Optimizing Non-Local Pixel Predictors for Reversible Data Hiding

Xiaocheng Hu, School of Information Science and Technology, University of Science and Technology of China, Hefei, China

Weiming Zhang, School of Information Science and Technology, University of Science and Technology of China, Hefei, China

Nenghai Yu, School of Information Science and Technology, University of Science and Technology of China, Hefei, China

ABSTRACT

This paper presents a two-step clustering and optimizing pixel prediction method for reversible data hiding, which exploits self-similarities and group structural information of non-local image patches. Pixel predictors play an important role for current prediction-error expansion (PEE) based reversible data hiding schemes. Instead of using a fixed or a content- adaptive predictor for each pixel independently, the authors first employ pixel clustering according to the structural similarities of image patches, and then for all the pixels assigned to each cluster, an optimized pixel predictor is estimated from the group context. Experimental results demonstrate that the proposed method outperforms state-of-art counterparts such as the simple rhombus neighborhood, the median edge detector, and the gradient-adjusted predictor et al.

Keywords: *Clustering, L1-Norm Optimization, Pixel Prediction, Reversible Data Hiding, Self-Similarities*

INTRODUCTION

As a technique that embeds messages into cover signals, information hiding has been widely applied in areas such as convert communication, copyright protection and media annotation. Reversible data hiding (RDH), as a special type of information hiding technique, has received much attention from the

DOI: 10.4018/ijdcf.2014070101

information community (Shi, Ni, Zou, Liang, & Xuan, 2004; Shi, 2004; Caldelli, Filippini, & Becarelli, 2010) in the last decade. Specifically, RDH ensures not only the embedded messages shall be extracted precisely, but also the cover itself should be restored losslessly. This property is important in some special scenarios such as medical imagery (Bao et al., 2005), military imagery and law forensics. In these applications, the cover is too precious or too important to be damaged (Feng et al., 2006). Moreover, it has been found recently that reversible data hiding can be quite helpful in video error-concealment coding (Chung et al., 2010).

A plenty of reversible data hiding algorithms have been proposed in the past decade. Classical RDH methods roughly fall into three categories. The first class of algorithms follows the idea of compression-embedding framework, which was first introduced by Fridrich, Goljan, and Du (2002). In these algorithms, a two-value feature is calculated for each pixel group, the sequence is compressible and messages can be embedded in the extra space left by lossless compression. The send class of techniques is based on difference expansion (DE) (Tian, 2003; Thodi & Rodriguez, 2007), in which the differences of each pixel groups are expanded, e.g., multiplied by 2, and thus the least significant bits (LSBs) of the differences are all-zeros and can be used for embedding messages. The last RDH schemes are based on histogram shift (HS) (Ni, Shi, Ansari, & Wei, 2006). The histogram of one special feature (for example, gray-scale value) of the nature image is quite uneven, which implies that the histogram can be modified for embedding data. For instance some space can be saved for watermarks by shifting the bins of histogram.

In fact, by applying DE of HS to the residual part of nature images instead, e.g., the prediction errors (PE) (Tsai, Hu, & Yeh, 2009; Luo et al., 2010; Peng, Li, & Yang, 2012; Li, Yang, & Zeng, 2011), better performance can be achieved. This extended method is called prediction-error expansion (PEE), which is currently a research hotspot and the most powerful technique of RDH. Unlike in DE where only the correlation of two adjacent pixels is considered, the local correlation of larger neighborhood is exploited in PEE. Most recently proposed RDH works are based on PEE by incorporating some strategies such as better prediction algorithm utilization (Sachnev, Kim, Nam, Suresh, & Shi, 2009; Fallahpour, 2008; Yang, Chung, Liao, & Yu, 2013; Ou, Li, Zhao, & Ni, 2013), double-layered embedding (Luo et al., 2010; Sachnev et al., 2009), embedding position selection (Li et al., 2011), context modification (Coltuc, 2011), optimal bins selection (Wang, Li, & Yang, 2010; Wu & Huang, 2012), etc.

Almost all recent PEE based methods consist of two steps. The first step generates a host sequence with small entropy, i.e., the host has a sharp histogram, which usually can be realized by using PE combined with better prediction strategies. The second step reversibly embeds messages into the host sequence by modifying its histogram with method like HS and DE. The performance of the overall embedding scheme is directly influenced by the accuracy of prediction, the steeper the prediction errors histogram is, the better the embedding performance can be achieved. Typical prediction methods either use a fixed neighborhood average model (Sachnev et al., 2009), or a content adaptive predictor such as the median edge detector (MED) predictor (Thodi & Rodriguez, 2007) and the gradient-adjusted predictor (GAP) (Fallahpour, 2008). They treat each pixel independently while structural self-similarities of non-local image patches are rarely considered.

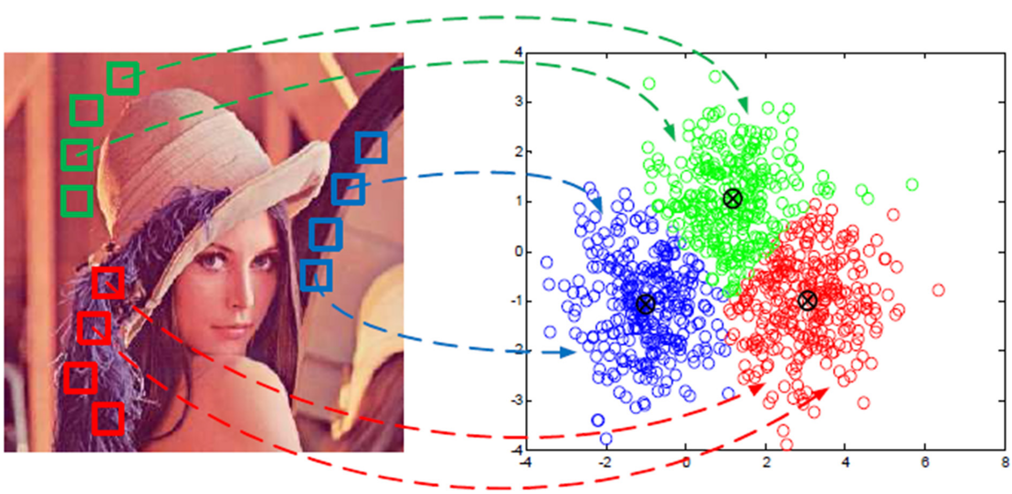
In this paper, we proposed to first divide all pixels into several clusters according to the patch-level structural similarities of their prediction context by means of clustering, as shown in Figure 1. Afterwards, for all the pixels assigned to each cluster, an optimized pixel predictor is estimated from the group context. Each pixel is predicted by a weighted linear combination of its nearest eight neighbors, and a quad- layered embedding scheme is proposed to traverse all the pixels in the cover image.

The rest of the paper is organized as follows. Section II gives a brief review to the PEE method. The proposed two-step clustering and optimizing scheme is presented in Section III. Experimental results compared to other pixel predictors are demonstrated in Section IV. And finally, Section V concludes this paper and discusses future research directions.

PREDICTION-ERROR EXPANSION (PEE)

Typical PEE based schemes divide cover image pixels into different parts, while a pixel of one part is predicted by its neighboring pixels in other parts.

Figure 1. Pixel clustering based on structural self-similarities of image patches



Take the rhombus pattern in Sachnev et al.'s double-layered embedding method (2009) for instance, in which all pixels are divided into two sets: the Cross set and the Dot set (see Figure 2). In the first round, the Cross set is used for embedding data and Dot set for computing predictions, while in the second round, the Dot set is used for embedding and Cross set for computing predictions. Since the two layers' embedding processes are similar in nature, we only take the Cross layer for illustration.

As shown in Figure 2, the Cross pixels $u_{i,j}$ in the cover image are collected into a sequence $\boldsymbol{u} = (u_1, u_2, \cdots, u_n)$ from left to right and from top to bottom. For each Cross pixel $u_{i,j}$, the rhombus predicted value $\hat{u}_{i,j}$ is computed using its four nearest Dot pixels:

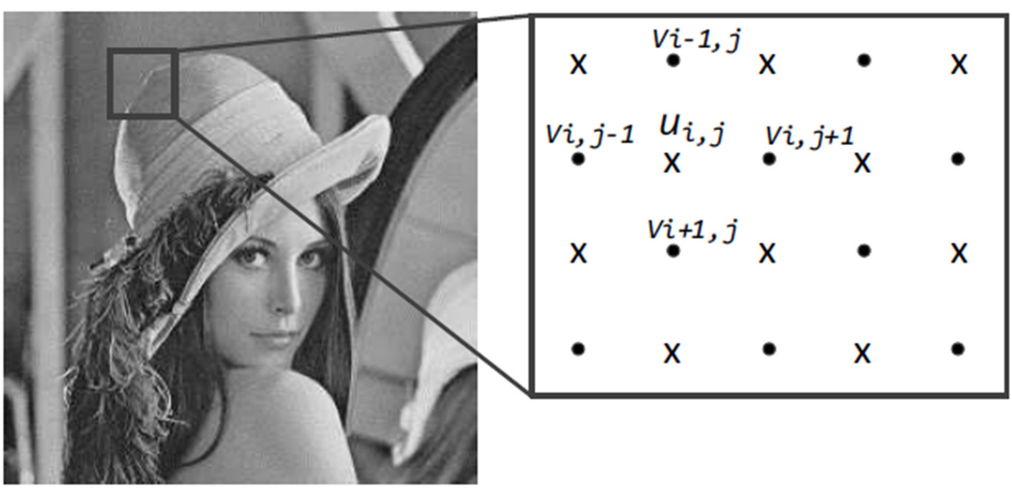
$$\hat{u}_{i,j} = \left\lfloor \frac{v_{i,j-1} + v_{i+1,j} + v_{i,j+1} + v_{i-1,j}}{4} \right\rfloor \tag{1}$$

Then, by subtracting the predicted value $\hat{u}_{i,j}$ from the original pixel value $u_{i,j}$, we obtain the prediction-error sequence $\boldsymbol{e} = (e_1, e_2, \cdots, e_n)$. Afterwards, secret data are embedded into the prediction-error sequence $\boldsymbol{e}$ through expanding and shifting. Specifically, for each e_i, it is expanded or shifted as:

$$e_i' = \begin{cases} 2e_i + m, & \text{if } e_i \in [T_n, T_p] \\ e_i + T_p + 1, & \text{if } e_i \in (T_p, +\infty) \\ e_i + T_n, & \text{if } e_i \in (-\infty, T_n) \end{cases} \tag{2}$$

where $T_n < 0$ and $T_p \geq 0$ are capacity-dependent integer valued parameter, and

Figure 2. Rhombus prediction pattern. The pixel value of u of the Cross set can be predicted by using the four neighboring pixel values of the Dot set and expanded to hide one bit of data



$m \in \{0,1\}$ is a to-be-embedded message bit. Here, the bins in $[T_n, T_p]$ are expanded to embed data, and those in $(-\infty,\ T_n) \cup (T_p, +\infty)$ are shifted outwards to create vacancies. Finally, each pixel value u_i is modified to $u_i^{'} = \hat{u}_i + e_i^{'}$ to obtain the marked image.

In PEE extraction procedure, the original prediction-error e_i is recovered from the marked prediction-error $e_i^{'}$ as:

$$e_i = \begin{cases} \lfloor e_i^{'} / 2 \rfloor, & \text{if } e_i^{'} \in [2T_n, 2T_p + 1] \\ e_i^{'} - T_p - 1, & \text{if } e_i^{'} \in (2T_p + 1, +\infty) \\ e_i^{'} - T_n, & \text{if } e_i^{'} \in (-\infty, 2T_n) \end{cases} \tag{3}$$

and the embedded message bits are extracted as the LSBs of those prediction-errors $e_i^{'} \in [2T_n, 2T_p + 1]$. Finally, the cover image is restored using the recovered prediction errors. Notice that, to guarantee the reversibility, the key point is that the prediction values used in extraction should be the same as that in embedding.

QUAD-LAYERED EMBEDDING SCHEME

For each pixel b in the cover image, instead of predicted by averaging its four nearest neighbors as shown in Figure 2, we compute its predicted value $\hat{b}$ through the linear combinations of its eight nearest neighbors $\boldsymbol{a} = (a_1, a_2, \cdots, a_8)$:

$$\hat{b} = \boldsymbol{a}\boldsymbol{x}^{\boldsymbol{T}} = (a_1, a_2, \cdots, a_8) \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_8 \end{pmatrix} \tag{4}$$

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_8)$ is the coefficients vector satisfies:

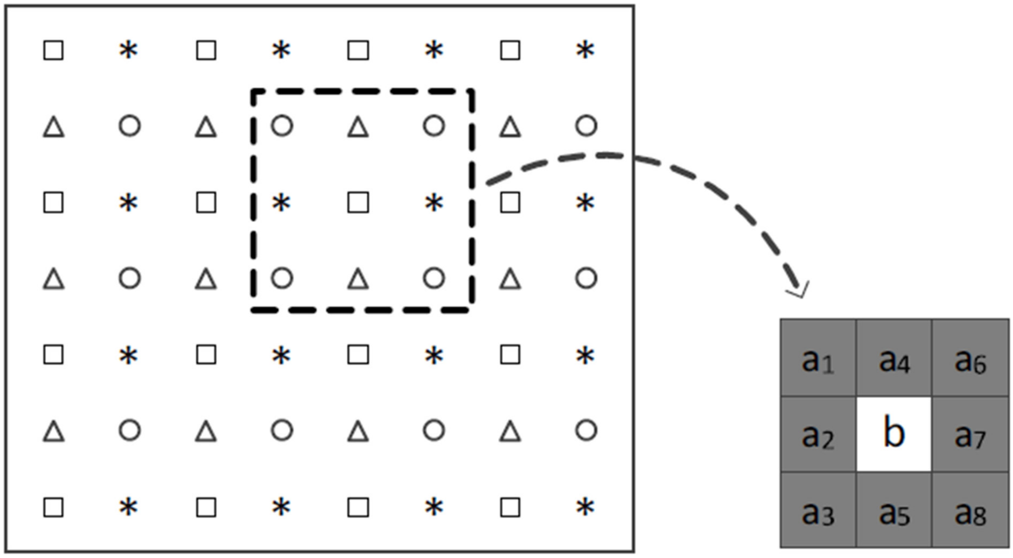
$$\begin{aligned} &\sum_{q=1}^{8} x_q = 1 \\ &0 \le x_q \le 1, q = 1, 2, \cdots, 8 \end{aligned} \tag{5}$$

As depicted in Figure 3, all the pixels in the cover image are divided into four sets: the Square set, the Star set, the Triangle set, and the Circle set. A pixel in each set is predicted by its eight neighbors from the other three sets. And in order to traverse all the pixels, a consecutive quad-layered embedding scheme is developed, while each layer covers a type of set. Without loss of generality, we take the Square layer for instance to elaborate our embedding scheme.

Two-Step Patch-Level Clustering

Firstly, all the image patches $b_i,\ i = 1, 2, \cdots, N_s$ in the Square set are roughly divided into two categories according to the smoothness of their prediction context vectors $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \cdots a_{i8})$. Here N_s is the total number of pixels in the Square set. The smoothness crite-

Figure 3. Quad-layered prediction pattern. All pixels in the cover image are divided into four sets, and a pixel in each set is predicted by its nearest eight neighboring pixels from the other three set.



rion v_i is defined by the variance of elements in $\boldsymbol{a}_i$:

$$v_i = \frac{1}{8}\sum_{j=1}^{8}(a_{ij} - m_i)^2 \tag{6}$$

where $m_i = \frac{1}{8}\sum_{j=1}^{8} a_{ij}$ is the average value of all the prediction context pixels. If the smoothness criterion v_i is less than a predefined threshold T_v, we classify the pixel b_i into the smooth category. Otherwise, the pixel b_i is classified into the complex category.

We can treat the above coarse classification as the first-step clustering, which results in two upper level clusters. Secondly, all the pixels b_i in the complex category are further subdivided into K clusters using the K-means clustering algorithm. In one way, the upper level smooth category can be treated as the $0\text{-}th$ cluster in the second-step clustering, so finally we get $K+1$ clusters together for all the pixels in the Square set.

Specifically, in the second-step clustering procedure, for each pixel b_i in the complex category, a patch-level structural feature f_i is calculated from its prediction context vector $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \cdots a_{i8})$ as:

$$f_i = \boldsymbol{a}_i - \frac{a_{i1} + a_{i2} + \cdots + a_{i8}}{8} \tag{7}$$

According to the extracted features f_i, we use the K-means clustering algorithm to subdivide all the pixels b_i in the complex category into K clusters as depicted in Figure 1. Here K is a predefined parameter for the K-means algorithm. Note that the initial cluster centroid pixel indexes for the K-means algorithm are selected every S pixels starting from the S-th pixel, namely $S, 2S, \cdots, KS$. And the interval parameter S is transmitted to the receiver side for the sake of repeating the K-means algorithm.

Optimizing Non-Local Pixel Predictors

In the two-step clustering procedure described above, all pixels $b_i,\ i = 1, 2, \cdots, N_s$ in the Square set are classified into $K+1$ clusters. After that, for all the pixels $b_j, j = c_1, c_2, \cdots, c_{N_k}$ assigned to a specified cluster, a content adaptive pixel predictor is estimated by optimizing the following problem:

$$\begin{aligned} &\text{minimize} \quad \left| A\boldsymbol{x} - \boldsymbol{b} \right| \\ &\text{subject to} \quad \sum_{q=1}^{8} x_q = 1 \\ &\qquad\qquad\quad 0 \le x_q \le 1, q = 1, 2, \cdots, 8 \end{aligned} \tag{8}$$

where $N_k,\ k = 0, 1, 2, \cdots, K$ is the total number of pixels dispatched to the cluster, and matrix A and vector $\boldsymbol{b}$ is given by:

$$A = \begin{vmatrix} a_{c_1 1} & a_{c_1 2} & \cdots & a_{c_1 8} \\ a_{c_2 1} & a_{c_2 2} & \cdots & a_{c_2 8} \\ \vdots & \vdots & \ddots & \vdots \\ a_{c_{N_k} 1} & a_{c_{N_k} 2} & \cdots & a_{c_{N_k} 8} \end{vmatrix} \quad \boldsymbol{b} = \begin{vmatrix} b_{c_1} \\ b_{c_2} \\ \vdots \\ b_{c_{N_k}} \end{vmatrix} \tag{9}$$

In (8), the l1-norm is used rather than the l2-norm due to the fact that we aims to optimize the eight coefficients $\boldsymbol{x} = (x_1, x_2, \cdots x_8)$ according to most of the pixels in the cluster, while the l1-norm is more robust to outliers. Another advantage of the l1-norm is that embedding modifications on the vector $\boldsymbol{b}$ result little changes for the optimized coefficients, this benefits us to reduce the overhead information for transmitting the optimized coefficients to the receiver side, which will be elaborated later.

When the eight coefficients $\boldsymbol{x} = (x_1, x_2, \cdots x_8)$ are estimated by solving (8), the prediction-error e_j is calculated as:

$$e_j = b_j - \left\lfloor \boldsymbol{a}_j \boldsymbol{x}^T \right\rfloor \tag{10}$$

Then we embedded messages by modifying e_j to $e_j^{'}$ using the expanding and shifting techniques described in (2), as a result the corresponding pixels values $b_j, j = c_1, c_2, \cdots, c_{N_k}$ are modified to:

$$b_j^{'} = e_j^{'} + \left\lfloor \boldsymbol{a}_j \boldsymbol{x}^T \right\rfloor \tag{11}$$

Compression of the Optimized Coefficients

For the receiver to calculate the modified prediction error $e_j^{'}$ correctly, the optimized coefficient $\boldsymbol{x} = (x_1, x_2, \cdots x_8)$ has to be recorded and transmitted to the receiver side. As mentioned in Section III-A, modifications on the vector $\boldsymbol{b}$ in (8) result little changes for the optimized coefficients $\boldsymbol{x} = (x_1, x_2, \cdots x_8)$. Using the modified vector $\boldsymbol{b}'$ after embedding, we can optimize (8) again and get a revised coefficients vector $\boldsymbol{x}' = (x_1', x_2', \cdots x_8')$. If we restrain the precision of the coefficients to d decimal places, a coefficients residual vector $\boldsymbol{r}$ can be derived as:

$$\begin{aligned} \boldsymbol{r} &= (r_1, r_2, \cdots r_8) \\ &= \lfloor \boldsymbol{x} \times 10^d \rfloor - \lfloor \boldsymbol{x}' \times 10^d \rfloor \end{aligned} \tag{12}$$

Next, we use a variable-length coding scheme to record the coefficients residual vector for each cluster. Specifically, we first check that the maximum absolute value of $\boldsymbol{r}$ is less than 2^M, otherwise we don't embed messages for this cluster. Here M is a predefined bit length. Then for each coefficient residual r_i, if r_i equals zero, we just add a bit "0" to the coding stream, or else we add $M+2$ bits to the coding stream. The $M+2$ bits consists of a bit "1", a sign indicator bit to record the sign of r_i, and M bits to record the absolute value of r_i. And finally, the coded coefficients residual bit streams for all the $K+1$ clusters are concatenated together to be transmitted to the receiver side.

Embedding and Extracting

This section describes the proposed quad-layered embedding process in detail and used ideas all discussed above. Figure 4 presents a simple block diagram representing the embedding and extracting processes. Before data embedding, all pixels are divided into four sets: Square set, Star set, Triangle set, and Circle set, each layer's embedding and extracting processes cover all pixels in one set.

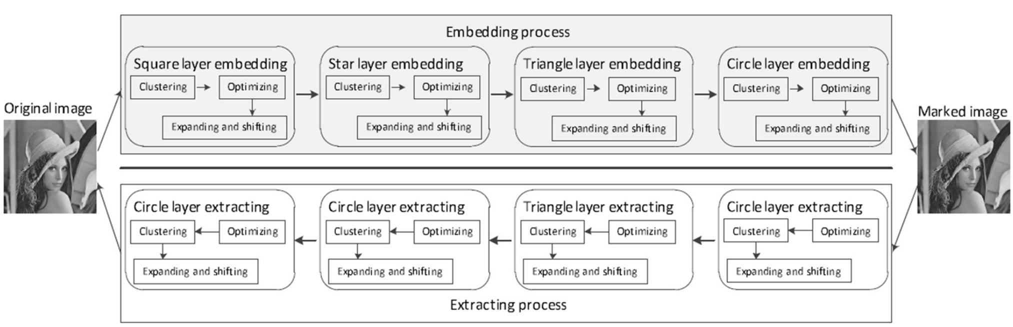
Since each layer's embedding and extracting processes are similar in nature, we discuss the Square layer to elaborate the procedures.

Embedding Process

The Square layer embedding process is designed as follows:

1. First-step clustering, divide all the pixels b_i, $i = 1, 2, \cdots, N_s$ in the Square set into the smooth category or the complex category;
2. Second-step clustering, extract features f_i from all the pixels b_i in the complex category, then select K pixel indexes every S spaced as

Figure 4. Framework of quad-layered embedding and extracting scheme



initial cluster centroids to run the K-means algorithm on the features space to subdivide all complex category pixels into K clusters;

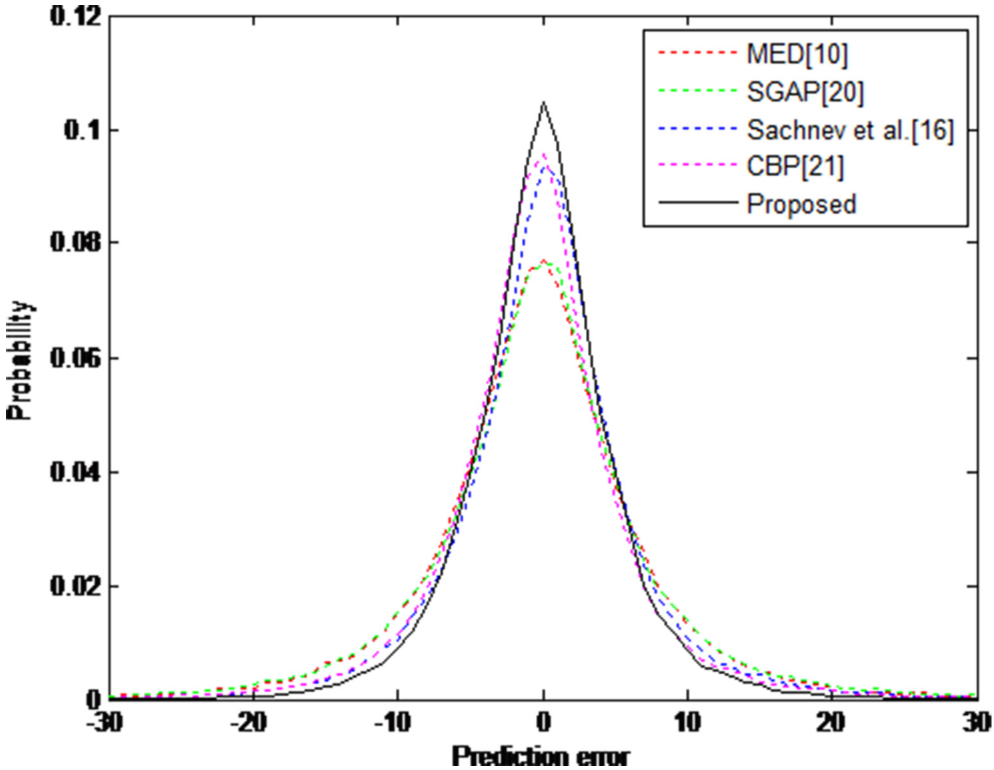
3. For $k = 0 : K$:
   a. Collect all pixels $b_j, j = c_1, c_2, \cdots, c_{N_k}$ and prediction context vectors $\boldsymbol{a}_j$ to form matrix A and vector $\boldsymbol{b}$ in (9);
   b. Estimate the optimal pixel predictor coefficients $\boldsymbol{x} = (x_1, x_2, \cdots x_8)$ by solving (8), and then calculate the corresponding prediction errors e_j using (10);
   c. Embed messages into the prediction errors e_j using expanding and shifting techniques described in (2), then calculate the coefficients residual vector $\boldsymbol{r}_k$ by (12) and generate the coded residual bit stream $\boldsymbol{r}_{sk}$ as discussed in Section III-C;
4. Finally, concatenate all the coded residual bit streams $\boldsymbol{r}_{sk}$, $k = 0,1,2,\cdots K$ together and embedded them into the LSBs of some preserved pixels.

Extracting Process

The Square layer extracting process is designed as follows:

1. Extract all the coded residual bit streams $\boldsymbol{r}_{sk}$, $k = 0,1,2,\cdots K$ from the LSBs of some preserved pixels;
2. First-step clustering, divide all the pixels b_i, $i = 1,2,\cdots,N_s$ in the Square set into the smooth category or the complex category;
3. Second-step clustering, extract features f_i from all the pixels b_i in the complex category, then select K pixel indexes every S spaced as initial cluster centroids run the K-means algorithm on the features space to subdivide all complex category pixels into K clusters;
4. For $k = 0 : K$:
   a. Collect all pixels $b'_j, j = c_1, c_2, \cdots, c_{N_k}$ and prediction context vectors $\boldsymbol{a}_j$ to form matrix A and vector $\boldsymbol{b}'$ in (9);
   b. Estimate the optimal pixel predictor coefficients $\boldsymbol{x}' = (x'_1, x'_2, \cdots x'_8)$ by solving (8), and together with the coded residual bit stream $\boldsymbol{r}_{sk}$, recover the original optimal pixel predictor coefficients $\boldsymbol{x} = (x_1, x_2, \cdots x_8)$;
   c. Calculate prediction errors e'_j using (11), then extract messages using expanding and shifting techniques described in (3), at the same time recover the original prediction errors e_j;
   d. Using the recovered prediction errors e_j, the original pixel values b_j can be restored losslessly through (10).

Figure 5. The prediction error histograms for all the five prediction methods on Lena image



SIMULATIONS

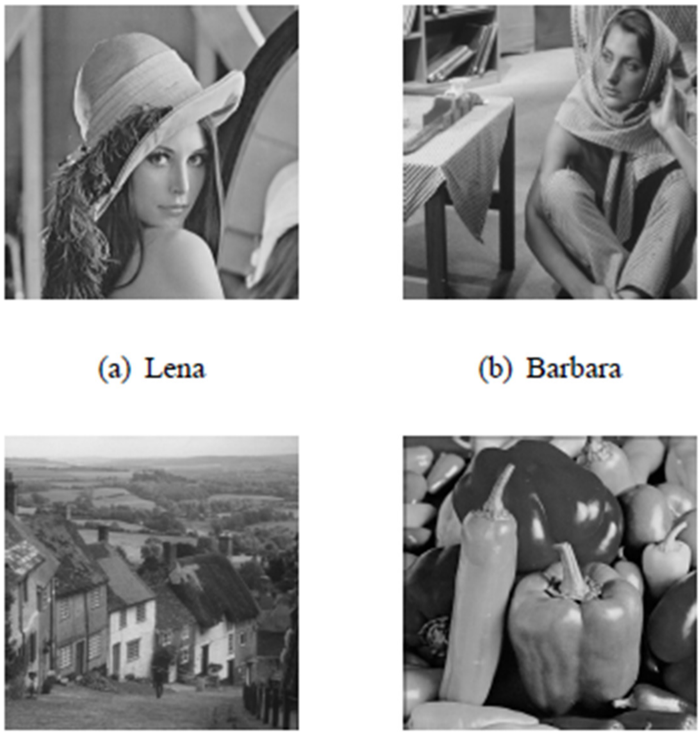
First, the proposed two-step clustering and optimizing scheme is compared to other four prediction methods considered, namely, MED (Thodi & Rodriguez, 2007), Sachnev et al.'s method (2009), the simplified gradient-adjusted predictor (SGAP) (Coltuc, 2011), and the checkerboard based prediction (CBP) (R.M, W, & Guo, 2014), using typical 512 × 512 gray scale images as shown in Figure 6. Table 1 records the comparison results of prediction accuracy in terms of MAE (mean absolute error), which is defined by:

$$MAE = \frac{1}{n} \times \sum_{i=1}^{n} \left| b_i - \hat{b}_i \right| \tag{13}$$

Here for simplicity, we only process the Square set pixels for comparison, mean only a quarter of pixels are predicted in the cover image. It can be observed from Table 1 that our clustering and optimizing scheme provides the best prediction accuracy among all the competitors for all the four test images.

For better visualization, the prediction error histograms for all the five prediction methods on the test Lena image are presented in Figure 5, from which

Figure 6. Test images



(a) Lena (b) Barbara

(c) Goldhill (d) Peppers

Table 1. Prediction accuracy (mean absolute error) comparison

Image	Lena	Barbara	Goldhill	Pepper
MED	4.4168	9.2978	5.5387	5.3624
SGAP	4.0362	8.8888	5.6568	4.8651
Sachnev et al.	3.2330	7.4485	4.5441	4.1278
CBP	3.1837	7.0533	4.5579	3.9778
Proposed	2.7294	4.3582	3.8348	3.5356

we can see that our proposed prediction method offers the sharpest prediction error histogram in comparison with other prediction methods.

Generally speaking, for the PEE based RDH schemes, a sharper prediction error histogram will lead to a better rate distortion embedding performance.

By observing (2), we can see that the embedding capacity is determined by the total number of prediction errors falling in the interval $[T_n, T_p]$, so the sharper the prediction error histogram is, the higher the embedding capacity can be achieved. And vice versa, for a given embedding payload, the sharper the histogram, the less the number of pixels falling out the interval $[T_n, T_p]$, which will result in less shifting distortion.

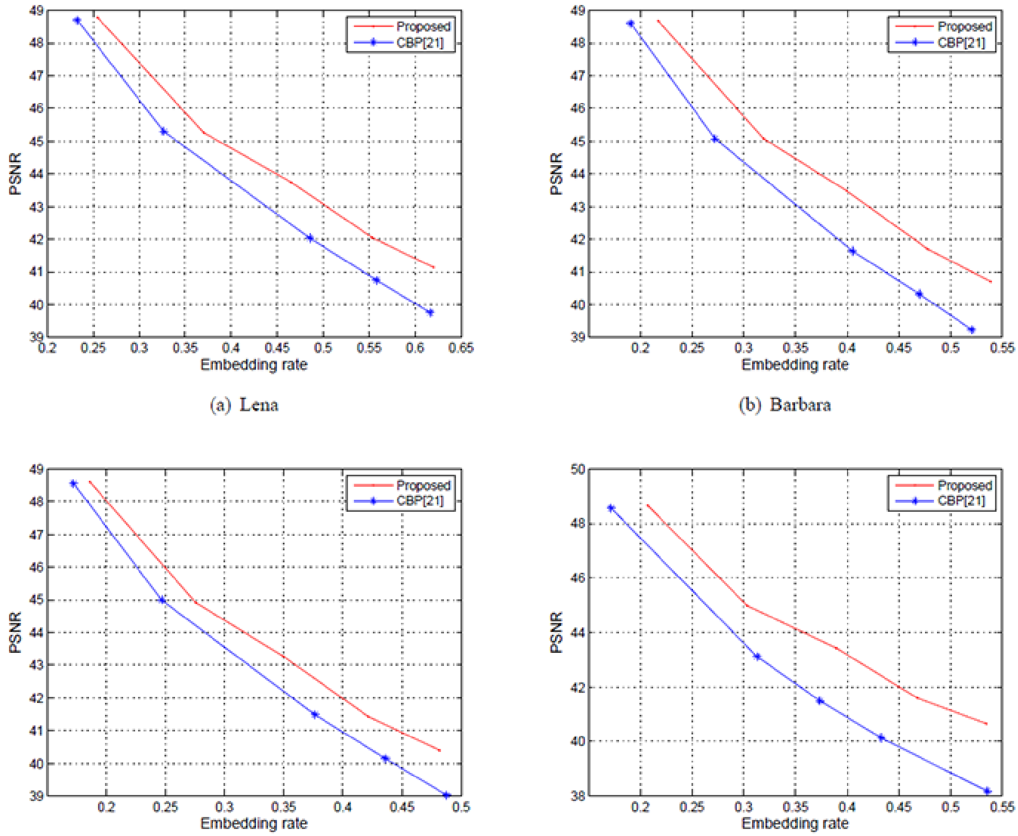
For our proposed two-step clustering and optimizing scheme, the main weakness is the extra bits to record the coded residual bit streams $\boldsymbol{r}_{sk},\ k = 0,1,2,\cdots K$ introduced in Section III-C. Next, to demonstrate that prediction accuracy directly influence the embedding performance of PEE based RDH methods. We compare our proposed quad-layered embedding scheme with the CBP method (R.M et al., 2014) to process all the pixels in the cover image. Both prediction methods are combined with the PEE scheme in Section II to embed messages. For our proposed scheme, the smooth threshold T_v is set to 25, the cluster parameter K is set to 25, the decimal place parameter d and bit length parameter M in Section III-B are set with $d = 2$ and $M = 3$. Embedding performance comparison results for various embedding rates are demonstrated by Figure 7.

Observing from Figure 7, for the Lena image and the Goldhill image, our proposed scheme earns 1.0dB higher PSNR than the CBP method (R.M et al., 2014) on average. And for the Barbara image and especially the Peppers image, the gains of PSNR increase to 2dB on average. As we can see in Figure 6, the Barbara image and the Peppers image exploit better structural self-similarities than the other two images, and this is why our clustering and optimizing scheme performs much better than the CBP method (R.M et al., 2014) method.

CONCLUSION

This paper presents a two-step clustering and optimizing pixel prediction method for prediction-error expansion (PEE) based reversible data hiding, which exploits self-similarities and group structural information of non-local image patches. And in order to traverse all the pixels in the cover image, a quad-layered embedding scheme is proposed accordingly. Compared to other fixed or content adaptive pixel predictors that treat each pixel independently, our proposed method offers the best prediction accuracy. Experimental results imply that structural self-similarities of intra non-local image patches is a good property to benefit pixel prediction, so structural self-similarities across multiple images or even between a dataset of images

Figure 7. Comparisons in terms of rate-distortion performance



(a) Lena (b) Barbara

(c) Goldhill (d) Peppers

could be more helpful to reversible data hiding methods. This may be our future research direction to work on.

ACKNOWLEDGMENT

This work was supported in part by the Natural Science Foundation of China under Grant 61170234 and Grant 60803155, by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA06030601, and by the Funding of Science and Technology on Information Assurance Laboratory under Grant KJ-13-003.

REFERENCES

Bao, F., Deng, R. H., Ooi, B. C., & Yang, Y. (2005, December). Tailored reversible watermarking schemes for authentication of electronic clinical atlas. *IEEE Transactions on Information Technology in Biomedicine*, *9*(4), 554–563. doi:10.1109/TITB.2005.855556 PMID:16379372

Caldelli, R., Filippini, F., & Becarelli, R. (2010, Jan). Reversible watermarking techniques: An overview and a classification. *EURASIP J.Inf. Secur, 2010*(2).

Chung, K., Huang, Y.-H., Chang, P.-C., & Liao, H.-Y. M. (2010, November). Reversible data hiding-based approach for intra-frame error concealment in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, *20*(11), 1643–1647. doi:10.1109/TCSVT.2010.2077577

Coltuc, D. (2011, September). Improved embedding for prediction-based reversible water- marking. *IEEE Trans. Inf. Forens. Security*, *6*(3), 873–882. doi:10.1109/TIFS.2011.2145372

Fallahpour, M. (2008, October). Reversible image data hiding based on gradient adjusted prediction. *IEICE Electronics Express*, *5*(20), 870–876. doi:10.1587/elex.5.870

Feng, J. et al. (2006, May). Reversible watermarking: Current status and key issues. *International Journal of Network Security*, *2*(3), 161–171.

Fridrich, J., Goljan, M., & Du, R. (2002, January). Lossless data embedding for all image formats. *Proceedings of the Society for Photo-Instrumentation Engineers*, *4675*, 572–583. doi:10.1117/12.465317

Hu, X., Zhang, W., Hu, X., Yu, N., Zhao, X., & Li, F. (2013, May). Fast Estimation of Optimal Marked-Signal Distribution for Reversible Data Hiding. *IEEE Trans. on Information Forensics and Security*, *8*(5), 779–788. doi:10.1109/TIFS.2013.2256131

Hu, X., Zhang, W., & Yu, N. (2014, June). Optimizing Pixel Predictors Based on Self-similarities for Reversible Data Hiding. Accepted by *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*.

Li, X., Yang, B., & Zeng, T. (2011, December). Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection. *IEEE Transactions on Image Processing*, *20*(12), 3524–3533. doi:10.1109/TIP.2011.2150233 PMID:21550888

Luo, L. et al. (2010, March). Reversible image watermarking using interpolation technique. *IEEE Trans. Inf. Forensics Security*, *5*(1), 187–193. doi:10.1109/TIFS.2009.2035975

Ni, Z., Shi, Y., Ansari, N., & Wei, S. (2006, March). Reversible data hiding. *IEEE Transactions on Circuits and Systems for Video Technology*, *16*(3), 354–362. doi:10.1109/TCSVT.2006.869964

Ou, B., Li, X., Zhao, Y., & Ni, R. (2013, October). Reversible data hiding based on PDE predictor. *Journal of Systems and Software*, *86*(10), 2700–2709. doi:10.1016/j.jss.2013.05.077

Peng, F., Li, X., & Yang, B. (2012, January). Adaptive reversible data hiding scheme based on integer transform. *Signal Processing*, *92*(1), 54–62. doi:10.1016/j.sigpro.2011.06.006

Rad, R. M., KokSheik Wong, , & Jing-Ming Guo, R.M. (2014, April). A unified data embedding and scrambling method. *IEEE Transactions on Image Processing*, *23*(4), 1463–1475. doi:10.1109/TIP.2014.2302681 PMID:24565789

Sachnev, V., Kim, H., Nam, J., Suresh, S., & Shi, Y. (2009, July). Reversible watermarking algorithm using sorting and prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, *19*(7), 989–999. doi:10.1109/TCSVT.2009.2020257

Shi, Y. (2004). Reversible data hiding. *Proc. IWDW* (vol. 3304, pp. 1-12).

Shi, Y., Ni, Z., Zou, D., Liang, C., & Xuan, G. (2004, May). Lossless data hiding: fundamentals, algorithms and applications. *Proc. IEEE ISCAS* (vol. 2, pp. 33-36). doi:10.1109/ISCAS.2004.1329201

Thodi, D., & Rodriguez, J. (2007, March). Expansion embedding techniques for reversible watermarking. *IEEE Transactions on Image Processing*, *16*(3), 721–730. doi:10.1109/TIP.2006.891046 PMID:17357732

Tian, J. (2003, August). Reversible data embedding using a difference expansion. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*(8), 890–896. doi:10.1109/TCSVT.2003.815962

Tsai, P., Hu, Y., & Yeh, H. (2009). Reversible image hiding scheme using predictive coding and histogram shifting. *Signal Processing*, *89*(6), 1129–1143. doi:10.1016/j.sigpro.2008.12.017

Wang, C., Li, X., & Yang, B. (2010). Efficient reversible image watermarking by using dynamical prediction-error expansion. *Proc. IEEE ICIP* (pp. 3673-3676).

Wu, H., & Huang, J. (2012, December). Reversible image watermarking on prediction errors by efficient histogram modification. *Signal Processing*, *92*(12), 3000–3009. doi:10.1016/j.sigpro.2012.05.034

Yang, W., Chung, K., Liao, H., & Yu, W. (2013, February). Efficient reversible data hiding algorithm based on gradient-based edge direction prediction. *Journal of Systems and Software*, *86*(2), 567–580. doi:10.1016/j.jss.2012.09.041

Zhang, W., Hu, X., Li, X., & Yu, N. (2013, July). Recursive Histogram Modification: Establishing Equivalency between Reversible Data Hiding and Lossless Data Compression. *IEEE Transactions on Image Processing*, *22*(7), 2775–2785. doi:10.1109/TIP.2013.2257814 PMID:23591495