., the improvement at 0.1 bpp is 0.2 dB. Since textures of "Baboon" are mostly irregular, the hybrid predictor cannot provide significant improvement. In addition, it is observed that the gain of the proposed method is more clear at large embedding capacity (especially larger than 0.5 bpp). This is mainly because that the pixels in smooth areas are able to provide enough payloads when the embedding capacity is small. Therefore, pixels in textural areas are excluded from the embedding capacity. According to Subsection 2.3, only the local prediction is used in smooth area. Thus, the improvement is diminished. When the embedding capacity increases, the pixels in textural areas are involved. The global prediction is employed and yields more accurate prediction results for these pixels than the predictors used in 2-4. Therefore, the most significant improvements appear at large embedding capacity.

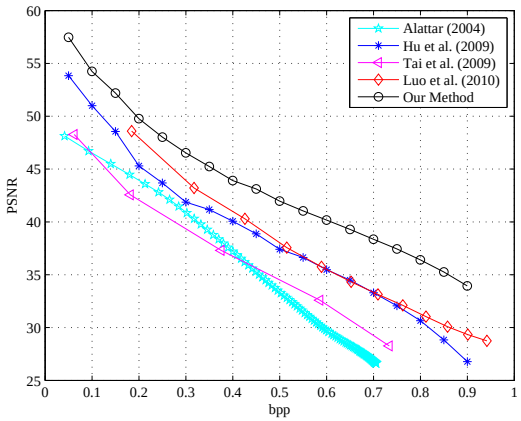
We also evaluate the performance of our method at small embedding capacity. We vary the embedding capacity from 0.005 bpp to 0.1 bpp with step size 0.005. Among the above methods 1-4, Hu *et al.*'s method reportedly has the best performance when



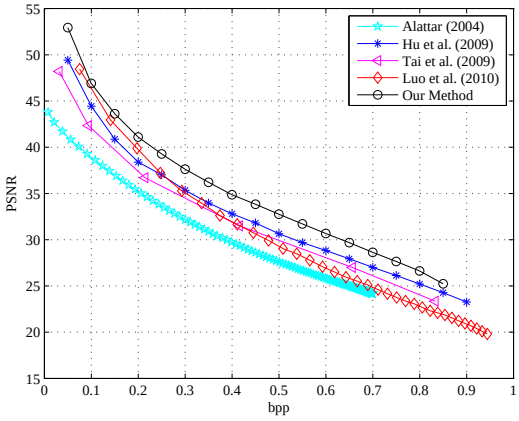
(a) Lena



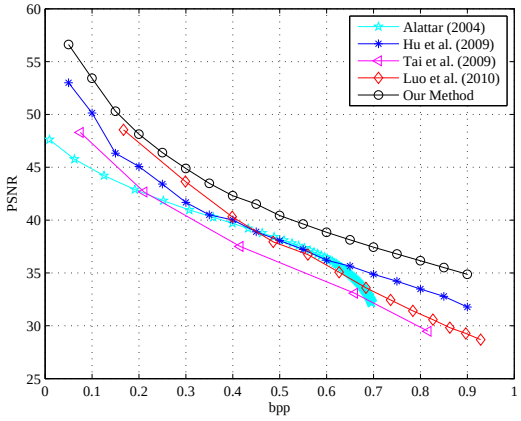
(b) Airplane



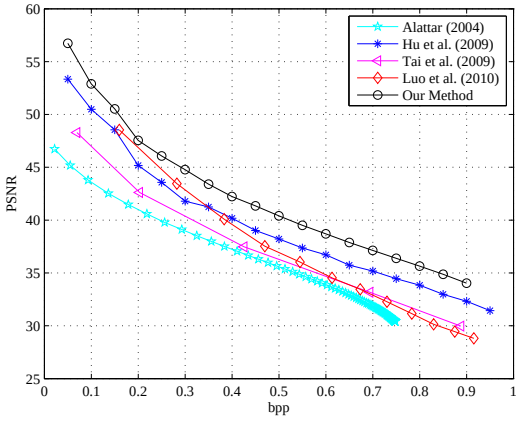
(c) Barbara



(d) Baboon



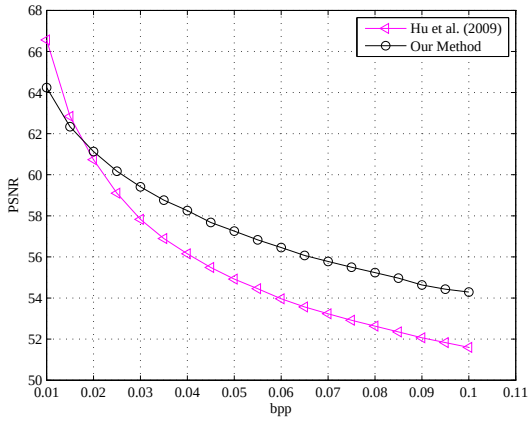
(e) Peppers



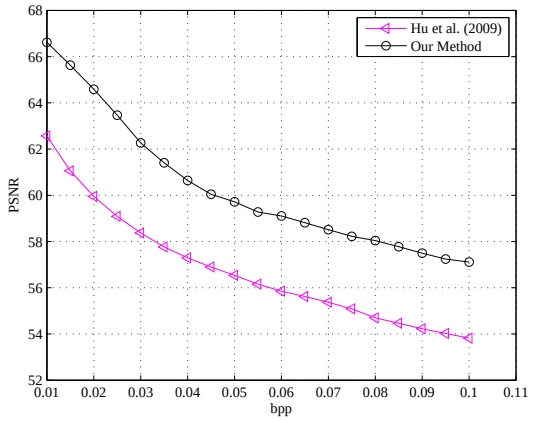
(f) Goldhill

FIGURE 5. Performance comparison of other four methods with the proposed method

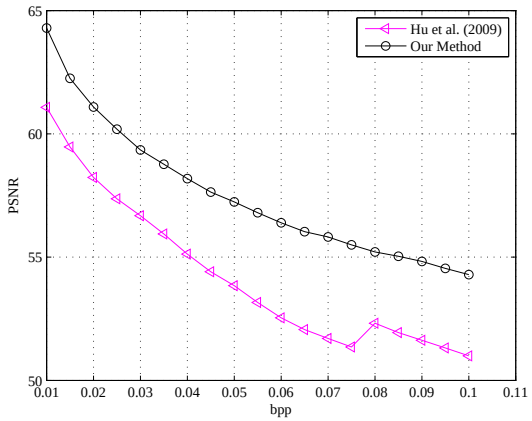
the payload is small. So, we only present the comparison between [19]'s method and the proposed method. Figure 6 indicates that the advantage of our method is less obvious when embedding capacity is small. The maximum improvement of PSNR is 4.1 dB



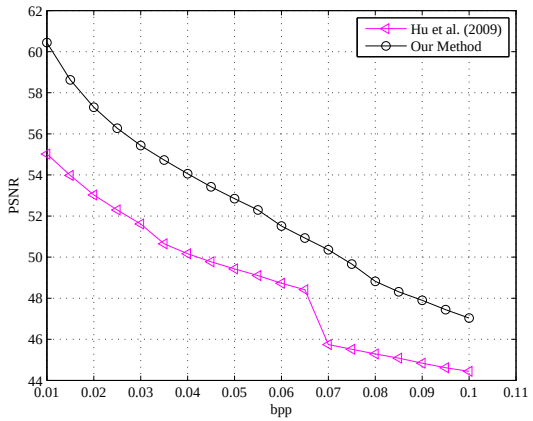
(a) Lena



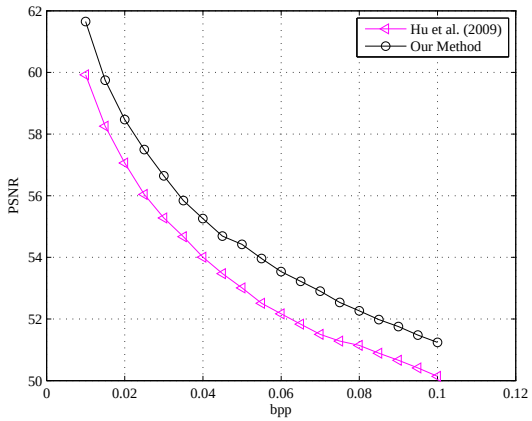
(b) Airplane



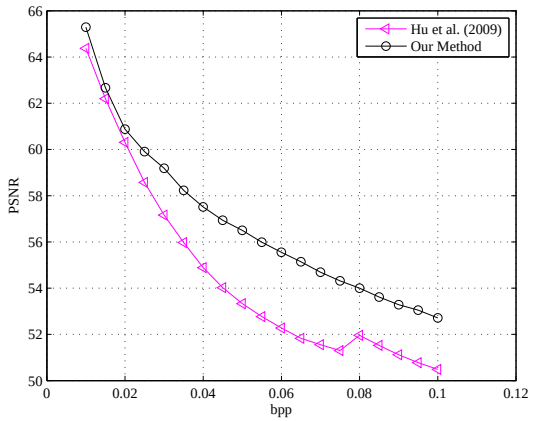
(c) Barbara



(d) Baboon



(e) Peppers



(f) Goldhill

FIGURE 6. Performance comparison between [19]'s method and the proposed method at small embedding capacity

(on "Barbara" at 0.1 bpp). For most cases, the improvement is less than 2 dB. When embedding capacity is less than 0.02 bpp, Hu *et al.*'s method outperforms the proposed

one on "Lena". It proves that the separate use of location prediction (without global prediction) cannot achieve significant gain over the other histogram shifting methods.

Note that, the reversible data hiding algorithms are designed for the protection of high-fidelity images. However, the high-fidelity applications usually have strict limitations on the embedding distortion of the host image, e.g., "almost lossless" and "nearly lossless" [22]. Therefore, introducing a little more distortion (even only one bit) may make a reversible data hiding method completely useless in a practical high-fidelity application. The experimental results clearly illustrate that the embedding performance of the proposed method is superior to those of some state-of-the-art reversible data hiding algorithms in terms of PSNR. Specifically, the improvement of PSNR is mostly larger than 2 dB, which indicates that the proposed algorithm is more widely applicable in high-fidelity image protection.

5. **Conclusion.** This paper proposed a global search prediction algorithm to improve the histogram shifting technique. Integrated with the traditional local prediction, the proposed predictor achieves a sharper histogram. Therefore, it is more suitable for histogram shifting. Furthermore, the sorting is introduced to the embedding process by designing an estimation equation, which significantly improves the performance of the histogram shifting. Experimental results illustrate that the proposed watermarking method outperforms many recently proposed methods.

Acknowledgment. This work is supported by the National Natural Science Foundation of China under Grant No. 61202391, No. 61172068 and No. U0835004, the Fundamental Research Funds for the Central Universities (Grant No. K50511010003 and No. K5051201030), Program for Changjiang Scholars and Innovative Research Team in University (Grant No. IRT1078), the 111 Project (No. B08038) and Program for New Century Excellent Talents in University (Grant No. NCET-11-0691).

REFERENCES

[1] J. Fridrich, M. Goljan and R. Du, Invertible authentication, *Security and Watermarking of Multimedia Contents III*, vol.4314, pp.197-208, 2001.

[2] M. U. Celik, G. Sharma, A. M. Tekalp and E. Saber, Lossless generalized-lsb data embedding, *IEEE Transactions on Image Processing*, vol.14, no.2, pp.253-266, 2005.

[3] J. Tian, Reversible data embedding using a difference expansion, *IEEE Transactions on Circuits and Systems for Video Technology*, vol.13, no.8, pp.890-896, 2003.

[4] A. M. Alattar, Reversible watermark using the difference expansion of a generalized integer transform, *IEEE Transactions on Image Processing*, vol.13, no.8, pp.1147-1156, 2004.

[5] X. Wang, X. Li, B. Yang and Z. Guo, Efficient generalized integer transform for reversible watermarking, *IEEE Signal Processing Letters*, vol.17, no.6, pp.567-570, 2010.

[6] C.-C. Chen and Y.-H. Tsai, Adaptive reversible image watermarking scheme, *Journal of Systems and Software*, vol.84, no.3, pp.428-434, 2011.

[7] W. Hong and T.-S. Chen, A local variance-controlled reversible data hiding method using prediction and histogram-shifting, *Journal of Systems and Software*, vol.83, no.12, pp.2653-2663, 2010.

[8] I. Usman, A. Khan, A. Ali and T.-S. Choi, Reversible watermarking based on intelligent coefficient selection and integer wavelet transform, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(A), pp.4675-4682, 2009.

[9] T.-C. Lu and C.-H. Tsai, An improved lossless hiding technique using integer transform function, *International Journal of Innovative Computing, Information and Control*, vol.5, no.9, pp.2645-2656, 2009.

[10] C.-C. Chen and D.-S. Kao, Dct-based zero replacement reversible image watermarking approach, *International Journal of Innovative Computing, Information and Control*, vol.4, no.11, pp.3027-3036, 2008.

[11] D. M. Thodi and J. J. Rodriguez, Expansion embedding techniques for reversible watermarking, *IEEE Transactions on Image Processing*, vol.16, no.3, pp.721-730, 2007.

[12] X. Gao, L. An, X. Li and D. Tao, Reversibility improved lossless data hiding, *Signal Processing*, vol.89, no.10, pp.2053-2065, 2009.
[13] W. Hong, T.-S. Chen and C.-W. Shiu, Reversible data hiding for high quality images using modification of prediction errors, *Journal of Systems and Software*, vol.82, no.11, pp.1833-1842, 2009.
[14] X. Gao, L. An, Y. Yuan, D. Tao and X. Li, Lossless data embedding using generalized statistical quantity histogram, *IEEE Transactions on Circuits and Systems for Video Technology*, no.99, 2011.
[15] S. Weng, Y. Zhao and J.-S. Pan, A novel reversible data hiding scheme, *International Journal of Innovative Computing, Information and Control*, vol.4, no.2, pp.351-358, 2008.
[16] X. Wang and Z. Guo, A novel reversible watermarking method using pre-selection strategy, *International Journal of Innovative Computing, Information and Control*, vol.7, no.11, pp.6187-6201, 2011.
[17] V. Sachnev, H. J. Kim, J. Nam, S. Suresh and Y. Q. Shi, Reversible watermarking algorithm using sorting and prediction, *IEEE Transactions on Circuits and Systems for Video Technology*, vol.19, no.7, pp.989-999, 2009.
[18] L. Kamstra and H. J. A. M. Heijmans, Reversible data embedding into images using wavelet techniques and sorting, *IEEE Transactions on Image Processing*, vol.14, no.12, pp.2082-2090, 2005.
[19] Y. Hu, H. Lee and J. Li, De-based reversible data hiding with improved overflow location map, *IEEE Transactions on Circuits and Systems for Video Technology*, vol.19, no.2, pp.250-260, 2009.
[20] W. L. Tai, C. M. Yeh and C. C. Chang, Reversible data hiding based on histogram modification of pixel differences, *IEEE Transactions on Circuits and Systems for Video Technology*, vol.19, no.6, pp.906-910, 2009.
[21] L. Luo, Z. Chen, M. Chen, X. Zeng and Z. Xiong, Reversible image watermarking using interpolation technique, *IEEE Transactions on Information Forensics and Security*, vol.5, no.1, pp.187-193, 2010.
[22] S. He, D. Kirovski and M. Wu, High-fidelity data embedding for image annotation, *IEEE Transactions on Image Processing*, vol.18, no.2, pp.429-435, 2009.