Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

Reversible Data Hiding with Context Modeling, Generalized Expansion and Boundary Map

Wei Fan · Zhenyong Chen · Ming Chen · Lixin Luo · Zhang Xiong

Received: date / Accepted: date

Abstract This paper proposes a reversible data hiding scheme with high capacity-distortion efficiency, which embeds data by expanding prediction-errors. Instead of using the MED predictor as did in other schemes, a predictor with context modeling, which refines prediction-errors through an error feedback mechanism, is adopted to work out prediction-errors. The context modeling can significantly sharpen the distribution of prediction-errors, and benefit the embedding capacity and the image quality. To expand prediction-errors, the proposed scheme utilizes a generalized expansion, which enables it to provide capacities larger than 1 bpp (bits per pixel) without resorting to multiple embedding. Besides, a novel boundary map is proposed to record overflow-potential pixels. The boundary map is much shorter compared with either a location map or an overflow map even though it is not compressed. The combination of the context modeling, the generalized expansion and the boundary map makes the overall scheme efficient in pursuing large embedding capacity and high image quality. Experimental results demonstrate that the proposed scheme provides competitive capacity compared with other state-of-the-art schemes when the image quality is kept at the same level.

Wei Fan
School of Computer Science and Engineering, Beihang University, Beijing, 100191 China
Tel.:+86-010-82338199
E-mail: fanwei@cse.buaa.edu.cn

Zhenyong Chen
School of Computer Science and Engineering, Beihang Universtiy, Beijing, 100191 China
E-mail: chzhyong@buaa.edu.cn

Ming Chen
School of Computer Science and Engineering, Beihang University, Beijing, 100191 China
E-mail: mingchen@cse.buaa.edu.cn

Lixin Luo
School of Computer Science and Engineering, Beihang University, Beijing, 100191 China
E-mail: lixinluo@cse.buaa.edu.cn

Zhang Xiong
School of Computer Science and Engineering, Beihang University, Beijing, 100191 China
E-mail: xiongz@buaa.edu.cn

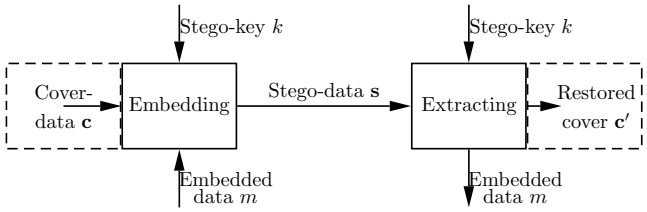


Fig. 1 Model of Reversible Data Hiding. The stego-data $\mathbf{s}$ should be perceptually close to the cover-data $\mathbf{c}$ to keep distortion low, and the cover-data should be recovered exactly to guarantee reversibility $\mathbf{c}' \equiv \mathbf{c}$. The reversible data hiding is one type of fragile data hiding and any slight modification to the host with data hided leads to the data extraction failure and the original one is not able to be recovered. So the transmission of $\mathbf{s}$ is considered to be noise-free in this paper.

Keywords Boundary map · context modeling · generalized expansion · reversible data hiding · watermarking

1 Introduction

Recently, data hiding has become a research hotspot thanks to the increasing popularity of digital multimedia applications. Data hiding is a technique that imperceptibly hides data into cover media contents such as audios, images, and videos. Being the essential of watermarking and steganography, data hiding is widely adopted in applications including copyright protection, content authentication and media asset management.

In conventional data hiding methods, a common problem is that the original host is inevitably distorted due to data hiding itself because of quantization, replacement, roundoff or truncation [8]. This distortion is unacceptable when exact representation of the host is demanded, e.g. in the annotation of valuable media documents or in military and medical imaging. Under these circumstances, reversible data hiding is needed to provide embedding payload without causing permanent irreparable distortion and exactly recover the host to the orginal state after data extraction the model of which is presented in Fig. 1.

Since its first invention [2], dozens of reversible data hiding schemes have been reported in the literature. Among the various categories of reversible data hiding schemes, two of them attract the most of the researchers' attention. They are 1) reversible data hiding using *Difference Expansion* (DE), 2) and reversible data hiding using *Histogram Operation* (HO).

Reversible data hiding using difference expansion was first proposed by Tian [24]. In the literature, many difference expansion based schemes have been reported, which can be further divided into two groups regarding the origins of the differences.

Schemes in the first group work out differences by comparing individual pixels. They include the schemes proposed in [1, 4, 9, 13, 16, 19, 24, 26, 27]. Since differences among closer pixels are smaller, comparisons are confined within small pixel blocks. So the scheme in this group is also called reversible data hiding schemes using *Block Difference Expansion* (BDE). It is worth to notice that not all differences are expandable due to perceptible artifact and the overflow[1] problem. Only differences within a

[1] In this paper, we use overflow to represent both overflow and underflow as a simplification.

confined range (e.g. $T_l \leq d \leq T_r$, where T_l and T_r are two thresholds and d is the difference) are expanded for data hiding, and to differentiate expanded differences from non-expanded ones, a bitmap named *location map* is often used. DE is generalized and applied to triplets and quads by Alattar [1], who proposed several DE schemes of better performance with enlarged overall capacity and lowered overhead cost. Schemes reported in [4, 26, 27] further improved the performance of DE by utilization of those differences that are impossible to overlap with the expanded differences to either provide more expandable differences or reduce the cost of the location map. By adaptively choosing the median pixel as the minuend for difference making, Lee [16] proposed a centralized difference expansion scheme in which a larger embedding capacity was gained. Kim et al. [13] proposed a novel difference expansion method with a simplified location map and new expandability. Furthermore, Lin et al. [19] proposed a DE scheme without location map, which completely gets rid of the large overhead cost. Hsiao et al. [9] partitioned the cover image into 3×3 blocks and classified them into smooth, normal and complex blocks. By embedding 2 bits into smooth blocks, their embedding capacity can be larger than 1 bpp in single embedding.

Schemes using difference expansion falling into the second group are reported in [3, 10,15,22,23]. Being different from the schemes in the first group, these schemes achieve reversible data hiding by expanding prediction-errors, i.e. the differences between the original and the predicted values of pixels. These schemes are called *Prediction-Error Expansion* (PEE) schemes. Thodi et al. proposed the first PEE scheme in [22], wherein predicted values are calculated by a predictor named Median Edge Detector (MED). As there are more prediction-errors than block differences, larger embedding capacities were reported in [22]. However, for the same reason, the size of location map is also larger. To overcome this drawback, Thodi et al. utilized a histogram shift to disambiguate expanded pixels and non-expanded pixels instead of using a location map, and only recorded those overflow pixels with a more compressible overflow map. Besides, Kuribayshi [15] also used the MED predictor for prediction. In [23], Thodi et al. also applied a histogram operation to prediction-errors and replaced the location map with a more compressible overflow map. Hu et al. proposed a scheme with an improved overflow map in [10], where the overflow map was significantly condensed and larger pure capacities were reported.

Reversible data hiding schemes using histogram operation (HO) include schemes proposed in [5–7, 11, 14, 17, 18, 21, 25]. The scheme proposed by Ni et al. [21] utilized a peak-zero pair of the image histogram for data hiding. Hwang et al. [11] improved Ni's scheme by taking zero histogram bins on both sides into consideration, and Chung et al. [7] gained more efficient trade-off between capacity and distortion by choosing different maximum and minimum histogram bins through dynamic programming. The capacities of schemes in [7,11,21] are quite limited because it is rare for the value of the peak histogram bin (the bin in the x-axis which has the largest value in the y-axis in the histogram) to be very large. By applying the operation to histogram of differences instead of histogram of pixels, schemes [5, 6, 14, 17, 18, 25] achieve much better performances. In [17], the differences are three-pixel block differences, namely differences between the two side pixels and the central pixel. In [14], differences are variances in four subimages of the cover-image. The scheme [18] is described to be a histogram operation on differences of two neighboring pixels. However, the differences therein are not block differences, but actually prediction-errors of a 1-order linear prediction. On the contrary, the scheme [25] is indeed a histogram operation on block differences, although it is entitled as if the differences are prediction-errors. In [5], the histogram

operation is generalized and formulated as additive prediction-error expansion, and a simpler but more efficient predictor is used to obtain prediction-errors. By exploiting the omnidirectional correlation of pixels, a novel full context predictor is proposed [6] as an effectual manner of enhancing the performance of data hiding. For most schemes using HO [5–7, 11, 14, 17, 18, 21, 25], their capacities for single embedding are smaller compared to the schemes using DE. However, they can achieve rather competitive performance through multiple embedding [5, 17, 18], and they have small overhead since no location map is needed.

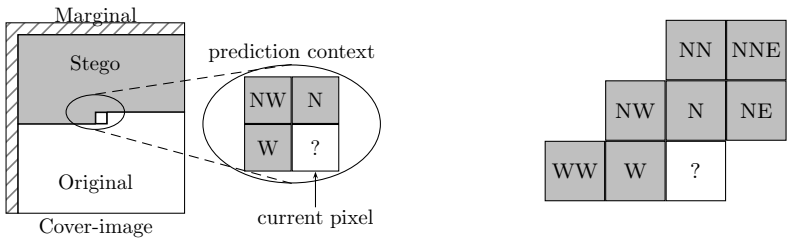
This paper presents a reversible data hiding scheme which is distinguished from other schemes with three features. 1) Context modeling is utilized to enhance the performances of reversible data hiding. The feasibility of reversible data hiding is largely due to existence of data redundancy. While prediction only exploits redundancy within a small local area, context modeling also exploits the redundancy within high-order structures like edges and textures. The context modeling can enlarge capacity and alleviate distortion of reversible data hiding by refining prediction-errors through an error feedback mechanism. 2) A generalized expansion is proposed to expand prediction-errors. A drawback of the expansion method in prior schemes using prediction-error expansion is that the embedding capacity is limited to 1 bpp, and when that limit is approached, capacity grows very slowly whereas the image quality degrades sharply. With the generalized expansion method, embedding capacity is no longer restricted to 1 bpp and the image quality at high capacity rate is significantly improved. 3) A novel boundary map is proposed as part of the scheme to record overflow-potential pixels. A boundary map fulfills the task of a location map or an overflow map, however, in a more efficient manner. Even without being compressed, the boundary map is much more compact compared with the latest technique serving the same purpose. The combination of context modeling, the generalized expansion method, and the boundary map makes the proposed scheme efficient in capacity-fidelity trade-off.

The remainder of this paper is structured as follows. Details of the proposed scheme including prediction with context modeling, generalized expansion and boundary map are described in Section 2. Capacity control, overhead data, the embedding and extracting process as implementation details are covered in Section 3. Experimental results are presented and evaluated in Section 4. Finally, conclusions are drawn in Section 5.

2 The Proposed Scheme

2.1 Prediction With Context Modeling

In the proposed scheme, prediction-errors are expanded to embed data. Prediction-errors are the differences between pixel values and their predicted values, i.e. $e = x - \hat{x}$, where $\hat{x}$ is the predicted value of pixel x. The predicted value $\hat{x}$ is given by a predictor which operates on a context containing causal pixels of current pixel x. Take the MED predictor [28] for example. It operates on a context containing three causal pixels as illustrated in Fig. 2(a), where the cover-image is marked with three different patterns. The hatched part identify the marginal area, where pixels are useless because of lack of prediction contexts. The gray part is the stego-area, where pixels are already altered through data hiding. The white part is the original area, which is not yet but to be processed.



(a) MED prediction context (b) GAP prediction context

Fig. 2 Prediction Contexts. Causal pixels are identified with their relative position the current pixel, e.g. x_n represents the pixel north to x.

After prediction-errors are expanded, e.g. $e' = 2e + b$ where e' is the expanded prediction-error and b is the secret bit, the resulting pixels are obtained through $x' = \hat{x}+e'$. Note that prediction involves only causal pixels, so the same predicted value $\hat{x}$ can be recalculated in the extracting process. Then the original pixel can be restored once the prediction-error e is recovered from $e' = x' - \hat{x}$ with the inverse of the generalized expansion.

Though prediction is good at exploiting image redundancy by decorrelating pixels, it alone cannot achieve total decorrelation in general. In lossless image compression, many excellent predictive compression schemes use some form of context modeling to further exploit correlation following a prediction stage. Context modeling can detect redundancy within high-order structures like edges and textures, which is beyond the capability of normal linear or piecewise linear prediction like MED. In previous reversible data hiding schemes, only prediction is used to calculate prediction-errors, whereas the context modeling, which can further refine prediction-errors through an error feedback mechanism, is never used. In this paper, we study the context modeling and add it into our reversible data hiding scheme. Particularly, we utilize the CALIC predictor [29], which is a predictor with context modeling, to calculate prediction-errors.

CALIC predictor operates on two stages. The first stage of the CALIC predictors is named Gradient Adjusted Prediction (GAP) [29]. It is carried out by a simple adaptive predictor that can adapt itself to the intensity gradients near the predicted pixels. Being different from traditional DPCM-like prediction, GAP weights the neighbors of current pixel according to the estimated local gradients of the image. Thus it is more robust than linear predictors particularly in areas of strong edges. The GAP operates on a context containing seven adjacent pixels as illustrated in Fig. 2(b). The coefficients of GAP are adaptively adjusted to the local gradient ot the predicted pixel. The GAP stage only removes part of the interpixel redundancy in the cover-image. The second stage functions the error feedback mechanism through context modeling. The variablility of prediction-errors still strongly correlates to the smoothness of the image around the predicted pixels. The CALIC caculates the error feedback to adaptively revise the prediction-errors.

The CALIC prediction seems sophisticated, however, its computation is efficient owing to the appropriate simplifications like quantization and rescaling.

2.2 The Generalized Expansion

The proposed generalized expansion is

$$e' = e \times n + (b)_n + c, \quad T_l \leq e \leq T_r, \tag{1}$$

where e' is the expanded prediction-error, $(b)_n$ is a n-ary secret digit to be embedded, c is a constant integer, and T_l and T_r are two thresholds to control the embedding capacity. Since the radix of the secret digit is n (n is an integer larger than 1), the generalized expansion embeds $\log_2 n$ bit(s) into a prediction-error. When n assumes a value larger than 2, more than one bit is embedded, so the capacity limit becomes larger than 1 bpp. The generalized expansion is reversible because we can extract the embedded secret digit through

$$(b)_n = (e' - c)\%n \tag{2}$$

and recover the original prediction-error through

$$e = \lfloor (e' - c)/n \rfloor. \tag{3}$$

The distortion caused by the generalized expansion is related to e, n, $(b)_n$, and c, which is obvious from

$$\Delta x = x' - x = e' - e = e \times (n-1) + (b)_n + c, \tag{4}$$

where x is the original pixel value, x' is the pixel value after the generalized expansion, and Δx is the distortion. In (4), the constant c is determined in the way that the average distortion can be alleviated by adding c. For example, as the range of expanded prediction-errors is $[T_l, T_r]$, c is most likely to be $(T_l + T_r) \times (n-1)/2$. Beside c, n and $(b)_n$ are always small integers, whereas e varies in a rather wide dynamic range. From (4), it is easy to observe that larger prediction-errors generally bring about severer distortion. So it is desirable to obtain a precise predictor as it produces small errors.

Note in (1) that only prediction-errors within $[T_l, T_r]$ are expanded. To guarantee reversibility, the locations of the expanded prediction-errors should also be recorded. A location map marking all expanded locations is a usual technique to fulfill this task. However, the location map can consume a large portion of the payload, especially when the embedding rate is low. An efficient alternative is to shift the histogram of prediction-errors to make the unexpanded prediction-errors not overlapped with the expanded ones. The histogram shift is

$$e' = \begin{cases} e + (T_r + 1) \times (n-1) + c, & e > T_r \\ e + T_l \times (n-1) + c \quad\quad , & e < T_l \end{cases}. \tag{5}$$

Since $0 \leq (b)_n \leq n-1$, we know that expanded prediction-errors should satisfy

$$T_l \times n + c \leq e' \leq T_r \times n + (n-1) + c. \tag{6}$$

With (5), we have unexpanded prediction-errors be either

$$e' > T_r \times n + (n-1) + c, \tag{7}$$

or

$$e' < T_l \times n + c. \tag{8}$$

Therefore, we can tell whether prediction-errors are expanded or not by testing condition (6). The recovery counterpart of this histogram shift is

$$e = \begin{cases} e' - (T_r + 1) \times (n-1) - c, & e' > T_r \times n + (n-1) + c \\ e' - T_l \times (n-1) - c \quad\quad , & e' < T_l \times n + c \end{cases} . \tag{9}$$

Moreover, to avoid the overflow problem, the expansion (1) should be further constrained to satisfy

$$\begin{cases} x + e \times (n-1) + (n-1) + c \leq 255 \\ x + e \times (n-1) + c \geq 0 \end{cases} , \tag{10}$$

since pixel values become $x' = x + e \times (n-1) + (b)_n + c$ after their prediction-errors are expanded.[2] The histogram shift should be constrained as well to satisfy

$$\begin{cases} x + T_r \times (n-1) + (n-1) + c \leq 255 \\ x + T_l \times (n-1) + c \geq 0 \end{cases} . \tag{11}$$

Overflows occur when prediction-errors fall within $T_l \leq e \leq T_r$ and the corresponding pixels conflict with condition (10), or when prediction-errors are not within $T_l \leq e \leq T_r$ and the corresponding pixels conflict with condition (11). These pixels should not be altered to avoid any occurrence of overflow, and should be recorded to tell that they are not altered. In schemes [10, 23], overflow maps, which are similar to location map but are more compressible, were used to record overflow-potential pixels. In this paper, we propose the boundary map, which will be described in detail in Section 2.3, to serve the same purpose of recording overflow-potential pixels.

The generalized expansion is a comprehensive data hiding strategy, and the underlying data embedding strategies of the two categories of reversible data hiding can be interpreted with it.

The case of DE [24], $d' = 2d + b$, is intuitive, which is obtainable by simply setting $n = 2$ and $c = 0$ of (1). The difference is that the expanded targets d in DE are not predictor-errors but actually the high-frequent components of the integer Haar wavelet transform. However, we can dispose the wavelet transform and obtain the same difference d using just $d = x_1 - x_2$, where x_1 and x_2 are the values of two adjacent pixels. Then the stego-pixels become

$$\begin{aligned} x_1' &= x_1 \\ x_2' &= x_2 - (x_1 - x_2) - (b)_2 \end{aligned} \tag{12}$$

instead of

$$\begin{aligned} x_1' &= x_1 + \lfloor (x_1 - x_2)/2 \rfloor + (b)_2 \\ x_2' &= x_2 - \lfloor (x_1 - x_2 + 1)/2 \rfloor \end{aligned} , \tag{13}$$

where x_1' and x_2' denote the values of x_1 and x_2 after the secret bin $(b)_2$ is embedded. We see that the integer transform is not indispensable, and its real contribution is to alleviate the distortion by averaging the change to both pixels. This also holds for the generalized integer [1] which is applicable to pixel blocks of all sizes.

As to the case of HO, we take Ni et al.'s scheme [21] for illustration. Actually, the peak-zero approach of scheme [21] is a combination of (1) and (5). In [21], pixels at

[2] We are considering 8-bit grayscale images in this paper, i.e., pixel values are between 0 and 255.

the peak histogram bin M are unchanged when '0' bits are embedded, or else they are changed to $M+1$ if '1' bits are embedded. This can be expressed as

$$x' = x + (b)_2, \quad x = M. \tag{14}$$

By setting $n = 2$, $c = -M$ and $T_l = T_r = M$, we get the exact formula of (14) from (1). To be reversible, scheme [21] also shifted pixels larger than M right by 1, namely

$$x' = x + 1, \quad x > M. \tag{15}$$

The generalized expansion is not only applicable to prediction-errors, it can be applied to other kinds of difference and pixel values as well. The generalized expansion is comprehensive and flexible, both of the two introduced data hiding strategies, i.e. difference expansion and histogram operation, can be interpreted with it. As shown, DE and HO are actually two special cases of the generalized expansion. Moreove, with the generalized expansion, $\log_2 n$ bits can embedded in each embeddable pixel. Capacities larger than 1 bpp in can be easily achieved in single embedding when $\gamma \times \log_2 n > 1$, where γ is the embedding rate.

2.3 The Boundary Map

To achieve reversibility, the overflow problem should be prevented. However, during the data embedding process, the pixel values might exceed the allowable range (for example, [0,255] for 8-bit grayscale images). So the locations of the overflow-potential pixels should be recorded as overhead information based on which the original image can be recovered losslessly during the data extracting process. And the location map and overflow map appear to play this important role. In DE schemes using location maps, overflow-potential pixels are not processed and marked as unexpanded in the location maps, whereas in DE schemes using overflow maps, overflow-potential pixels are specifically recorded using the payload-independent overflow maps. Although overflow maps are much more compressible than location maps, their compressed sizes are still expensive burdens to the overall capacity, and that is why Hu et al. were concentrating on improving the overflow map in their paper [10]. However, the compressed size of the improved overflow map is still large. To solve this problem, we propose a boundary map to record those overflow-potential pixels. Instead of being a 2-D map, the boundary map is actually a binary array. However, we still name it as a map since its function is to "find" the locations of those overflow-potential pixels.

For the generalized expansion, overflows can happen during both the expansion (1) and the histogram shift (5). To avoid it, pixels should be constrained to satisfy condition (10) for the expansion, or condition (11) for the histogram shift. Since only pixels within $[T_l, T_r]$ are expanded, condition (11) is stricter than condition (10). As long as condition (11) is satisfied, all overflows are prevented no matter it is for expansion or histogram shift. To find pixels not satisfying (11), we give the following definition:

Definition 1 For the generalized expansion (1), *boundary pixels* are pixels satisfying

$$x > 255 - (T_r + 1) \times (n - 1) - c \tag{16}$$

or

$$x < -T_l \times (n - 1) - c. \tag{17}$$

We see that boundary pixels are actually pixels conflicting with condition (11). All overflows can be avoided if we leave the boundary pixels unchanged during the data embedding process. However, during the extracting process, we cannot tell whether a pixel is a boundary pixel or not by simply testing (16) and (17), since it is possible for a non-boundary pixel to become a boundary pixel during the embedding process. To solve this ambiguity, we give another definition:

Definition 2 For the generalized expansion (1), *pseudo-boundary pixels* are pixels satisfying

$$255 - (T_r + 1) \times (n - 1) - c \geq x > 255 - 2(T_r + 1) \times (n - 1) - 2c \tag{18}$$

or

$$-T_l \times (n - 1) - c \leq x < -2T_l \times (n - 1) - 2c. \tag{19}$$

By Definition 1 and Equation (1) and (5), it is easy to observe that pseudo-boundary pixels are non-boundary pixels that can become boundary pixels during the embedding process. When a boundary pixel is encountered during the extracting process, it is originally either a boundary pixel or a pseudo-boundary pixel. Therefore, to find original boundary pixels, we only need to tell whether boundary pixels in the stego-image are genuine or pseudo. The boundary map is the very judge to distinguish between genuine and pseudo. It is a binary array with its every element corresponding to a boundary pixel in the stego-image, 0 for genuine and 1 for pseudo.

The building of a boundary map is simple. Without loss of generality, we consider the cover-image is processed in raster scan order. Once a boundary pixel is encountered, we leave it unchanged and append a '0' into the boundary map. Or else, we process it with the generalized expansion through either expansion or histogram shift. If the stego-pixel is a boundary pixel, we put a '1' into the boundary map. The usage of a boundary map is also intuitive. We process the stego-image in raster scan order. Once a boundary pixel is encountered, we withdraw a bit from the boundary map and if it is '0', we leave it unchanged. Otherwise, we apply the reverse of the generalized expansion to it.

Here, an example is presented to illustrate the idea of the boundary map. For simplicity, we set $T_l = T_r = 0$, $n = 2$ and $c = 0$ for the generalized expansion. From Definition 1 and Definition 2, it is easy to see that only pixels valued 255 are boundary pixels, and only pixels valued 254 are pseudo-boundary pixels. A cover-image is exemplified in Fig. 3. Consider the cover-image in raster scan order. Two '0' bits are first appended to the boundary map since the first two pixels are boundary pixels. Then one '1' bit is appended to the boundary map since the 3rd pixel 254 becomes a boundary pixel during the embedding process. Another '0' bit is appended to indicate the 7th pixel to be a boundary pixel. At last, one '1' is appended since the 8th pixel is changed from 254 to 255.

In the extracting process, it can be known from the stego-image that the length of the boundary map is 5 because there are 5 boundary pixels therein. Based on this fact, we retrieve the boundary map from overhead data, for example extract the first 5 bits of the overhead to form the boundary map. Then we can know that the 1st, 2nd and 4th boundary pixels are genuine whereas the 3rd and 5th boundary pixels are pseudo. By applying the inverse of the generalized expansion to all but the genuine boundary pixels, the cover-image can be recovered exactly.

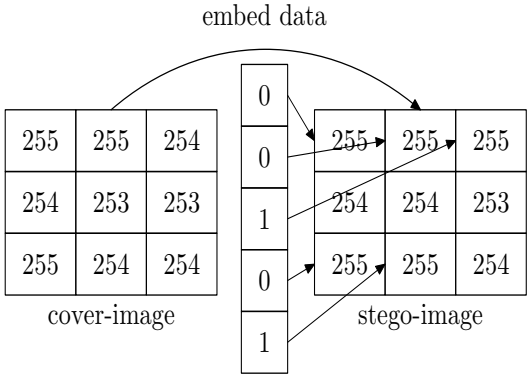


Fig. 3 The Boundary Map. '0' is appended to the boundary map when a boundary pixel is scaned. '1' is appended to the boundary map when a pseudo-boundary pixel becomes a boundary pixel in the data embedding process.

3 Implementation Details

3.1 Capacity Control

The proposed scheme provides three means to control the capacity in different granularity from coarse to fine. Coarse-grained control is achieved by changing the radix of secret data (n in (1)); median-grained control is achieved by tuning the thresholds of prediction-errors (T_l and T_r); and fine-grained control is achieved by embedding an end-token to indicate where the data hiding ends in the image.

The radix (n) is determined first by estimating the desired pure capacity. Suppose the desired pure payload is u in bpp, then the radix n should be no less than $\lceil 2^u \rceil$. The thresholds (T_l and T_r) are determined in an iterative manner. Since the histogram of prediction-errors is in the shape of zero-mean Laplace distribution, we set $c = 0$ and start from $T_l = T_r = 0$. Let $\mathcal{P}$, the number of prediction-errors within $[T_l, T_r]$, be the estimate of the overall capacity, and $\mathcal{C}$, the number of boundary and pseudo-boundary pixel, be the estimate of the overhead cost. We interleavingly increase T_r and decrease T_l by 1 until $\mathcal{P} - \mathcal{C} > \mathcal{U}$ where $\mathcal{U}$ is the desired pure payload in bit. During the determination of the thresholds, we record the distribution of prediction-errors around zero, and adjust thresholds base on it. An example is presented in Fig. 4 to illustrate the process, where overhead is not included as a simplification.

The radix and the thresholds are determined in a way that desired pure payload can at least be satisfied. However, they cannot achieve control in fine granularity to ensure the size of embedded pure payload is exact the size desired and no data unnecessarily is embedded. This fine-grained control is achieved through another strategy: to embed an end-token identifying the end of the embedding. After the radix and the thresholds are determined, the embedding begins from the first non-marginal pixel. As the cover-image is processed pixel by pixel in raster scan order, we keep the pure payload achieved so far updated. Once the achieved pure payload reaches the desired amount, we stop the embedding and record the stop position as the end-token.

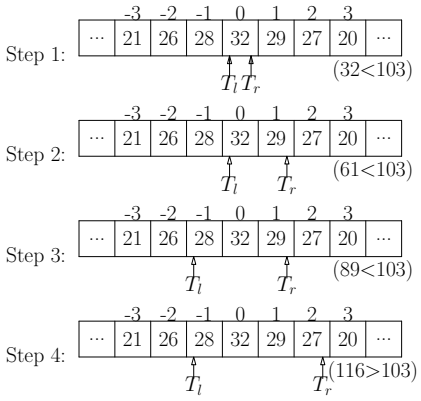


Fig. 4 Capacity control. The array records the distribution of prediction-errors around 0, and the desired payload is 103.

3.2 Overhead Data

The overhead data of the proposed scheme comprises two parts. The first part, of a varying length, records the boundary map. To record the boundary map as overhead data, a preprocess is needed to work out the boundary map in advance, otherwise the length and contents of the boundary map will remain unknown until the end of the embedding process. The preprocess is to apply the generalized expansion to pseudo-boundary pixels, because only pseudo-boundary pixels will introduce new boundary pixels. The second part, of a fixed length, records the values of the adopted parameters and some ancillary information. This contains totally 72 bits including 16 bits for an identifier to avoid false alarm, 4 bits ($2 \leq n \leq 16$) for the radix, 8 bits ($-7 \leq c \leq 8$) for the constant, 8 ($-15 \leq T_l \leq 0 \leq T_r \leq 15$) bits for the thresholds, 18 bits ($2^{18} = 512^2$) for the end-token, 18 bits ($2^{18} = 512^2$) for the length of the boundary map. Actually, the length of the boundary map is dispensable, since it can be obtained by counting the boundary pixels in the stego-image (excluding marginal pixels). However, it is still included because this will bring some extra functions. For example, false alarm can be averted further by matching it with the length of the boundary map inferred from the stego-image.

The overhead data should be recorded as the head of the embedded data, so that it can be obtained immediately once the extracting begins. However, the second part of the overhead data contains parameters of the generalized expansion, which should be know beforehand to initialize the extracting process. A special manipulation is needed to deal with this paradox situation. Note in Fig. 2 that there are some marginal pixels not used for data hiding. We can utilize these pixels to accommodate the second part of the overhead data using LSB replacement, and record the 72 bits placed out as overhead data. This manipulation causes no problem as long as we only consider the last 72 marginal pixels for LSB replacement and recovered them once their original LSB bits are extracted. This idea is verbosely covered in [5].

3.3 Embedding Process

To prepare for the embedding process, all prediction-errors are calculated and the parameters including c, n, T_l, T_r are determined according to the desired pure capacity. Then the cover-image is scanned twice to finish the embedding. The first pass performs the preprocess to work out the boundary map, and the second pass embeds the overhead data together with the secret data into the cover-image. The embedding process is enumerated as follows.

1. Calculate all prediction-errors using the predictor with context modeling as discussed in Section 2.1. Determine the parameters of the proposed generalized expansion as discussed in Section 3.1.
2. Undertake the preprocessing to work out boundary map and the end-token: Scan the cover-image from the first non-marginal pixel. Put a '0' into the boundary map $\mathcal{B}$ if the current pixel is a boundary pixel. Otherwise check whether the corresponding prediction-error is within $[T_l, T_r]$. If yes, increase the capacity which can be achieved so far (denoted with la in bits) by 1. Then check whether the pixel is a pseudo-boundary pixel. If yes, apply the generalized expansion and put a '1' into the boundary map $\mathcal{B}$ if the pseudo-boundary pixel becomes a boundary pixel. Continue until $la = 72 + ||\mathcal{B}|| + ||\mathcal{U}||$. Record the end position as the end-token.
3. Assemble the overhead data: Take the boundary map $\mathcal{B}$ as the first part of the overhead data. Exact the LSB bits of the last 72 marginal pixels as the second part of the overhead data. Concatenate them to form the overhead data.
4. Embed the overhead data and the remaining secret data: Remove from $\mathcal{U}$ the data already been embedded in Step (2), and concatenate the overhead data and the remaining secret data to form a new data stream. Apply the generalized expansion to the cover-image (excluding all boundary and pseudo-boundary pixels) till all the new data stream is embedded.
5. Embed the 72 bits of the identifier, the radix, the thresholds, the constant, the end-token, and the length of the boundary map into the last 72 marginal pixels using LSB replacement.

In Step (2), operator $||\cdot||$ returns the length in bit, and $la = 72+||\mathcal{B}||+||\mathcal{U}||$ denotes the end the embedding, because la is the overall embedding capacity, $72 + ||\mathcal{B}||$ is the cost of the overhead data, and $||\mathcal{U}||$ is the desired pure payload. Actually, it should also be guaranteed that $la \geq ||\mathcal{B}||$ in Step (2), or else the boundary map will be not available when it is needed to tell between genuine boundary pixels and pseudo-boundary pixels in the extracting process. However, this always holds since the boundary map is tiny and negligible to the pure capacity, which is also verified in our experiments. And if this occurs when it happens to exist some boundary pixels in the very beginning, we can avoid this problem by intentionally left some beginning pixels untouched, or change the scan order of the embedding process.

3.4 Extracting Process

In the extracting process, it is important to recover the boundary map first, and then the 72 marginal pixels, and at last the pure payload. It is worth to notice that it is not necessary to fully reconstruct the whole boundary map before it is used. For example, the first boundary pixel in the stego-image can be examined by the boundary map

as long as the first bit of the boundary map is already reconstructed. The extracting process scan the image only once, and it is enumerated as follows.

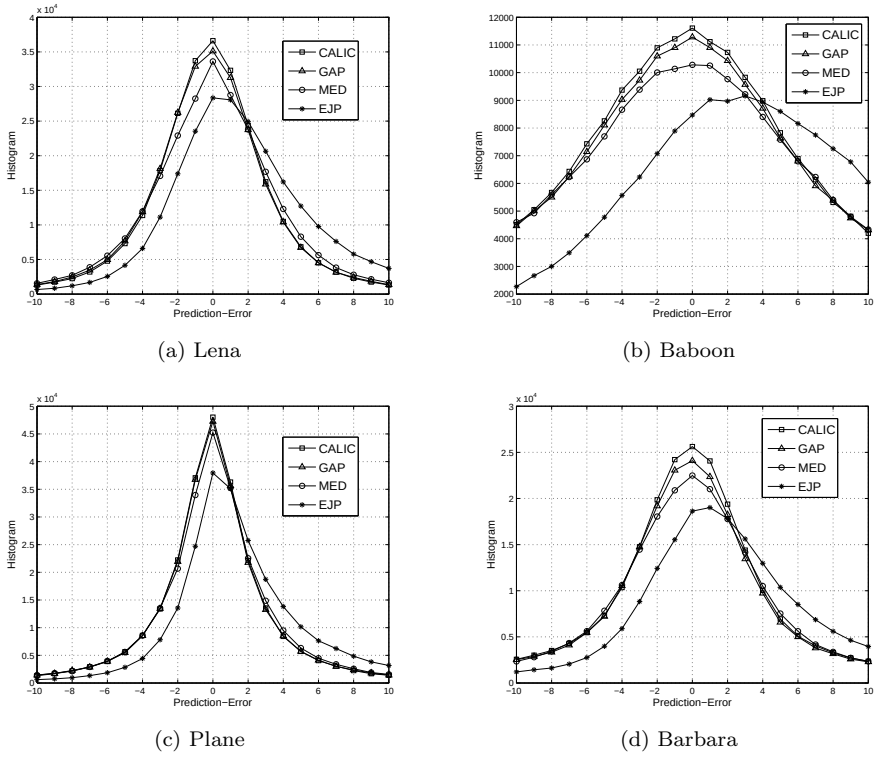
1. Examine whether the image is a stego-image of the proposed scheme: Extract the LSB bits of the last 72 marginal pixels from the image. Decode the 72 bits and check whether the identifier matches with the one of the proposed scheme. Count the number of boundary pixels before the end-token within the image and check whether it is equal to the length of the boundary map recorded in the 72 bits. If both tests succeed, we take the image as a stego-image.
2. Extract data in stego-image: Scan the image from the first non-marginal pixel to the end-token. If the pixel is not a boundary pixel, apply the inverse of the generalized expansion to its prediction-error and extract the data. Put the extracted data into the boundary map $\mathcal{B}$ if it is not completely reconstructed yet. Or use the extracted data to restore the LSB bits of the last 72 marginal pixels if they are not recovered yet. Otherwise, save the extracted data as the pure payload. If the pixel is a boundary pixel, check the boundary map to see whether it is genuine or pseudo. If it is genuine, leave it unchanged, and if it is pseudo, process it in the same way of processing non-boundary pixel.

4 Experimental Results

In this section, we study the proposed reversible data hiding scheme via experiments. The scheme is implemented and tested using MATLAB. All test images are 512×512 8-bit grayscale standard images. Image qualities are measured with PSNR, and the embedding capacities are measured with bpp (bit per pixel).

4.1 Testing of Prediction With Context Modeling

Working as an extra procedure based on normal prediction, the context modeling refines prediction-errors through an error feedback mechanism. The advantage of prediction with context modeling overall piece-wise linear prediction was well developed in [20] when used in lossless image compression. In Fig. 5, histograms of the prediction-errors produced by CALIC, GAP, MED and EJP (an enhanced predictor for JPEG-LS) [12] are compared. Only prediction-errors between -10 and 10 are considered here, because these prediction-errors are the great majority of the total (65% to 95%) and they are small ones with priority in reversible data hiding. For all the four images including Lena, Baboon, Plane and Barbara, the CALIC predictor performs better than others with more prediction-errors are concentrated near the center 0. The ascendancy of the CALIC predictor is most apparent for the complex image Baboon, for which the histogram is significantly concentrated and promoted. As to the smooth image Plane, the improvement brought by the context modeling is minor. This is explainable since smoothness implies there are less high-order structures like edges and textures, so redundancy is chiefly within small local area which is the dominating region of prediction instead of context modeling.



(a) Lena (b) Baboon

(c) Plane (d) Barbara

Fig. 5 Histograms of Prediction-errors on test image Lena, Baboon, Plane and Barbara. CALIC predictor is compared with GAP, MED and EJP predictors.

4.2 Testing of Boundary Map

The technique of recording overhead data plays a key role in determining the performance of reversible data hiding schemes. In the first DE scheme [24], the size of the overhead data is considerable large because of the location map, which can be even larger than the overall capacity at low embedding level. Since then, many efforts [4, 10, 13, 14, 19, 23] have been made to condense or eliminate the location map. Thodi et al. [23] also proposed a scheme without location map by using a histogram shift and their scheme can achieve a capacity of 1 bpp. In their scheme, the location map is replaced with a much more compressible overflow map, which records only overflow-potential pixels. Thodi et al.'s scheme was further improved by Hu et al., who proposed a scheme with an improved overflow map, which is shorter than the original one.

The boundary map proposed in this paper is short and needs no compression. The boundary maps of the most popular 12 standard test images are worked out when different thresholds are set. The sizes of the boundary maps are tabulated in Table 1, where $n = 2$, $c = 0$, and $T = -T_l = T_r$. For images including Lena, Baboon, Man, Goldhill, Sailboat and Milkdrop, the sizes of the boundary maps are negligible. While for other four images, the largest size is merely 2574 bits.

Table 1 Sizes of the Boundary Maps (bits). The boundary maps are worked out when different thresholds are set $T = -T_l = T_r$.

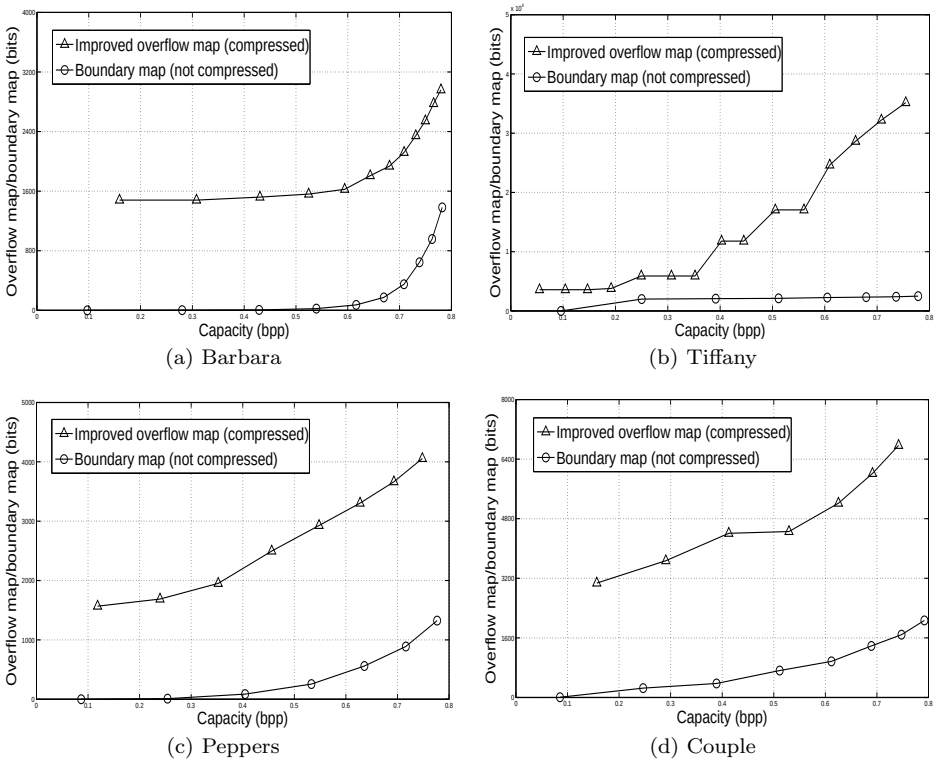
Predictor	T=0	T=1	T=2	T=3	T=4	T=5	T=6	T=7	T=8
Lena	0	0	1	2	3	3	4	5	5
Baboon	0	0	0	0	0	0	0	0	0
Plane	0	8	32	74	119	185	250	338	431
Barbara	0	2	5	24	74	174	365	637	978
Boat	2	20	36	54	86	133	188	274	429
Tiffany	0	2000	2068	2127	2256	2328	2392	2494	2574
Man	0	0	0	0	0	0	1	1	2
Goldhill	0	0	0	0	0	0	0	0	0
Peppers	0	10	86	259	570	893	1328	1947	2545
Sailboat	0	0	0	0	0	0	2	15	32
Couple	5	250	374	721	972	1396	1685	2065	2406
Milkdrop	0	0	0	0	0	0	9	58	143

To evaluate the boundary map, we compare it with the improved overflow map [10], which represents the latest and the best technique of recording overhead data. In the comparison, we only consider the four images having large boundary maps in Table 1. The results are shown in Fig. 6, where the curves of embedding capacity versus uncompressed boundary map are compared with the curves of embedding capacity versus compressed overflow map. It is apparent that, even not compressed, the boundary maps are much shorter than the improved overflow maps for all the four images.

4.3 Testing of Low Level Embedding

So far, we have studied the performances of the prediction with context modeling and the boundary map when other influencing factors are excluded. Now, we test the proposed reversible data hiding scheme within real applications where all factors are taken into account. Here we only consider low level embedding when capacities are at most 1 bpp, namely when the radix n is 2. The constant c is set 0, and a single threshold ($T = -T_l = T_r$) is used to control the capacity.

The embedding capacity versus image quality curves are shown in Fig. 7. The performances of four other schemes are also included in Fig. 7 for comparison. P3 is the algorithm P3 proposed by Thodi et al., which is the best one among the six algorithms in [23]. K1 is the algorithm proposed in [14], and K2 is the algorithm proposed in [13]. H1 is the algorithm proposed by Hu et al. in [10], which is the one with the improved overflow map. The performance of the proposed scheme is better than the performance of the other four schemes for all the tested images. The ascendancy of the proposed scheme is most apparent for the complex image Baboon and is least apparent for the smooth image Plane. This accords with the experimental results of the CALIC prediction. The reason is also similar: context modeling takes effect when complex high-order structures are present in images, while prediction performs well when images are smooth. We note that the curves of the PEE schemes (P3, H1 and ours) drop down drastically when the limit (1 bpp) is approached. Actually, the proposed scheme balks at gain capacity when the threshold T becomes as large as 10, and at the same time, the image quality slumps. This is also the reason why only small thresholds are considered in Section 4.1 and 4.2.



(a) Barbara (b) Tiffany

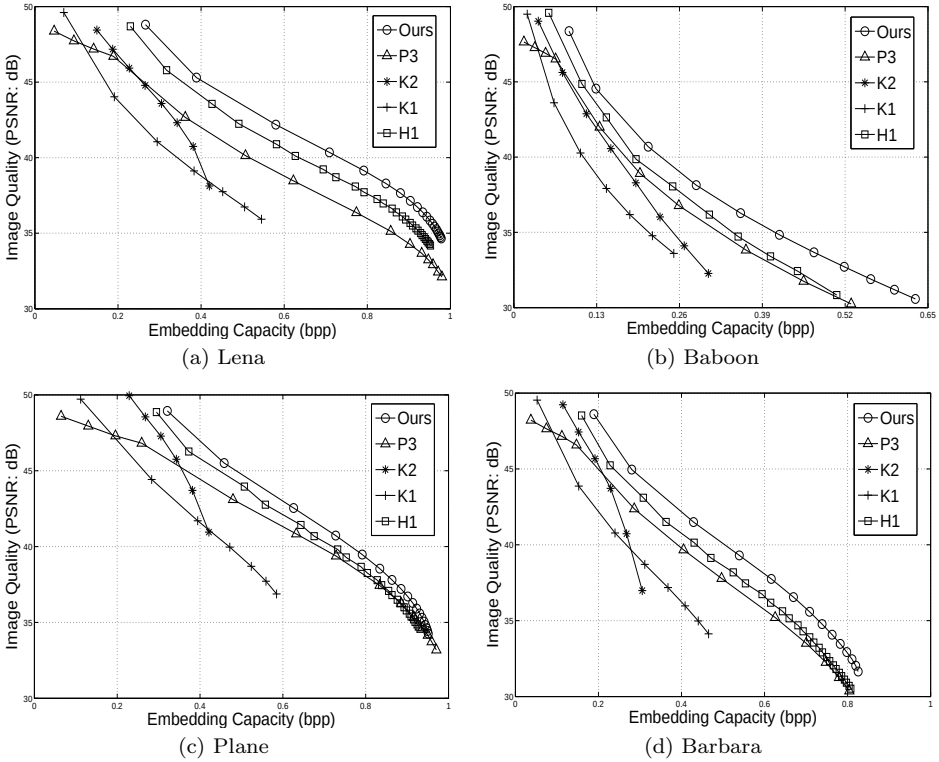
(c) Peppers (d) Couple

Fig. 6 Comparison between the Boundary Map and the Improved Overflow Map. The boundary maps are not compressed while the improved overflow maps are compressed using JBIG.

4.4 Testing of High Level Embedding

One drawback of previous PEE schemes is that the embedding capacity is limited to 1 bpp in single embedding. Although larger capacity can be achieved by multiple embedding, it is time-consuming to undertake the embedding recursively. However, with the generalized expansion, capacity larger than 1 bpp can be achieved in single embedding. This is done by setting the radix n to be larger than 2.

The performance of high level embedding of the proposed scheme are presented in Table 2. We see that the capacity can be larger than 1 bpp when n is larger than 2. When the same threshold is set, the capacity of N3 and N4 are $\log_2 3 = 1.58$ and $\log_2 4 = 2$ times of the capacity of N2. The high level embedding of the proposed scheme is compared with other schemes which can also achieve capacity larger than 1 bpp. The comparison on Lena is given in Fig. 8, where three other schemes (L1, L2, H2) are considered. L1, L2, and H2 are the schemes proposed in [18], [17], and [9] respectively. Schemes L1 and L2 achieve large capacities through multiple embedding, while H2 accomplish this in single embedding. In Fig. 8, N2 performs better than all the others when the capacity is lower than 1 bpp. However, the capacity of N2 cannot be higher than 1 bpp, and it is N3 that performs the best when the capacity is between



(a) Lena (b) Baboon

(c) Plane (d) Barbara

Fig. 7 Performance Comparison on Test Images Lena, Baboon, Plane and Barbara. P3, K2, K1 and H1 are the schemes proposed in [23], [13], [14] and [10].

1 bpp and 1.5 bpp. Once its limit is approached, N3 also lags behind, and N4 in turn stands out to be the best one when the capacity is higher than 1.5 bpp.

5 Conclusions

This paper proposes a reversible data hiding scheme with high capacity-distortion efficiency. A generalized expansion is proposed to embed secret data into prediction-errors calculated by a predictor equipped with a context modeling tool, and an efficient method of recording overhead data, named boundary map, is proposed to cut overhead data significantly. The proposed scheme demonstrates very competitive performance in embedding capacity and image quality, and it can provide capacities larger than 1 bpp without resorting to multiple embedding.

The generalized expansion embodies features of the two categories of reversible data hiding using different expansion (DE) and histogram operation (HO). Its data hiding strategy is an improvement to difference expansion, with which large capacity even larger than 1 bpp becomes achievable in single embedding. It also utilizes a histogram shift to solve the ambiguities between expanded prediction-errors and non-expanded ones, so that there is no more need to record the locations of expanded prediction-errors with a location map.

Table 2 Embedding capacity (bpp) and image quality (PSNR) when the radix n assumes different values. N2, N3 and N4 are the cases when n assumes 2, 3 and 4. The capacities are pure capacities containing no overhead data.

Image	T=0		T=2		T=4		T=6		T=8	
	bpp	PSNR	bpp	PSNR	bpp	PSNR	bpp	PSNR	bpp	PSNR
Lena (N2)	0.14	51.28	0.58	42.19	0.79	39.15	0.88	37.65	0.92	36.73
Lena (N3)	0.22	45.37	0.92	36.22	1.25	33.16	1.39	31.66	1.46	30.73
Lena (N4)	0.28	41.88	1.16	32.72	1.58	29.65	1.76	28.15	1.84	27.23
Plane (N2)	0.18	51.20	0.63	42.54	0.79	39.48	0.86	37.81	0.90	36.71
Plane (N3)	0.29	45.32	0.99	36.59	1.25	33.50	1.36	31.82	1.42	30.73
Plane (N4)	0.36	41.85	1.25	33.09	1.57	30.00	1.71	28.33	1.77	27.25
Baboon (N2)	0.04	51.25	0.21	40.69	0.36	36.28	0.47	33.67	0.56	31.90
Baboon (N3)	0.07	45.26	0.33	34.68	0.56	30.27	0.75	27.66	0.89	25.88
Baboon (N4)	0.09	41.75	0.42	31.17	0.71	26.75	0.94	24.14	1.11	22.36
Barbara (N2)	0.10	51.25	0.43	41.50	0.62	37.75	0.71	35.59	0.77	34.08
Barbara (N3)	0.15	45.30	0.68	35.51	0.98	31.75	1.12	29.60	1.18	28.11
Barbara (N4)	0.19	41.80	0.86	32.00	1.23	28.26	1.35	26.15	1.39	24.71

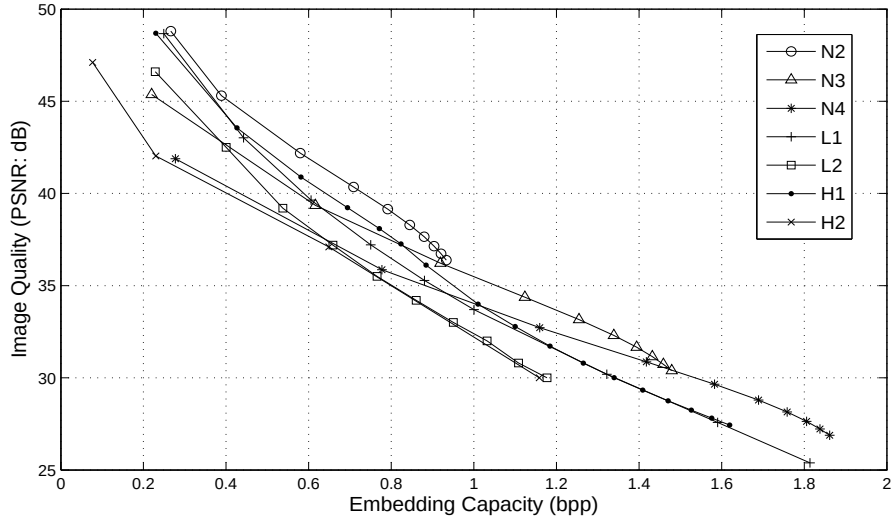


Fig. 8 Performance Comparison of High Level Embedding on Lena. N2, N3 and N4 are the cases when n assumes 2, 3 and 4 of the proposed scheme. L1, L2, H1, and H2 are the schemes proposed in [18], [17], [10] and [9].

Predictor-errors generated by the predictor with context modeling is smaller and beneficial to reversible data hiding. The context modeling tool, which acts as an extra procedure on prediction, refines prediction-errors through an error feedback mechanism by further exploiting redundancy within high-order structures like edges and textures. With the context modeling, distribution of prediction-errors becomes more concentrated around zero, which benefits both the embedding capacity and the image quality of the proposed scheme because smaller prediction-errors are more expandable and bring about less distortion.

To prevent overflows, overflow-potential pixels are not expanded in the proposed scheme, and a novel boundary map, instead of an overflow map, is proposed to recorded those overflow-potential pixels and differentiate them from other ambiguous pixels. The

boundary map is very economical in overhead cost. Therefore, only a negligible portion of the payload is needed to serve as overhead data, which significantly enlarges the pure capacity of the proposed reversible data hiding scheme. Moreover, being a binary array, the boundary map requires no compression, which is a relief to the overall scheme despite the incorporation of the context modeling.

Acknowledgements This work is supported by the Fundamental Research Funds for the Central Universities, and Development Program of China and Beihang University Graduate Innovation Fund. The authors would like to thank the anonymous reviewers for their valuable comments to improve the quality of the paper.

References

1. Alattar, A.M.: Reversible watermark using the difference expansion of a generalized integer transform. IEEE Trans. Image Processing **3**(8), 1147–1156 (2004)
2. Barton, J.M.: Method and apparatus for embedding authentication information within digital data. U.S. Patent 5 646 997 (1997)
3. Chang, C.C., Lu, T.C.: A difference expansion oriented data hiding scheme for restoring the original host images. Journal of Systems and Software **79**(12), 1754–1766 (2006)
4. Chang, Z., Kou, W., Xu, J.: More compressible location map for reversible watermarking using expansion embedding. Electron. Lett. **43**, 1353–1354 (2007)
5. Chen, M., Chen, Z., Zeng, X., Xiong, Z.: Reversible data hiding using additive prediction-error expansion. In: ACM Multimedia and Security 09, pp. 19–24. Princeton, New Jersey (2009)
6. Chen, M., Chen, Z., Zeng, X., Xiong, Z.: Reversible image watermarking based on full context prediction. In: ICIP 2009, pp. 4253–4256. Cairo, Egypt (2009)
7. Chung, K.L., Huang, Y.H., Yang, W.N., Hsu, Y.C., Chen, C.H.: Capacity maximization for reversible data hiding based on dynamic programming approach. Applied Mathematics and Computation **208**(1), 284–292 (2009)
8. Fridrich, J., Goljan, M., Du, R.: Lossless data embedding - new paradigm in digital watermarking. EURASIP Journal on Applied Signal Processing **2002**, 185–196 (2002)
9. Hsiao, J.Y., Chan, K.F., Chang, J.M.: Block-based reversible data embedding. Signal Processing **89**, 556569 (2009)
10. Hu, Y., Lee, H.K., Li, J.: DE-based reversible data hiding with improved overflow location map. IEEE Trans. Circuits and Systems for Video Technology **19**(2), 250–260 (2009)
11. Hwang, J., Kim, J., Choi, J.: Reversible image watermarking based on histogram modification. In: IWDW 2006, LNCS 4283, pp. 348–361. Jeju Island, Korea (2006)
12. Jiang, J., Guo, B., Yang, S.Y.: Revisiting the JPEG-LS prediction scheme. In: Vision, Image and Signal Processing, pp. 575–580 (2000)
13. Kim, H.J., Sachnev, V., Shi, Y.Q., Nam, J., Choo, H.G.: A novel difference expansion transform for reversible data embedding. IEEE Trans. Information Forensic and Security **3**(3), 456–465 (2008)
14. Kim, K.S., Lee, M.J., Lee, H.K., Suh, Y.H.: Histogram-based reversible data hiding technique using subsampling. In: ACM Multimedia and Security 08, pp. 69–75. Oxford, UK (2008)
15. Kuribayashi, M., Morii, M., Tanaka, H.: Reversible watermark with large capacity based on the prediction error expansion. IEICE Trans. Fundamentals **E91**(7), 1780–1790 (2008)
16. Lee, C.C., Wu, H.C., T., C.S., Chu, Y.P.: Adaptive lossless steganographic scheme with centralized difference expansion. Pattern Recognition **41**(6), 2097–2106 (2008)
17. Lin, C.C., Hsueh, N.L.: A lossless data hiding scheme based on three-pixel block differences. Pattern Recognition **41**(4), 1415–1425 (2008)
18. Lin, C.C., Tai, W.L., Chang, C.C.: Multilevel reversible data hiding based on histogram modification of difference images. Pattern Recognition **41**, 3582–3591 (2008)
19. Lin, C.C., Yang, S.P., Hsueh, N.L.: Lossless data hiding based on difference expansion without a location map. In: 2008 Congress on Image and Signal Processing, pp. 8–12 (2008)
20. Memon, N., Wu, X.: Recent developments in context-based predictive techniques for lossless image compression. The Computer Journal **40**(2), 127–136 (1997)

21. Ni, Z., Shi, Y.Q., Ansari, N., Su, W.: Reversible data hiding. IEEE Trans. Circuits and Systems for Video Technology **16**(3), 354–362 (2006)
22. Thodi, D.M., Rodriguez, J.J.: Prediction-error based reversible watermarking. In: ICIP 2004, pp. 1549–1552. Singapore (2004)
23. Thodi, D.M., Rodriguez, J.J.: Expansion embedding techniques for reversible watermarking. IEEE Trans. Image Processing **16**(3), 721–730 (2007)
24. Tian, J.: Reversible data embedding using a difference expansion. IEEE Trans. Circuits and Systems for Video Technology **13**(8), 890–896 (2003)
25. Tsai, P., Hu, Y.C., Yeh, H.L.: Reversible image hiding scheme using predictive coding and histogram shifting. Signal Processing **89**, 1129–1143 (2009)
26. Tseng, H.W., Chang, C.C.: An extended difference expansion algorithm for reversible watermarking. Image and Vision Computing **26**(8), 1148–1153 (2008)
27. Vasiliy, S., Kim, H.J., Xiang, S., Nam, J.: An improved reversible difference expansion watermarking algorithm. In: IWDW 2007, LNCS 5041, pp. 254–263. Guangzhou, China (2008)
28. Weinberger, M.J., Seroussi, G., Sapiro, G.: The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS. IEEE Trans. Image Processing **9**(8), 1309–1324 (2000)
29. Wu, X., Memon, N.: Context-based, adaptive, lossless image coding. IEEE Trans. Communication **45**(4), 437–444 (1997)