Recursive Histogram Modification: Establishing Equivalency Between Reversible Data Hiding and Lossless Data Compression

Weiming Zhang, Xiaocheng Hu, Xiaolong Li, and Nenghai Yu

***Abstract*— State-of-the-art schemes for reversible data hiding (RDH) usually consist of two steps: first construct a host sequence with a sharp histogram via prediction errors, and then embed messages by modifying the histogram with methods, such as difference expansion and histogram shift. In this paper, we focus on the second stage, and propose a histogram modification method for RDH, which embeds the message by recursively utilizing the decompression and compression processes of an entropy coder. We prove that, for independent identically distributed (i.i.d.) gray-scale host signals, the proposed method asymptotically approaches the rate-distortion bound of RDH as long as perfect compression can be realized, i.e., the entropy coder can approach entropy. Therefore, this method establishes the equivalency between reversible data hiding and lossless data compression. Experiments show that this coding method can be used to improve the performance of previous RDH schemes and the improvements are more significant for larger images.**

***Index Terms*— Reversible data hiding, histogram shift, difference expansion, recursive code construction, rate-distortion.**

I. Introduction

AS A TECHNIQUE that embeds the secret message into cover signals, information hiding has been widely applied in areas such as covert communication, media annotation and integrity authentication. Reversible data hiding (RDH) is one kind of information hiding techniques with the characteristics such that not only the secret message needs to be precisely extracted, but also the cover itself should be restored losslessly. This property is important in some special scenarios such as medical imagery [1], military imagery and law forensics. In these applications, the cover is too precious or too important to be damaged [2]. Moreover, it has been found that RDH can be quite helpful in video error-concealment coding [3].

Manuscript received October 19, 2012; revised February 13, 2013; accepted March 31, 2013. Date of publication April 12, 2013; date of current version May 22, 2013. This work was supported in part by the Natural Science Foundation of China under Grants 61170234 and 60803155, and by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA06030601. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Vladimir Stankovic.

W. Zhang, X. Hu, and N. Yu are with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, China (e-mail: weimingzhang@yahoo.cn; hxc@mail.ustc.edu.cn; ynh@ustc.edu.cn).

X. Li is with the Institute of Computer Science and Technology, Peking University, Beijing 100871, China (e-mail: lixiaolong@icst.pku.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2013.2257814

A plenty of RDH algorithms have been proposed in the past decade. Classical RDH algorithms roughly fall into three categories. The first class of algorithms follow the idea of compression-embedding framework, which was first introduced by Fridrich [4]. In these algorithms, a compressible two-value feature is calculated from the cover, and the message can be embedded in the extra space left by lossless compression. The second class of methods are based on difference expansion (DE) [5], 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 the message. Another kind of RDH is based on histogram shift (HS) [6]. The histogram of one special feature (e.g., grayscale value) for natural image is quite uneven, so some space can be saved for watermarks by shifting the bins of histogram. In fact, better performance can be achieved by applying DE or HS to the residual part of images, e.g., the prediction errors (PE) [7]–[12]. Recently, some improved method for RDH in PE have been presented. Li *et al*. [13] proposed to adaptively embed 2 bits into each expandable pixel of flat regions and 1 bit into that of rough regions. Coltuc [14] proposed to change the value of PE by modifying not only the predicted pixel but also its prediction context.

Almost all recent RDH 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 the sorting technique [11] or pixel selection [13]. The second step reversibly embeds the message into the host sequence by modifying its histogram with methods like HS and DE. These are for the sake of maximizing the payload for a given distortion constraint or minimizing the overall distortion for a given payload. Many PE techniques have been applied to RDH, such as JPEG-LS prediction errors [7], rhombus prediction errors [11], and interpolation errors [12]. The sorting technique [11] and pixel selection [13] give priority to prediction errors in smooth regions, so a sharper histogram can be obtained. After generating a good histogram for RDH, the following two problems are: 1) what is the maximum embedding rate for the given histogram and distortion constraint; 2) how to realize the optimal modification on the histogram for achieving the maximum embedding rate? Herein, embedding rate is defined as the average number of message bits carried by one host signal.

For independent and identically distributed (i.i.d.) host signals, the first problem has been solved by Kalker and Willems [15], who formulated the RDH as a rate-distortion problem, and obtained the rate-distortion function, i.e., the upper bound of the embedding rate under a given distortion constraint, as follows:

$$\rho_{rev}(\Delta) = maximize\{H(Y)\} - H(X) \tag{1}$$

where X and Y denote the random variables of host signal and stego signal respectively. Maximizing entropy is over all transition probabilities $P_{Y|X}(y|x)$ satisfying the distortion constraint

$$\sum_{x,y} P_X(x) P_{Y|X}(y|x) D(x, y) \leq \Delta \tag{2}$$

where the distortion metric $D(x, y)$ is usually defined as the square error distortion, i.e., $D(x, y) = (x - y)^2$.

In fact, the optimal solution $P_{Y|X}(y|x)$ for (1) implies the optimal modification manner on the histogram of the host signal X. However, how to efficiently realize the optimal modification remains a problem. For a binary host sequence, i.e., $x \in \{0, 1\}$, Kalker and Willems [15] proposed a recursive code construction and Zhang *et al*. [16], [17] improved the recursive code construction to approach the rate-distortion bound (1). Recently, Lin *et al*. [18] proposed a code construction for gray-scale signals, i.e., $x \in \{0, 1, \ldots, B-1\}$ where B is an integer larger than 1. Although Lin *et al*.'s method is close to the rate-distortion bound (1), it modifies the host sequence in a signal by signal manner, which suffers from high complexity of implementation and is not proved to approach the rate-distortion bound.

In this paper, we extend the recursive code construction from binary signals to gray-scale signals, which modifies the histogram in a bin by bin manner according to the optimal transition probability. This novel code can be easily implemented by recursively applying the decompression process and compression process of an entropy coder. We prove that the proposed code can approach the rate-distortion bound (1) as long as the entropy coder reaches entropy. In other words, the proposed code is optimal for RDH if the entropy coder is optimal for lossless data compression (LDC), which essentially establishes the equivalency between RDH and LDC. Furthermore, to show the power of the proposed code, we improve Luo *et al*.'s method [12] and Sachnev *et al*.'s method [11] by applying the code to the sequence of PEs.

The rest of the paper is organized as follows. Section II briefly introduces how to solve the optimal distribution for the rate-distortion problem (1). The recursive code construction for gray-scale signals with optimality proof is elaborated in Section III. Furthermore, two application cases are given in Section IV by exploiting the code construction to improve the RDH schemes presented in [11], [12], and finally we conclude this paper in Section V.

II. Optimal Transition Probability

Throughout this paper, we denote matrices and vectors by boldface fonts, and use capital letters for the random variables and small letters for their realizations. We denote the entropy by $H(X)$ and the conditional entropy by $H(Y|X)$. Specially, for the probability distribution $(P(0), \ldots, P(B-1))$ such that $P(i) > 0$ and $\Sigma_{i=0}^{B-1} P(i) = 1$, the B-ary entropy function is defined as $H(P(0), \ldots, P(B-1)) = -\Sigma_{i=0}^{B-1} P(i) \log_2 P(i)$.

To estimate the rate-distortion bound or construct codes approaching the bound, we should first estimate the optimal solution $P_{Y|X}(y|x)$ for (1). Due to its convexity, many convex optimization algorithms can be used to solve it, like gradient projection or interior-point methods. However, although interior-point methods take very few iterations to converge, they have difficulty in handling problems with large scale because the complexity of computing the step direction is $O((B \times B)^3)$, if both x and y belong to $\{0, 1, \ldots, B-1\}$.

Fortunately, it has been proved both in [18] and [19] that the optimal channel transition matrix of problem (1) has a Non-Crossing-Edges property. To be specific, given an optimal $P_{Y|X}$, for any two distinct possible transition events $P_{Y|X}(y_1|x_1) > 0$ and $P_{Y|X}(y_2|x_2) > 0$, if $x_1 < x_2$, then $y_1 \leq y_2$ holds. Lin *et al*. [18] pointed out that, by using this Non-Crossing-Edges property, the joint distribution of X and Y can be expressed as $P_{X,Y}(x, y) = \max\{0, \min\{P_{CX}(x), P_{CY}(y)\}\} - \max\{P_{CX}(x-1), P_{CY}(y-1)\}$. Herein, $P_{CX}(x)$ and $P_{CY}(y)$ are cumulative probability distribution of X and Y defined by $P_{CX}(x) = \sum_{i=0}^{x} P_X(i), x = 0, \ldots, B-1$, and $P_{CY}(y) = \sum_{i=0}^{y} P_Y(i), y = 0, \ldots, B-1$. It is noted that $P_{CX}(-1) = P_{CY}(-1) = 0$, and $P_{CX}(B-1) = P_{CY}(B-1) = 1$.

So the problem (1) can be simplified to find the optimal marginal distribution of the stego-signal $P_Y(y)$. Lin *et al*. [18] proposed a Backward and Forward Iterative (BFI) algorithm to estimate $P_Y(y)$. Recently, we proposed a fast algorithm to estimate the optimal marginal distribution of the stego-signal based on lagrangian duality [20].

Problem (1) is about a sender with a distortion constraint. In practice, we may also consider a sender with a given embedding rate R and minimize the average distortion, such that

$$\begin{aligned} &\text{minimize } \textstyle\sum_{x,y} P_X(x) P_{Y|X}(y|x) D(x, y) \\ &\text{subject to } H(Y) = R + H(X). \end{aligned} \tag{3}$$

Problem (3) is dual with problem (1), and the minimization is over all transition probabilities $P_{Y|X}(y|x)$. In [20], we also proposed a fast algorithm to estimate the optimal solution of problem (3). We provided the source codes for solving problems (1) and (3) at the web site [21].

For problem (1) or (3), we denote the optimal transition probability matrix from X to Y by $\mathbf{Q}_{Y|X}$, and the optimal transition probability matrix from Y to X by $\mathbf{Q}_{X|Y}$, such that

$$\mathbf{Q}_{Y|X} = (\mathbf{Q}_{Y|0}, \mathbf{Q}_{Y|1}, \ldots, \mathbf{Q}_{Y|B-1}) \tag{4}$$

$$\mathbf{Q}_{X|Y} = (\mathbf{Q}_{X|0}, \mathbf{Q}_{X|1}, \ldots, \mathbf{Q}_{X|B-1}). \tag{5}$$

The xth column of $\mathbf{Q}_{Y|X}$ is the transition probability distribution under the condition $X = x - 1$ $(1 \leq x \leq B)$, that is,

$$\mathbf{Q}_{Y|x-1} = \big(P_{Y|X}(0|x-1), \ldots, P_{Y|X}(B-1|x-1)\big)^T. \tag{6}$$

The yth column of $\mathbf{Q}_{X|Y}$ is the transition probability distribution under the condition $Y = y - 1$ $(1 \le y \le B)$, that is,

$$\mathbf{Q}_{X|y-1} = \big(P_{X|Y}(0|y-1), \ldots, P_{X|Y}(B-1|y-1)\big)^T \quad (7)$$

III. Optimal Histogram Modification for RDH

In this section, we will present a histogram modification method for RDH to approach the rate-distortion bound (1), which has been motivated by the recursive code construction [15], [17]. We will first present an analysis of the methodology; then some implementation details are discussed; finally, a thorough algorithm diagram is presented.

A. Recursive Histogram Modification

In this section, we propose a recursive histogram modification (RHM) method for RDH, which divides the host sequence into disjoint blocks and embeds the message by recursively modifying the histogram of each block in a bin by bin manner.

Assume that a memoryless source produces the host sequence $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ with the identical distribution $P_X(x)$ such that $x \in \{0, 1, \ldots, B-1\}$. The message is usually encrypted before being embedded, so we assume that the secret message $\mathbf{m} = (m_1, m_2, \ldots)$ is a binary random sequence with $m_i \in \{0, 1\}$. To recursively embed the message, we first divide the host sequence $\mathbf{x}$ into g disjoint blocks, in which the first $g-1$ blocks have the same length K, and the last block has the length L_{last}, and thus $N = K(g-1)+L_{last}$. To finish the embedding, we have to set L_{last} to be larger than K, and we will discuss how to determine L_{last} in Subsection III-C. The ith cover block is denoted by $\mathbf{x}_i$, and the corresponding stego block is denoted by $\mathbf{y}_i$, $i = 1, \ldots, g$.

We embed the message into each block by an embedding function $Emb()$, such that $(\mathbf{M}_{i+1}, \mathbf{y}_i) = Emb(\mathbf{M}_i, \mathbf{x}_i)$, with $i = 1, \ldots, g$ and $\mathbf{M}_1 = \mathbf{m}$. In other words, the embedding process in the ith block outputs the message to be embedded into the $(i+1)$th block. Fig. 1 briefly depicts the data embedding process, in which $\mathbf{M}_{i+1}$ consists of the the rest message bits and the overhead information, $O(\mathbf{x}_i)$, for restoring $\mathbf{x}_i$. The message extraction and cover reconstruction are processed in a backward manner with an extraction function $Ext()$, such that $(\mathbf{M}_i, \mathbf{x}_i) = Ext(\mathbf{M}_{i+1}, \mathbf{y}_i)$, with $i = g, \ldots, 1$.

Now we consider a sender with a distortion constraint Δ. To maximize the embedding rate, we first use the method in [20] to estimate the optimal transition probability matrix $\mathbf{Q}_{Y|X}$ of problem (1) according to Δ and the host distribution P_X, and then we can calculate the transition probability matrix $\mathbf{Q}_{X|Y}$. The embedding and extracting processes will be realized by the decompression and compression algorithms of an entropy coder (e.g., arithmetic coder) with $\mathbf{Q}_{Y|X}$ and $\mathbf{Q}_{X|Y}$ as parameters. We denote the compression and decompression algorithms by $Comp()$ and $Decomp()$ respectively. For simplicity, we assume that perfect compression can be realized, i.e., the entropy coder can reach entropy.

In each host block $\mathbf{x}_i$, the embedding function $Emb()$ executes two tasks. One task is to embed some bits of the message and generate the stego-block $\mathbf{y}_i$ by decompressing the message sequence according to $\mathbf{Q}_{Y|X}$. The other task is to produce the overhead information $O(\mathbf{x}_i)$ for restoring the host block $\mathbf{x}_i$ by compressing it according to $\mathbf{y}_i$ and $\mathbf{Q}_{X|Y}$. The overhead information will be embedded into the next block $\mathbf{x}_{i+1}$ as a part of $\mathbf{M}_{i+1}$ (see Fig. 1).

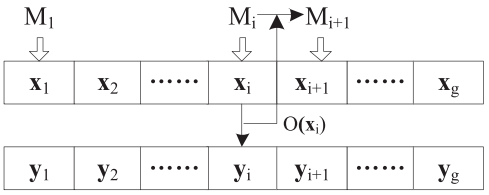


Fig. 1. Recursive blockwise data embedding.

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	
$\mathbf{M}_1$	1	0	0	1	0	1	1	0	0	1	1	0	1	⋯
$\mathbf{x}_1$	0	2	0	1	0	0	0	1	0	0				
$\mathbf{y}'_{1,0}$	0		0		2	0	1		0	0				
$\mathbf{y}'_{1,1}$				1				2						
$\mathbf{y}_1$	0	2	0	1	2	0	1	2	0	0				
$\mathbf{x}'_{1,1}$				1			0							
$\mathbf{x}'_{1,2}$		2			0			1						

$O(\mathbf{x}'_{1,1}) = (1,0)$ $\quad$ $O(\mathbf{x}'_{1,2}) = (0,1,1,0,1)$

$\mathbf{M}_2 = O(\mathbf{x}'_{1,1}) \| O(\mathbf{x}'_{1,2}) \| (0,1,\cdots) = (1,0,0,1,1,0,1,0,1,\cdots)$

Fig. 2. Example of the proposed code construction. Assume that the first eight bits of $\mathbf{M}_1$ are decompressed into $\mathbf{y}'_{1,0}$, and the following three bits of $\mathbf{M}_1$ (bits in the shadow) are decompressed into $\mathbf{y}'_{1,1}$.

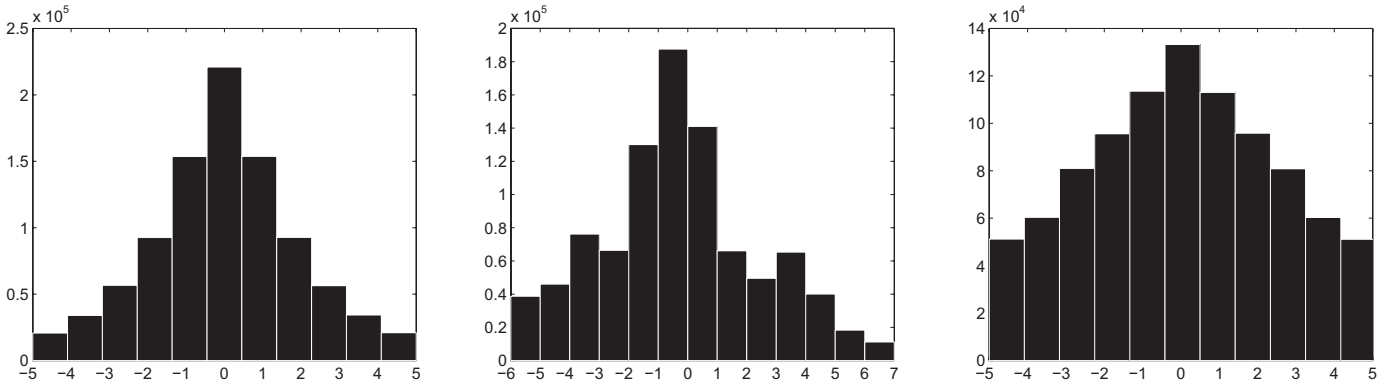
In each stego block $\mathbf{y}_i$, the extraction function $Ext()$ also executes two tasks. One task is to decompress the overhead information extracted from $\mathbf{y}_{i+1}$ according to $\mathbf{Q}_{X|Y}$ and restore the host block $\mathbf{x}_i$. The other task is to extract the message by compressing $\mathbf{y}_i$ according to $\mathbf{x}_i$ and $\mathbf{Q}_{Y|X}$.

Next, we describe the embedding, extracting and restoring processes in details.

1) Data Embedding Process: The embedding is done by substituting signals of the cover with sequences obtained by decompressing the message bits in accordance with the the optimal transition probability matrix $\mathbf{Q}_{Y|X}$. In other words, for each bin x, $x \in \{0, \ldots, B-1\}$, we decompress a part of the message sequence according to the distribution $\mathbf{Q}_{Y|x}$, and then substitute all host signals equal to x with the decompressed sequence. Thus, the histogram of the cover block is modified in a bin by bin manner.

Taking the first block as an example, we describe how to do $(\mathbf{M}_2, \mathbf{y_1})$=$Emb(\mathbf{M}_1, \mathbf{x_1})$. Denote the frequency of the bin x, i.e., the number of x's in $\mathbf{x_1}$, by h_x, and sequentially decompress the message sequence $\mathbf{M}_1$ by the decompression algorithm $Decomp()$ according to the distribution $\mathbf{Q}_{Y|x} = \big(P_{Y|X}(0|x), \ldots, P_{Y|X}(B-1|x)\big)^T$ until the length of decompressed sequence is equal to h_x for $x = 0, 1, \ldots, B-1$. The decompression process is formulated by the following equation.

$$(b_1, \mathbf{y}'_{1,0}, \ldots, \mathbf{y}'_{1,B-1}) = Decomp(\mathbf{M}_1, \mathbf{Q}_{Y|X}, h_0, \ldots, h_{B-1}) \quad (8)$$



(a) (b) (c)

Fig. 3. Histograms: (a) Histogram of the host sequence. (b) Modified histogram obtained by PEE. (c) Modified histogram obtained by the proposed code.

Eq. (8) means that the first b_1 bits of $\mathbf{M}_1$, i.e., $(m_1, \ldots, m_{b_1})$, are decompressed into a sequence consisting of B sub-sequences, $\mathbf{y}'_{1,x}$, $0 \le x \le B-1$. The xth sub-sequence, $\mathbf{y}'_{1,x}$, has length h_x and is obtained with the distribution $\mathbf{Q}_{Y|x}$.

After that, we substitute all symbols "x" of $\mathbf{x_1}$ with $\mathbf{y}'_{1,x}$ for $x = 0, 1, \ldots, B-1$, by which we embed the first b_1 bits of $\mathbf{M}_1$ and generate the first stego block $\mathbf{y}_1$. In this process, the cover signal x is modified to the stego signal y with probability $P_{Y|X}(y|x)$. At the receiver side, the embedded bits can be extracted by compressing $\mathbf{y}'_{1,x}$ according to $\mathbf{Q}_{Y|x}$ for $x = 0, 1, \ldots, B-1$.

Because the expectation of h_x is equal to $KP_X(x)$ and we have assumed the compression algorithm can reach entropy, the average number of message bits embedded into the bin x is equal to

$$KP_X(x)H(\mathbf{Q}_{Y|x}), \quad x = 0, 1, \ldots, B-1. \tag{9}$$

Therefore, the average message length embedded into the first block, i.e., the expectation of b_1, can be estimated by

$$K\sum_{x=0}^{B-1} P_X(x)H\left(\mathbf{Q}_{Y|x}\right). \tag{10}$$

As our data embedding method is reversible, we should be able to restore $\mathbf{x}_1$, which can be fulfilled by embedding some overhead information of $\mathbf{x}_1$ into the subsequent blocks. Now, the question is, what's the overhead of $\mathbf{x}_1$? A natural strategy is to take the compressed version of $\mathbf{x}_1$ as its overhead. Because the receiver can restore $\mathbf{x}_1$ under the condition of known-$\mathbf{y}_1$, we implement a conditional compression to cut down the length of the overhead.

Denote the index set of all symbols "y" in $\mathbf{y}_1$ by $\mathbf{IY}_y$ and extract a sub-sequence $\mathbf{x}'_{1,y}$ from $\mathbf{x}_1$ according to $\mathbf{IY}_y$, such that $\mathbf{x}'_{1,y} = \{x_i | x_i \in \mathbf{x}_1 \text{ and } i \in \mathbf{IY}_y\}$. Thus, the block $\mathbf{x}_1$ is divided into B disjointed sub-sequences $\mathbf{x}'_{1,y}$ for $y = 0, 1, \ldots, B-1$. The probability distribution of the sub-sequence $\mathbf{x}'_{1,y}$ can be estimated by $\mathbf{Q}_{X|y}$, so we compress it according to $\mathbf{Q}_{X|y}$ such that

$$O(\mathbf{x}'_{1,y}) = Comp(\mathbf{x}'_{1,y}, \mathbf{Q}_{X|y}), \quad y = 0, 1, \ldots, B-1. \tag{11}$$

	−5	−4	−3	−2	−1	0	1	2	3	4	5
−5	1.000	0.884									
−4		0.116	1.000								
−3			0.000	0.868							
−2				0.132	0.542						
−1					0.458	0.195					
0						0.610					
1						0.195	0.457				
2							0.543	0.133			
3								0.867	0.000		
4									1.000	0.156	
5										0.844	1.000

Fig. 4. The optimal transition probability matrix $\mathbf{Q}_{Y|X}$ for Example 2, where the empty elements mean zeros.

Because the average length of $\mathbf{x}'_{1,y}$ is equal to $KP_Y(y)$ and we assume that the compression algorithm can reach entropy, the average length of $O(\mathbf{x}'_{1,y})$ can be estimated by

$$KP_Y(y)H(\mathbf{Q}_{X|y}). \tag{12}$$

We concatenate these compressed sub-sequences and get the overhead of $\mathbf{x}_1$, that is,

$$O(\mathbf{x}_1) = O(\mathbf{x}'_{1,0})||O(\mathbf{x}'_{1,1})||\cdots||O(\mathbf{x}'_{1,B-1}). \tag{13}$$

The average length of $O(\mathbf{x}_1)$ is given by

$$K\sum_{y=0}^{B-1} P_Y(y)H\left(\mathbf{Q}_{X|y}\right). \tag{14}$$

We concatenate $O(\mathbf{x}_1)$ at the front of the rest bits of $\mathbf{M}_1$ (i.e., except for the first b_1 bits that have been embedded) to generate $\mathbf{M}_2$. In the same manner as above, some bits at the front of $\mathbf{M}_2$ are embedded into $\mathbf{x}_2$ and the overhead of $\mathbf{x}_2$ is concatenated with the rest bits of $\mathbf{M}_2$ to generate $\mathbf{M}_3$, and then some bits of which are embedded into $\mathbf{x}_3$ and so forth. This process is implemented sequentially until the $(g-1)$th block.

In the last host block $\mathbf{x}_g$, we embed $\mathbf{M}_g$, that is just the overhead of $\mathbf{x}_{g-1}$, by replacing the LSBs of $\mathbf{x}_g$. The LSBs of $\mathbf{x}_g$ can be embedded into the previous $g-1$ blocks as a part of the message. The parameters for the recipient, including

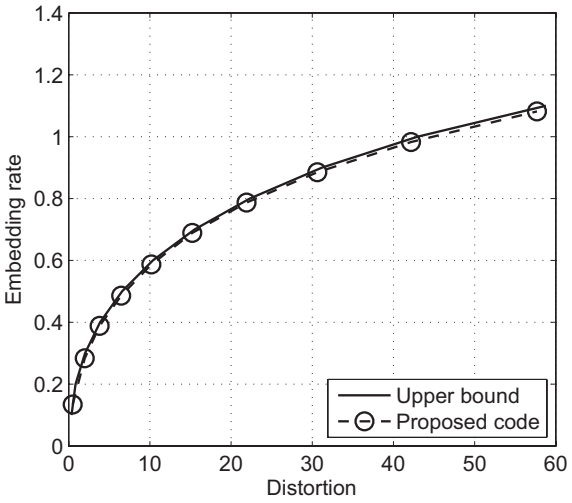


Fig. 5. Experiments on the performance of the proposed code construction.

(a) Lena.pgm (b) Baboon.pgm (c) Goldrill.pgm (d) Peppers.pgm

Fig. 6. Test images sized 512×512.

(a) Man.tiff (b) Grass.tiff

Fig. 7. Test images sized 1024×1024.

the host distribution $(P_X(0), \ldots, P_X(B-1))$, the distortion constraint Δ, and the block length K, are also embedded into the last block by LSB replacement. For the last block, we set the output $\mathbf{M}_{g+1}$ to be empty, meaning that the embedding process stops.

2) Data Extraction and Cover Restoration Processes: The data extraction and cover restoration are processed in a backward manner, such that $(\mathbf{M}_i, \mathbf{x}_i) = Ext(\mathbf{M}_{i+1}, \mathbf{y}_i)$ for $i = g-1, \ldots, 1$. From the $(i+1)$th stego block, we can extract the overhead $O(\mathbf{x}_i)$, by which we reconstruct the ith cover block $\mathbf{x}_i$. With the help of $\mathbf{x}_i$, we can extract the message from $\mathbf{y}_i$ by decompressing it according to the the optimal transition probability matrix $\mathbf{Q}_{Y|X}$.

First, from the LSBs of the last stego block $\mathbf{y}_g$, we extract $(P_X(0), \ldots, P_X(B-1))$, Δ, K, $\mathbf{M}_g$. According to the extracted parameters, we can calculate the optimal transition probability matrices $\mathbf{Q}_{Y|X}$ and $\mathbf{Q}_{X|Y}$.

Next, we describe how to do $(\mathbf{M}_{g-1}, \mathbf{x}_{g-1}) = Ext(\mathbf{M}_g, \mathbf{y}_{g-1})$. We denote the number of y in $\mathbf{y}_{g-1}$ by l_y, and sequentially decompress the message sequence $\mathbf{M}_g$ by the decompression algorithm $Decomp()$ according to the distribution $\mathbf{Q}_{X|y}$ until the length of decompressed sequence is equal to l_y for $y = 0, 1, \ldots, B-1$. The decompression process is formulated by the following equation.

$$(c_g, \mathbf{x}'_{g-1,0}, \ldots, \mathbf{x}'_{g-1,B-1}) = Decomp(\mathbf{M}_g, \mathbf{Q}_{X|Y}, l_0, \ldots, l_{B-1}). \quad (15)$$

Eq. (15) means that the first c_g bits of $\mathbf{M}_g$ are decompressed into a sequence consisting of B sub-sequences $\mathbf{x}'_{g-1,y}$, $0 \le y \le B-1$. The yth sub-sequence, $\mathbf{x}'_{g-1,y}$, has length l_y and is obtained with the distribution $\mathbf{Q}_{X|y}$. After that, we substitute all symbols "y" of $\mathbf{y}_{g-1}$ with $\mathbf{x}'_{g-1,y}$ for $y = 0, 1, \ldots, B-1$, by which we reconstruct the host block $\mathbf{x}_{g-1}$.

After that, we extract message bits from $\mathbf{y}_{g-1}$ with the help of $\mathbf{x}_{g-1}$. For $x = 0, \ldots, B-1$, denote the index set of all symbols "x" in $\mathbf{x}_{g-1}$ by $\mathbf{IX}_x$ and extract the sub-sequence $\mathbf{y}'_{g-1,x}$ from $\mathbf{y}_{g-1}$ according to $\mathbf{IX}_x$, such that $\mathbf{y}'_{g-1,x} = \{y_i | y_i \in \mathbf{y}_{g-1} \text{ and } i \in \mathbf{IX}_x\}$. We compress $\mathbf{y}'_{g-1,x}$ according to $\mathbf{Q}_{Y|x}$ for $x = 0, \ldots, B-1$, and concatenate the compressed sequences, which outputs the message embedded in $\mathbf{x}_{g-1}$, denoted by $\mathbf{M}'_{g-1}$. Finally, we generate the $\mathbf{M}_{g-1}$ by concatenating $\mathbf{M}'_{g-1}$ at the front of the rest bits of $\mathbf{M}_g$ (i.e. except for the c_g bits that have been decompressed.)

In the same manner, we can do $(\mathbf{M}_i, \mathbf{x}_i) = Ext(\mathbf{M}_{i+1}, \mathbf{y}_i)$, for $i = g-2, \ldots, 1$, which will restore the first $g-1$ host blocks and finally output the message $\mathbf{M}_1$. From $\mathbf{M}_1$, we take out the LSBs of the last block and restore the last block with LSB replacement.

Now we use a simple example with only one block to illustrate the coding and decoding processes of the method described above.

Example 1: As shown in Fig. 2, the host is a ternary sequence with distribution $(P_X(0), P_X(1), P_X(2)) = (0.7, 0.2, 0.1)$, and the block length $K = 10$. The first cover block $\mathbf{x}_1$ consists of seven "0", two "1" and one "2". Assume that, for the distortion constraint $\Delta = 0.6$, the optimal transition probability matrix

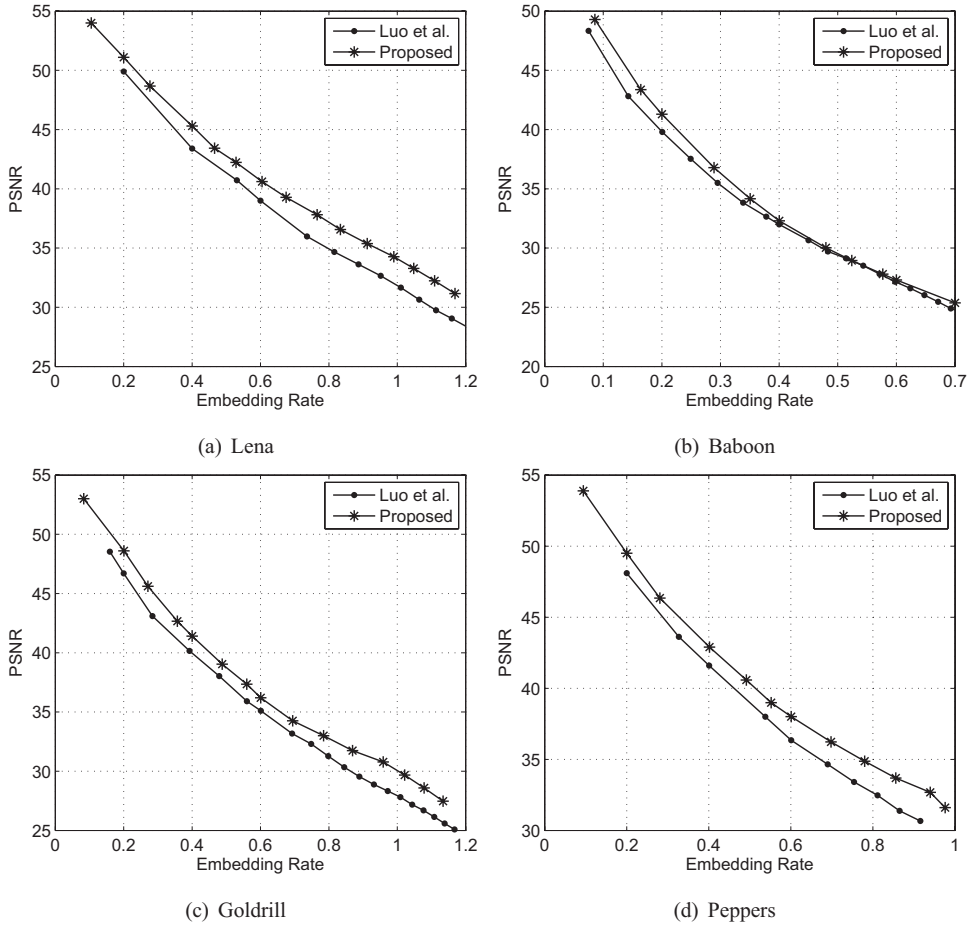
$$\mathbf{Q}_{Y|X} = \begin{pmatrix} 5/7 & 0 & 0 \\ 1/7 & 1/2 & 0 \\ 1/7 & 1/2 & 1 \end{pmatrix} \quad (16)$$

and we can calculate the other transition probability matrix

$$\mathbf{Q}_{X|Y} = \begin{pmatrix} 1 & 1/2 & 1/3 \\ 0 & 1/2 & 1/3 \\ 0 & 0 & 1/3 \end{pmatrix}. \quad (17)$$

Because the number of "0" in $\mathbf{x}_1$ is equal to 7 (i.e., $h_0 = 7$), we sequentially decompress the message bits into a 7-length ternary sequence according to the distribution $(5/7, 1/7, 1/7)$ that is the first column of $\mathbf{Q}_{Y|X}$. Assume that[1] the first 8 bits of the message are decompressed into a 7-length sequence $\mathbf{y}'_{1,0} =$

[1]Because the sequences in this example are too short, the compressed or decompressed sequences are artificial, which are only used to illustrate the processes of embedding, extraction and restoration.



(a) Lena (b) Baboon

(c) Goldrill (d) Peppers

Fig. 8. The experimental results on improving Luo *et al.*'s scheme [12].

$(0, 0, 2, 0, 1, 0, 0)$. Next, assume that the following 3 message bits are decompressed into a 2-length sequence $\mathbf{y}'_{1,1} = (1, 2)$ according to the second column of $\mathbf{Q}_{Y|X}$ because the number of "1" in $\mathbf{x}_1$ is equal to 2. No message is embedded in the bin "2" by decompression, because the entropy of the last column of $\mathbf{Q}_{Y|X}$ is zero. In $\mathbf{x}_1$, replacing all "0" by $\mathbf{y}'_{1,0}$ and all "1" by $\mathbf{y}'_{1,1}$, we get the stego block $\mathbf{y}_1$.

Next, we generate the overhead for restoring $\mathbf{x}_1$ according to $\mathbf{y}_1$ and $\mathbf{Q}_{X|Y}$. First, by recording the positions of "0", "1", and "2" in $\mathbf{y}_1$, we get $\mathbf{IY}_0 = \{1, 3, 6, 9, 10\}$, $\mathbf{IY}_1 = \{4, 7\}$, and $\mathbf{IY}_2 = \{2, 5, 8\}$, respectively. Note that the first column of $\mathbf{Q}_{X|Y}$ implies that we can directly interpret the "0" of $\mathbf{y}_1$ as "0" of $\mathbf{x}_1$ and thus we do not need to compress the five symbols of $\mathbf{x}_1$ at the index $\mathbf{IY}_0$. According to $\mathbf{IY}_1$ and $\mathbf{IY}_2$, we extract elements from $\mathbf{x}_1$ and get $\mathbf{x}'_{1,1} = (1, 0)$ and $\mathbf{x}'_{1,2} = (2, 0, 1)$. Assume that $\mathbf{x}'_{1,1}$ is compressed into $O(\mathbf{x}'_{1,1}) = (1, 0)$ according to the second column of $\mathbf{Q}_{X|Y}$, and $\mathbf{x}'_{1,2}$ is compressed into $O(\mathbf{x}'_{1,2}) = (0, 1, 1, 0, 1)$ according to the third column of $\mathbf{Q}_{X|Y}$. Concatenating $O(\mathbf{x}'_{1,1})$ and $O(\mathbf{x}'_{1,2})$ at the front of the rest bits of $\mathbf{M}_1$ (i.e., $(0, 1, \cdots)$), we get $\mathbf{M}_2 = (1, 0, 0, 1, 1, 0, 1, 0, 1, \cdots)$ that is embedded into the next block.

To reconstruct the host block $\mathbf{x}_1$ and extract the message from the stego block $\mathbf{y}_1$, we should first extract the message $\mathbf{M}_2$ from the second stego block $\mathbf{y}_2$, and observe $\mathbf{Q}_{X|Y}$ and $\mathbf{y}_1$. According to the first column of $\mathbf{Q}_{X|Y}$, the five "0" of $\mathbf{y}_1$ are directly interpreted as "0" of $\mathbf{x}_1$. Next, we count the number of "1" in $\mathbf{y}_1$ that is equal to 2, and thus sequentially decompress $\mathbf{M}_2$ according to the second column of $\mathbf{Q}_{X|Y}$ until generating a 2-length sequence. As a result, the first 2 bits of $\mathbf{M}_2$ are decompressed into $\mathbf{x}'_{1,1} = (1, 0)$. Similarly, the following 5 bits of $\mathbf{M}_2$ are decompressed into a 3-length sequence $\mathbf{x}'_{1,2} = (2, 0, 1)$ according to the third column of $\mathbf{Q}_{X|Y}$ because the number of "2" in $\mathbf{y}_1$ is 3. In $\mathbf{y}_1$, replacing all "1" by $\mathbf{x}'_{1,1}$ and "2" by $\mathbf{x}'_{1,2}$, we restore $\mathbf{x}_1$. After that, the rest bits of $\mathbf{M}_2$ are $(0, 1, \ldots)$.

To extract the message from $\mathbf{y}_1$, we record the positions of "0", "1" and "2" in $\mathbf{x}_1$ and get $\mathbf{IX}_0 = \{1, 3, 5, 6, 7, 9, 10\}$, $\mathbf{IX}_1 = \{4, 8\}$ and $\mathbf{IX}_2 = \{2\}$, according to which we extract elements from $\mathbf{y}_1$ and get $\mathbf{y}'_{1,0} = (0, 0, 2, 0, 1, 0, 0)$, $\mathbf{y}'_{1,1} = (1, 2)$, and $\mathbf{y}'_{1,2} = (2)$. According to the first column of $\mathbf{Q}_{Y|X}$, $\mathbf{y}'_{1,0}$ is compressed into $(1, 0, 0, 1, 0, 1, 1, 0)$, and according to the second column of $\mathbf{Q}_{Y|X}$, $\mathbf{y}'_{1,1}$ is compressed into $(0, 1, 1)$. No message has been embedded into bin "2" because the entropy of the last column of $\mathbf{Q}_{Y|X}$ is equal to zero. Concatenating the compressed sequences at the front of the rest bits of $\mathbf{M}_2$ (i.e., $(0, 1, \ldots)$), we get $\mathbf{M}_1$.

In the next example (Example 2), we compare the proposed code with one popular method for modifying prediction errors, called prediction-error expansion (PEE), which has

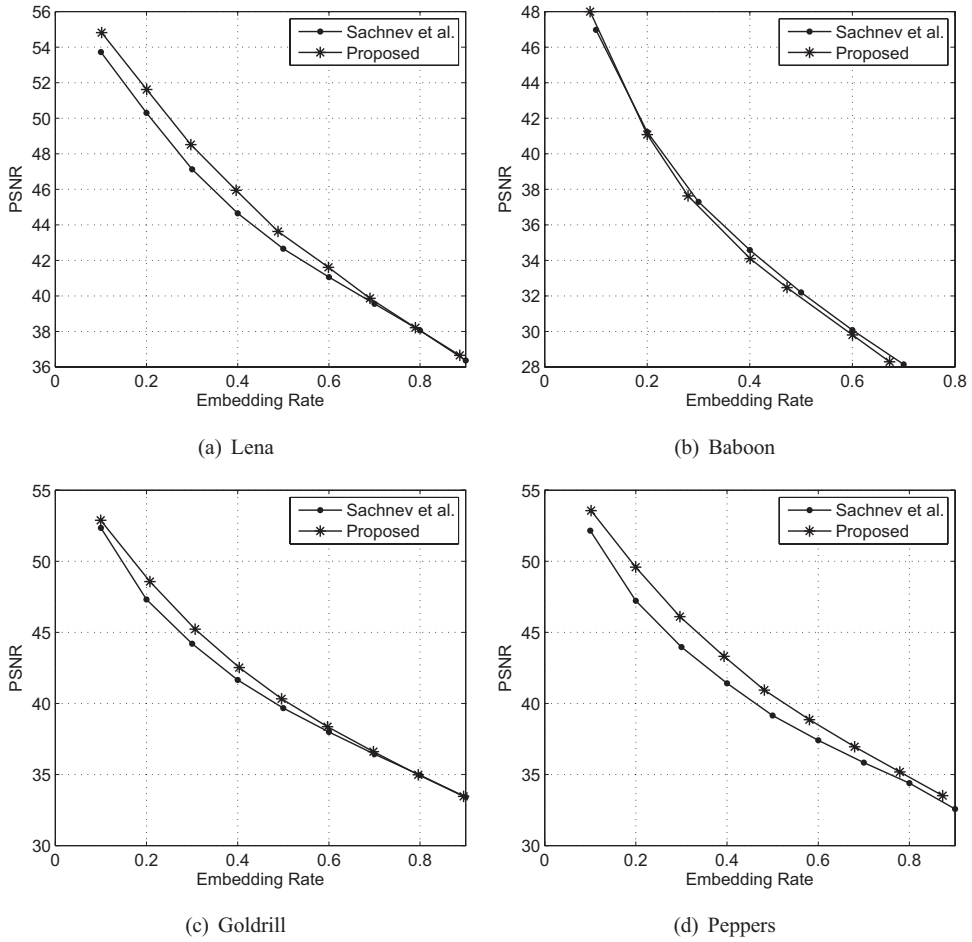


Fig. 9. The experimental results on improving Sachnev *et al.*'s scheme [11].

been adopted in many RDH schemes such as those proposed in [7], [8], [11]. In general, the prediction errors satisfy a Laplacian distribution centered at zero, so PEE uses a set around zero, $[T_n, T_p]$, to carry the message, where T_n is the negative threshold value, and T_p is the positive threshold value. Predicted errors in $[T_n, T_p]$ are expanded for embedding message bits, and the prediction errors not belonging to $[T_n, T_p]$ will be shifted to make room for the expansion. PEE modifies the prediction error e as follows.

$$e' = \begin{cases} 2e + b & \text{if } e \in [T_n, T_p] \\ e + T_p + 1 & \text{if } e > T_p \\ e + T_n & \text{if } e < T_n \end{cases}. \tag{18}$$

The decoder extracts the embedded bit b and reconstructs the prediction errors e in following manner:

$$b = e' \bmod 2, \quad e' \in [2T_n, 2T_p + 1] \tag{19}$$

$$e = \begin{cases} \lfloor e'/2 \rfloor & \text{if } e' \in [2T_n, 2T_p + 1] \\ e' - T_p - 1 & \text{if } e' > 2T_p + 1 \\ e' - T_n & \text{if } e' < 2T_n \end{cases}. \tag{20}$$

Example 2: In this example, the host sequence $\mathbf{x} = (x_1, \ldots, x_n)$ is drawn from a discrete Laplacian distribution with mean $\mu = 0$ and scale parameter $\delta = 2$. The length of host sequence $N = 10^6$, and the signal $x_i \in [-5, 5]$ for $1 \le i \le N$. The histogram of the host is shown in Fig. 3(a). We embed the message with embedding rate, 0.3, into $\mathbf{x}$ by using the proposed code and PEE respectively. When using PEE, we should set $T_n = -1$ and $T_p = 1$ for the embedding rate 0.3. To use the proposed code, we first calculate the optimal transition probability matrix $\mathbf{Q}_{Y|X}$, which is shown in Fig. 4.

According to the transition probability matrix in Fig. 4, the maximum amplitude of modifications for the proposed code is 1. For example, Fig. 4 means that "0" is modified to "1" and "−1" with equal probability 0.195. However, to shift the bins in PEE, we should add 2 to signals larger than 1. Consequently, for the embedding rate 0.3, the average distortion introduced by PEE is 0.77, while the average distortion of the proposed code is only 0.63. The histograms of stego sequences obtained by PEE and the proposed code are depicted in Fig. 3(b) and 3(c) respectively, which show that the tails of the host's histogram are extended to −6 and 7 by PEE.

B. Optimality

In this subsection, we will prove that our method is optimal in the sense that as long as the entropy coder reaches entropy, the proposed code asymptotically approaches the rate-distortion bound (1) when the number of blocks, g, tends to infinity. In fact, when g tends to infinity, the influence of the last block is negligible, and thus the average embedding rate

and distortion of the code construction can be estimated within one K-length block.

For simplicity, we assume Y is just a random variable satisfying the optimal marginal distribution P_Y that is determined by $\mathbf{Q}_{Y|X}$ and P_X. Therefore, the rate-distortion bound (1) can be rewritten as

$$\begin{aligned}\rho_{rev}(\Delta) &= maximize\{H(Y)\} - H(X) \\ &= H(Y) - H(X) \\ &= (H(X,Y) - H(X|Y)) - (H(X,Y) - H(Y|X)) \\ &= H(Y|X) - H(X|Y). \end{aligned} \tag{21}$$

On the other hand, in a K-length block of the code construction, we modify the host signal x to y according to the optimal transition probability $P_{Y|X}(y|x)$, so the average distortion d is given by

$$d = \sum_{x,y} P_X(x) P_{Y|X}(y|x) D(x,y). \tag{22}$$

Note that $P_{Y|X}$ is the solution of (1) under the condition (2), so we have $d \le \Delta$.

In a K-length block, the average number of embedded message bits is given by (10) and the average capacity cost for reconstructing this block is given by (14), so the embedding rate R in one block is given by ((10)–(14))/K, that is,

$$\begin{aligned} R &= \sum_{x=0}^{B-1} P_X(x) H\left(\mathbf{Q}_{Y|x}\right) - \sum_{y=0}^{B-1} P_Y(y) H\left(\mathbf{Q}_{X|y}\right) \\ &= \sum_{x=0}^{B-1} P_X(x) H\left(Y|X=x\right) - \sum_{y=0}^{B-1} P_Y(y) H\left(X|Y=y\right) \\ &= H(Y|X) - H(X|Y). \end{aligned} \tag{23}$$

Thus, we get $R = \rho_{rev}(\Delta)$.

C. Implementation Issues and Experimental Results

There are a few parameters that the recipient has to know for successful decoding, including the host distribution $(P_X(0), \ldots, P_X(B-1))$, the distortion constraint Δ, and the block length K. The probability $P_X(i)$, $(0 \le i \le B-1)$, can be calculated from the frequency f_i of symbol i, so we only need to communicate the vector of frequencies that can be easily compressed, and denote the length of the compressed frequency vector by L_f. Usually 10 bits are enough for saving Δ, and 13 bits are enough for K. We embed these parameters into the LSBs of the last block $\mathbf{x}_g$ by LSB replacement.

The last block should be long enough to accommodate these parameters. We denote the length of the last block by L_{last}, and embed it into the LSBs of the 15 elements at the tail of the host sequence. Note that, in the last block, we also should embed the overhead $O(\mathbf{x}_{g-1})$, whose length can be estimated by (14). Therefore, the last block must have a capacity for the overhead $O(\mathbf{x}_{g-1})$, L_f bits of the compressed frequencies, 10 bits of Δ, 13 bits of K, and 15 bits of L_{last}, so L_{last} must satisfy

$$L_{last} > K \sum_{y=0}^{B-1} P_Y(y) H\left(\mathbf{Q}_{X|y}\right) + L_f + 38. \tag{24}$$

Algorithm 1 Recursive Histogram Modification (RHM)

Data Embedding

- **0**. Input the cover sequence $\mathbf{x}$ with length N, the message sequence $\mathbf{m}$, and the distortion constraint Δ.
- **1**. Calculate frequencies f_i from $\mathbf{x}$, and set $P_X(i) = f_i/N$ for $0 \le i \le B-1$. According to P_X and Δ, calculate the optimal marginal distribution P_Y and optimal transition matrices $\mathbf{Q}_{Y|X}$ and $\mathbf{Q}_{X|Y}$.
- **2**. Set block length K, the length of the last block L_{last} and then determine the number of blocks g.
- **3**. Embed the message and the LSBs of the last block into the first $g-1$ blocks with the coding method described in Subsection III-A.
- **4**. Embed the overhead $O(\mathbf{x}_{g-1})$ and parameters, including compressed frequencies of host, Δ, K, and L_{last}, into the last block by LSB replacement.
- **5**. If $O(\mathbf{x}_{g-1})$ and the parameters can be completely embedded into the last block, output the stego sequence $\mathbf{y}$; otherwise, set $L_{last} = L_{last} + K$ and $g = g - 1$, and redo Step 3–Step 5.

Data Extraction and Host Restoration

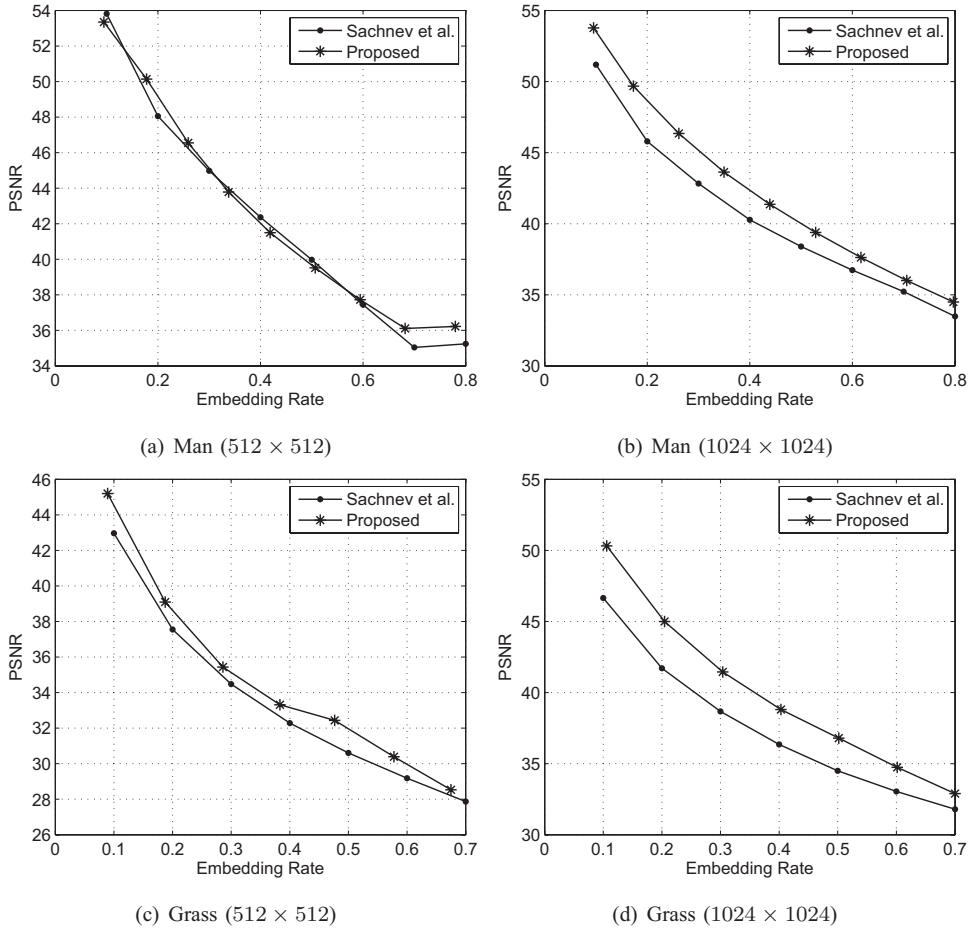
- **0**. Input the stego sequence $\mathbf{y}$ with length N.
- **1**. Extract the length of the last block, L_{last}, from the LSBs of the 15 elements at the tail.
- **2**. From the last block, extract $O(\mathbf{x}_{g-1})$ and the parameters, including compressed frequencies of host, Δ, and K.
- **3**. Decompress the frequencies, f_i, and set $P_X(i) = f_i/N$ for $0 \le i \le B-1$. According to P_X and Δ, calculate the optimal transition matrices $\mathbf{Q}_{Y|X}$ and $\mathbf{Q}_{X|Y}$.
- **4**. Implement the data extraction and host restoration procedure as described in Subsection III-A. Output the message $\mathbf{m}$, the LSBs of the last block, and the first $g-1$ blocks of the host sequence $\mathbf{x}$.
- **5**. Reconstruct the last block by LSB replacement.

We use (24) to estimate the lower bound of L_{last}, and set a somewhat larger L_{last} for enough capacity in the last block.

As a conclusion of above discussion, we present our code construction within an algorithm diagram (Algorithm 1).

Algorithm 1 is for the sender with a distortion constraint. A similar algorithm can be implemented for a sender with a given embedding rate R, in which we should first calculate the optimal distribution of problem (3) according to P_X and R.

We implemented the proposed code construction with arithmetic coder as the entropy coder. In the experiment, 10^6 8-bit gray-scale signals are drawn from a discrete Laplacian distribution with mean $\mu = 127.5$ and scale parameter $\delta = 5$, in which we delete signals whose frequencies are less than 100. Consequently, the rest host sequence includes 70 symbols, i.e., $B = 70$, with length equal to 999076. We set the block length $K = 100 \times B = 7000$. In this example, 980 bits are needed to communicate the 70 frequencies of host, i.e., $L_f = 980$, and the length of last block is about 4000, i.e., $L_{last} \approx 4000$, which fluctuates with different distortion constraints. The test was performed on Intel Core i7 running



(a) Man (512×512) (b) Man (1024×1024)

(c) Grass (512×512) (d) Grass (1024×1024)

Fig. 10. Comparison on improving Sachnev *et al.*'s scheme [11] for images with different sizes.

at 2.0 GHz with 2 GB RAM. The algorithm was implemented in Matlab R2008b. The average running time of one turn data embedding is 9.58 s in this setting. As shown in Fig. 5, the experimental results are quite close to the theoretic upper bound. This also proves the asymptotically optimality of the proposed code construction.

IV. Applications

We take Luo *et al.*'s scheme [12] and Sachnev *et al.*'s scheme [11] as examples to show how to improve previous RDH schemes with the proposed code.

Luo *et al.*'s scheme [12] divides the image into three parts, and predicts the pixel of one part with the pixels in the other two parts by interpolation. For each pixel x, denote the corresponding interpolation value by x', and then the predicted error (PE) is obtained via $e = x - x'$. Messages are embedded into the three PE sequences by histogram shift.

Sachnev *et al.*'s scheme [11] divides the image into two parts, and predicts one part with the other. The PEs are sorted according to magnitude of its local variance, so the PEs in smooth areas (i.e., PEs having values near to zero) are first used to embed the message. The message is embedded with the PEE method described in Example 2. The sorting technique makes Sachnev *et al.*'s scheme outperform most state-of-the-art methods.

We improve Luo *et al.*'s scheme and Sachnev *et al.*'s scheme by replacing histogram shift and PEE with the proposed code, respectively. When improving Sachnev *et al.*'s scheme, we do not need to sort the PEs because the code modifies host signals according to the optimal transition probability matrix which also gives priority to PEs with small absolutes.

Note that the PEs usually have a Laplacian-like distribution, in which the frequencies of errors having large absolute values are very small. When applying the proposed code, we first truncate the histogram of PEs according to a threshold T and only keep errors such that $f_e > T$, $e \in [-e_l, e_r]$, where f_e is the frequency of e, and the bounds, $-e_l$ and e_r, are determined by T. We embed the message into the PEs with Algorithm 1.

The threshold T should be adjusted according to the needed embedding rate (or distortion constraint). In fact, a smaller T leads to a larger B, for which we have to set a larger block length K in the coding process. Assume that the length of PEs is N, and then the block number is determined by N/K. To approach the rate-distortion bound with the proposed code, we hope N/K is large enough, so the block length K should be small for a finite N. However, when B is large, the distribution of signals in a short block can not be accurately estimated by the global distribution P_X and the entropy coder can not approach entropy, which means K should increase with B.

TABLE I
THE IMPROVEMENT OF PSNR (dB) FOR LUO et al.'S SCHEME [12]

Capacity (bpp)	Lena	Baboon	Goldrill	Peppers	Average
0.2	1.2	1.5	1.9	1.4	1.5
0.4	1.9	0.3	1.3	1.3	1.2
0.6	1.6	0.1	1.1	1.7	1.1

TABLE II
THE IMPROVEMENT OF PSNR (dB) FOR SACHNEV et al.'S SCHEME [11]

Capacity (bpp)	Lena	Baboon	Goldrill	Peppers	Average
0.2	1.3	−0.1	1.3	2.4	1.2
0.4	1.3	−0.5	1.0	1.9	0.9
0.6	0.5	−0.3	0.3	1.4	0.5

On the other hand, the maximum achievable embedding rate decreases with decreasing B. As a result, we set smaller values of B for smaller embedding rates, which can be realized by adjusting T. For Luo *et al.*'s scheme, which divides the image into three parts and thus produces short PE sequences, we set the threshold T according to the embedding rate R such that $T = \max\{400-300\times R, 50\}$. Because Sachnev *et al.*'s scheme produces longer PE sequences by dividing the image into two parts, we can set the threshold T to be somewhat small with $T = \max\{400 - 800 \times R, 10\}$.

In our experiments, four typical 8-bit gray-scale images [22] sized 512×512 are used as covers (Fig. 6). The improved results for three typical embedding rates are listed in Tables I and II, where the embedding rate is defined as bits per pixel (bpp). The proposed code improves Luo *et al*'s scheme by 1.5 dB on average at 0.2 bpp, 1.2 dB at 0.4 bpp, and 1.1 dB at 0.6 bpp; and improves Sachnev *et al.*'s scheme by 1.2 dB on average at 0.2 bpp, 0.9 dB at 0.4 bpp, and 0.5 dB at 0.6 bpp. More comparison results are shown in Figs. 8 and 9 respectively. On the image Baboon.pgm, the improvement for Luo *et al*'s schemes is small, and the proposed method is even worse than Sachnev *et al.*'s scheme. In fact, for the test image "Baboon.pgm", the histogram of the PEs is not that steep, so we have to set a large B and thus a large block length K even for a small embedding rate. Consequently, only few blocks are available for the coding process, which will greatly decrease the performance of the proposed code.

As proved in Section III-B, the proposed code can asymptotically approach the rate-distortion bound only when the block number tends to infinity. Therefore, if the cover length is short or the block number is small, the power of the code will be severely limited. To illustrate this point, we also do experiments on two large images [23] with size of 1024×1024 (Fig. 7). We first resize the images into 512×512, and then compare Sachnev *et al.*'s scheme with the improved method in the large image and its reduced version respectively. As shown in Fig. 10 the performances of the two methods are similar on the small images, while the proposed code can greatly improve Sachnev *et al.*'s scheme on the large images.

V. CONCLUSION

In this paper, we proposed a recursive code construction for RDH in gray-scale signals based on an entropy coder, and the main contributions include.

- The code construction is proved to be asymptotically optimal when the entropy coder is optimal, which establishes equivalency between RDH and lossless data compression.
- Experiment results show that the proposed code can improve previous RDH schemes, and the improvements will be more significant for larger cover images.

Note that the present code construction is for memoryless hosts. In fact, for RDH in a memory host, a higher rate-distortion bound maybe exists. So the interesting problems we will study in the future include: what is the rate-distortion bound and how to efficiently approach the bound for a memory host sequence?

REFERENCES

[1] F. Bao, R. H. Deng, B. C. Ooi, and Y. Yang, "Tailored reversible watermarking schemes for authentication of electronic clinical atlas," *IEEE Trans. Inf. Technol. Biomed.*, vol. 9, no. 4, pp. 554–563, Dec. 2005.

[2] J. Feng, I. Lin, C. Tsai, and Y. P. Chu, "Reversible watermarking: Current status and key issues," *Int. J. Netw. Security*, vol. 2, no. 3, pp. 161–171, May 2006.

[3] K. Chung, Y. Huang, P. Chang, and H.-Y. M. Liao, "Reversible data hiding-based approach for intra-frame error concealment in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 11, pp. 1643–1647, Nov. 2010.

[4] J. Fridrich and M. Goljan, "Lossless data embedding for all image formats," *Proc. SPIE*, vol. 4675, pp. 572–583, Jan. 2002.

[5] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.

[6] Z. Ni, Y. Shi, N. Ansari, and S. Wei, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.

[7] D. Thodi and J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.

[8] Y. Hu, H. Lee, and J. Li, "DE-based reversible data hiding with improved overflow location map," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 2, pp. 250–260, Feb. 2009.

[9] P. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," *Signal Process.*, vol. 89, no. 6, pp. 1129–1143, 2009.

[10] W. Hong, T.-S. Chen, and C.-W. Shiu, "Reversible data hiding for high quality images using modification of prediction errors," *J. Syst. Softw.*, vol. 82, no. 11, pp. 1833–1842, 2009.

[11] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 989–999, Jul. 2009.

[12] L. Luo, Z. Chen, M. Chen, X. Zeng, and Z. Xiong, "Reversible image watermarking using interpolation technique," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 1, pp. 187–193, Mar. 2010.

[13] X. Li, B. Yang, and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3524–3533, Dec. 2011.

[14] D. Coltuc, "Low distortion transform for reversible watermarking," *IEEE Trans. Image Process.*, vol. 21, no. 1, pp. 412–417, Jan. 2012.

[15] T. Kalker and F. M. Willems, "Capacity bounds and code constructions for reversible data-hiding," in *Proc. 14th Int. Conf. Digital Signal Process.*, 2002, pp. 71–76.

[16] W. Zhang, B. Chen, and N. Yu, "Capacity-approaching codes for reversible data hiding," in *Proc. 13th Inf. Hiding Conf.*, LNCS 6958. 2011, pp. 255–269.

[17] W. Zhang, B. Chen, and N. Yu, "Improving various reversible data hiding schemes via optimal codes for binary covers," *IEEE Trans. Image Process.*, vol. 21, no. 6, pp. 2991–3003, Jun. 2012.
[18] S.-J. Lin, and W.-H. Chung, "The scalar scheme for reversible information-embedding in gray-scale signals: Capacity evaluation and code constructions," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 4, pp. 1155–1167, Apr. 2012.
[19] F. Willems, D. Maas, and T. Kalker, "Semantic lossless source coding," in *Proc. 42nd Annu. Allerton Conf. Commun., Control Comput.*, Monticello, IL, USA, 2004, pp. 1–8.
[20] X. Hu, W. Zhang, X. Hu, N. Yu, X. Zhao, and F. Li, "Fast estimation of optimal marked-signal distribution for reversible data hiding," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 5, pp. 779–788, May 2013.
[21] (2013). *Source Codes for Estimating the Rate-Distortion Bound of RDH* [Online]. Available: http://home.ustc.edu.cn/~hxc
[22] (2013). *Miscelaneous Gray Level Images* [Online]. Available: http://decsai.ugr.es/cvg/dbimagenes/g512.php
[23] (2013). *SIPI Images* [Online]. Available: http://sipi.usc.edu/database/database.php

Weiming Zhang received the M.S. degree and Ph.D. degree in 2002 and 2005, respectively, from the Zhengzhou Information Science and Technology Institute, Zhengzhou, China. Currently, he is an Associate Professor with the School of Information Science and Technology, University of Science and Technology of China, Hefei, China. His research interests include multimedia security, information hiding and cryptography.

Xiaocheng Hu received the B.S. degree in 2010 from the University of Science and Technology of China, Hefei, China, where he is now pursuing the Ph.D. degree. His research interests include multimedia security, image and video processing, video compression and information hiding.

Xiaolong Li received the B.S. degree from Peking University, Beijing, China, the M.S. degree from Ecole Polytechnique, Palaiseau, France, and the Ph.D. degree in mathematics from ENS de Cachan, Cachan, France, in 1999, 2002, and 2006, respectively. Before joining Peking University as a researcher, he worked as a postdoctoral fellow at Peking University in 2007–2009. His research interests are image processing and information hiding.

Nenghai Yu received the B.S. degree in 1987 from Nanjing University of Posts and Telecommunications, Nanjing, China, the M.E. degree in 1992 from Tsinghua University, Beijing, China, and the Ph.D. degree in 2004 from the University of Science and Technology of China, Hefei, China, where he is currently a Professor. His research interests include multimedia security, multimedia information retrieval, video processing and information hiding.