Skewed Histogram Shifting for Reversible Data Hiding using a Pair of Extreme Predictions

Suah Kim, Xiaochao Qu, Vasily Sachnev, Hyoung Joong Kim

***Abstract*—Reversible data hiding hides data in an image such that the original image is recoverable. This paper presents a novel embedding framework with reduced distortion called skewed histogram shifting using a pair of extreme predictions. Unlike traditional prediction error histogram shifting schemes, where only one good prediction is used to generate a prediction error histogram, the proposed scheme uses a pair of extreme predictions to generate two skewed histograms. By exploiting the structure of the skewed histogram, only the pixels from the peak and the short tail are used for embedding, which decreases the distortion from the lesser number of pixels being shifted. Detailed experiments and analysis are provided using several image databases.**

***Index Terms*—Reversible data hiding, reversible watermarking, skewed histogram shifting**

I. Introduction

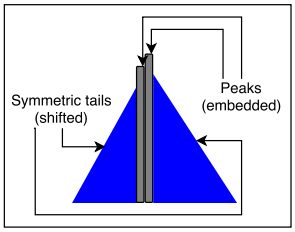
REVERSIBLE data hiding offers an interoperable way to store information in an image, such that the original image can be recovered when required.

There are several uses for reversible data hiding. The first use is for distortion-sensitive applications, such as military and medical imaging, where the authentication data is embedded as a watermark. In this case, image enhancement filters are applied directly to the watermarked image, while the original image can be recovered, and data can be extracted, when requested.

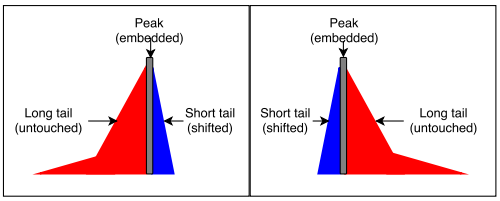
Data hiding in an encrypted image [1] is another use, where the original image recovery information is embedded as a reversible watermark. In this case, the original image recovery is offered only to privileged users, while watermarked image recovery is provided to all users.

Lastly, reversible data hiding is also finding applications in areas where contrast enhancement is needed with the support of the original image recovery [2–5]. In this case, the focus is on achieving contrast enhancement, data hiding and file size reduction [5], all at the same time.

S. Kim, and H. J. Kim are with Graduate School of Information Security, Korea University, Seoul, 02841, South Korea (email: suahnkim@gmail.com, khj-@korea.ac.kr). X. Qu is with MTlab (Meitu, Inc.), Beijing, 100080, China (email: qxc@meitu.com). V. Sachnev is with Catholic University of Korea, Bucheon, 14662, Korea (email: bassvasys@hotmail.com). This work was supported in part by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korean government (MSIP) (No.2018-0-00365, Development of on-off hybrid blockchain technology for real-time large-scale data distribution), in part by under the framework of international cooperation program managed by the National Research Foundation of Korea (2018K2A9A2A06024168, FY2018).



(a) Symmetric histogram



(b) Left skewed histogram (c) Right skewed histogram

Fig. 1: Different histogram distributions.

In this paper, we investigate a new low-distortion embedding framework based on the even and odd embedding of Sachnev et al.'s scheme [6] using skewed histograms.

In Sachnev et al.'s work [6], pixels are even, if the sum of a pixel's horizontal and vertical positions is an even number, and odd, if the sum is an odd number. Without loss of generality, the even set is embedded first, and then the odd set. This division allows for parallel predictions, and the use of an accurate prediction method called, rhombus prediction. The rhombus prediction is the average of the four direct neighbors, and its accuracy is much higher than the traditional predictors from image compression, because the traditional predictors can only utilize at most two direct neighbors, for the prediction. This usually results in a prediction error histogram, which is symmetric with sharp peaks around 0 and -1. The prediction error $p_e = p - \hat{p}$ is embedded using a map such as the one in Fig. 2, and the modification is reflected to the watermarked pixel p':

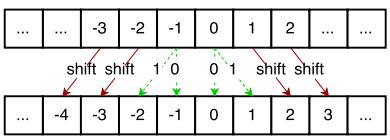


Fig. 2: Prediction error histogram mapping for Sachnev et al. [6].

$$p' = \begin{cases} p+b & \text{if } p-\hat{p}=0 \\ p-b & \text{if } p-\hat{p}=-1 \\ p+1 & \text{if } p-\hat{p}>0 \\ p-1 & \text{if } p-\hat{p}<-1 \end{cases} \quad (1)$$

where $b \in \{0,1\}$ is the payload bit.

Pixels with prediction errors 0 and -1 are used for embedding, while pixels with prediction errors that are smaller than -1 or greater than 0 are shifted away from the middle, causing distortion.

In this paper, we explore using not one, but two prediction values to create two skewed histograms, and use them to reduce the number of shifted pixels. Skewed histograms, histograms where bins are asymmetrically distributed and concentrated to one side, have better distribution for histogram shifting than symmetric histograms. Fig. 1b and c show that skewed histograms consist of a peak and two tails: a short and a long tail, which extend on either sides of the peak. In the proposed scheme, the peak is used for embedding, the short tail is used for shifting (due to embedding), while the long tail is left untouched. The number of shifted pixels can be smaller compared to embedding in a symmetric histogram, due to many pixels in the long tail not being modified.

Our contributions are as follows:

- A novel embedding scheme called skewed histogram shifting using a pair of extreme predictions is proposed.
- The effectiveness of the skewed histogram shifting is demonstrated.
- Extensive experiments and analysis are performed using images of varying sizes and payload sizes.

II. Related Work

Reversible data hiding for distortion-sensitive applications has been realized using techniques that are based on lossless compression [7–11], difference expansion [12–18], histogram shifting [6, 12, 19–42], integer transform [43–49], and distortion compensation [50, 51].

One of the key reversible data hiding techniques is called histogram shifting. This technique uses a histogram of pixels or some transformed version of the pixels, and shifts a set of peaks of the histogram in a reversible way to hide data. For example, the pixel-based histogram shifting technique proposed by Ni et al. [20] uses the most frequent pixels for data hiding. The difference expansion technique proposed by Tian [13] uses the histogram of the difference between the pixel and its immediate neighbor for data hiding. More advanced techniques include the histogram shifting in prediction error histogram, which requires less modification to hide data than the pixel or pixel difference histograms.

Extending the research in prediction error histograms, several researchers [12, 19, 39, 46, 52–55] have focused on embedding peak selection optimization. These techniques choose embedding peaks not by the frequency, but by distortion minimization for the specific payload size. While these techniques generally perform very well, they come with higher computational cost.

Identification of smooth and rough areas has also been of interest. This is because embedding in smooth areas tends to cause less distortion than in rough areas. Exceptional work proposed by Kamstra and Heijmans [15], and Sachnev et al. [6] sorted pixels by local complexity values in order to embed in smooth areas before embedding in rough areas. Since then, improved local complexity functions have been proposed by Ou et al. [23] and Li et al. [55], which performs better than previous works.

Recent research has focused on histogram shifting using pixel value ordering [26–28], which uses an ordering of the pixel blocks with prediction error histogram shifting to achieve very high PSNR for very small payload.

On the other hand, embedding large payload has been explored by Dragoi and Coltuc [56], Lee et al. [57], and Hwang et al. [58]. These techniques utilize regression-based prediction methods to increases the accuracy of the predictions. Although it increases the computational power, a more sharply distributed prediction error histogram suitable for embedding large payload is produced.

Li et al. [24] and Gui et al. [59] explored a new embedding point optimization called two-dimensional (2D) histogram shifting. In 2D histogram shifting, each neighboring pair of prediction errors (e_1, e_2) is either embedded or shifted according to a predefined map. Modifying the prediction errors as a pair gives more precise control over which pixels are embedded, and which are shifted. Although Refs. [24, 59] only focused on using a 2D pixel difference prediction error histogram for embedding, it provided inspiration and the foundation for the recent state of the art techniques called pairwise prediction error expansion technique by Ou et al [23], and adaptive pairing reversible data hiding by Dragoi and Coltuc [17]. Among the several maps proposed by Refs. [23], one map called 'T1' stands out, as it shuffles around the embedding points to allow embedding in prediction error values equal to not just 0 and -1, but other points as well; check Refs. [23] for more details. This method is further improved by using an improved local complexity function, which takes consideration of more neighbors. The combination of the two ideas has demonstrated superior performance against existing 1D methods. Refs. [23] also presents several graphical representations of 2D prediction error histogram shifting mappings, which can exploit the correlation of the pairs of prediction errors. Their experimental results show the superiority of their method when compared to 1D methods. On the other hand, Refs. [17] is an optimized version of Refs. [23], in which pairing the pixels is not fixed; instead, it is adaptive, creating a more compact 2D prediction error histogram.

Other recent works also include techniques in distortion compensation [50, 51], a double-embedding technique in which the distortion from the first embedding maybe removed or reduced by the second embedding. This research area is quite fresh, and provides an interesting alternative to the traditional scheme.

Like distortion compensation techniques, Chen et al. [60] proposed a double-embedding technique based on directed prediction error expansion. The technique generates two slightly different prediction error histograms, using two vari-

ations of gradient adjusted predictor (GAP), and embeds in each histogram in a consecutive manner.

III. Proposed: Skewed Histogram Shifting Scheme

Traditional image coding techniques rely on the symmetric Laplacian or Gaussian distribution of the prediction error for compression. Similarly, traditional reversible data hiding utilizes the symmetric sharp distribution to embed in the frequent peaks to maximize the embedding capacity while minimizing the shifting distortion. The proposed method takes a different approach by investigating different methods to generate skewed asymmetrical histograms that are suited to histogram shifting: asymmetric histograms with a short tail and a long tail relative to their modes.

Generating a skewed prediction error histogram is quite simple. Consider the prediction error equation: $p_e = p - \hat{p}$. If the prediction $\hat{p}$ is a high estimate, the prediction error p_e will likely be negative, *i.e.,* the majority of the prediction error values will be negative, forming a left skewed histogram (see Fig. 1b). Conversely, a prediction histogram generated using a low estimate will form a right skewed histogram (see Fig. 1c).

The proposed method takes advantage of the asymmetry of the skewed histogram and shifts the pixels only within the short tail part, leaving the pixels in the long tail part almost unmodified. In conclusion, asymmetric prediction histograms can have the previously unobserved benefits of reducing distortion.

In this section, we present a reversible data hiding scheme called skewed histogram shifting that uses a pair of extreme predictions. The steps are as follow:

1) Make a pair of extreme predictions for the pixel.
2) Embed using positive histogram shifting (PHS).
3) Embed using negative histogram shifting (NHS).

where, PHS and NHS are the unidirectional histogram shifting method, and their shifting direction is fixed to their respective signs, *i.e.,* PHS only allows histogram bins to move towards the positive direction, whereas NHS only allows histogram bins to move towards the negative direction.

Sections III-A and III-B describe the embedding, extraction, and recovery of the skewed histogram shifting. Section III-C outlines the proposed extreme prediction predictors. Finally, Section III-D presents the proposed local complexity function.

A. Embedding

Even and odd embedding is used in the proposed method; without loss of generality, we embed the even sets first, and then embed the odd sets. Pixels are even, if the sum of the horizontal and vertical positions is even; likewise, odd, if the sum is odd.

Each pixel is embedded twice in a row, first using the high estimate $\hat{p}_h$, and then using the low estimate $\hat{p}_l$. The exact formulations of $\hat{p}_h$ and $\hat{p}_l$ are discussed in the next subsection.

The first skewed prediction error histogram is $p - \hat{p}_h$. One bit is embedded in p using PHS:

$$p' = \begin{cases} p + b_1 & \text{if } p - \hat{p}_h = 0 \\ p + 1 & \text{if } p - \hat{p}_h > 0 \\ p & \text{else} \end{cases} \tag{2}$$

where $b_1 \in \{0, 1\}$ is the first message bit, and p' is the watermarked pixel. One bit is embedded only if the prediction error is 0, and the other pixels are shifted by 1 only if their prediction errors are strictly positive.

The second skewed prediction error histogram is $p' - \hat{p}_l$. One bit is embedded in p' using NHS:

$$p'' = \begin{cases} p' - b_2 & \text{if } p' - \hat{p}_l = 0 \\ p' - 1 & \text{if } p' - \hat{p}_l < 0 \\ p' & \text{else} \end{cases} \tag{3}$$

where $b_2 \in \{0, 1\}$ is the second message bit, and p'' is the watermarked pixel. A bit is embedded only if the prediction error is 0, and other pixels are shifted if the prediction errors are strictly negative.

The proposed scheme only modifies the pixel value by at most $\pm$ 1, because PHS modifies it by either 0 or +1, whereas NHS modifies it by either 0 or -1.

B. Extraction and Recovery

For extraction and recovery, the steps from embedding are done in reverse order, *i.e.,* the odd pixels are recovered before the even pixels. First, the odd pixels are sorted. Message bits b_1 and b_2 can be extracted using the following equations:

$$b_2 = \begin{cases} 0 & \text{if } p'' - \hat{p}_l = 0 \\ 1 & \text{if } p'' - \hat{p}_l = -1 \end{cases} \tag{4}$$

$$b_1 = \begin{cases} 0 & \text{if } p' - \hat{p}_h = 0 \\ 1 & \text{if } p' - \hat{p}_h = 1 \end{cases} \tag{5}$$

Then, pixel p is recovered using the following equations:

$$p' = \begin{cases} p'' + 1 & \text{if } p'' - \hat{p}_l < 0 \\ p'' & \text{else} \end{cases} \tag{6}$$

$$p = \begin{cases} p' - 1 & \text{if } p' - \hat{p}_h > 0 \\ p' & \text{else} \end{cases} \tag{7}$$

The same steps are repeated for the even set as well. The embedding capacity and the distortion will vary depending on the pair of extreme predictions $\hat{p}_h$ and $\hat{p}_l$. The next section presents three prediction methods for $\hat{p}_h$ and $\hat{p}_l$.

C. Proposed pairs of extreme predictors

In the proposed method, the predictors are evaluated as a pair $(\hat{p}_h, \hat{p}_l)$ using the neighboring pixel values. First, the four immediate neighboring pixel values {N, W, S, E} (see Fig. 4a) are sorted in ascending order: $x_1 \leq x_2 \leq x_3 \leq x_4$. Table 1, in the bottom of the page, lists three proposed predictor pairs: Predictor 1 has the most extreme prediction values (highest and lowest neighboring values), whereas Predictor 3

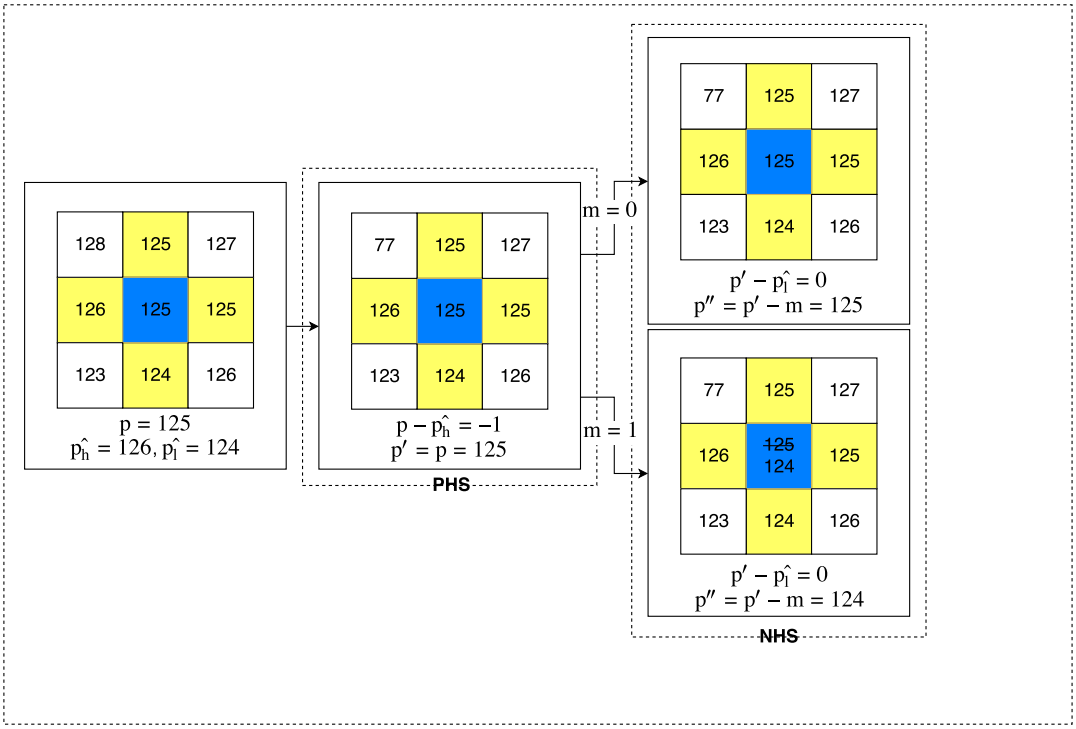


Fig. 3: Embedding example using predictor 1.

Predictor #	1	2	3
$\hat{p}_h$	x_4	$\left\lfloor \frac{x_4 + x_3}{2} \right\rceil$	$\left\lfloor \frac{x_4 + x_3 + x_2}{3} \right\rceil$
$\hat{p}_l$	x_1	$\left\lfloor \frac{x_2 + x_1}{2} \right\rceil$	$\left\lfloor \frac{x_3 + x_2 + x_1}{3} \right\rceil$

TABLE I: Predictor pairs. $\lfloor . \rceil$ rounds to the nearest integer.

returns the most collaborative pair of prediction values (most neighbors are used for prediction).
Following the trend, Predictor 4 would be:

$$\hat{p}_h = \hat{p}_l = \left\lfloor \frac{x_4 + x_3 + x_2 + x_1}{4} \right\rceil \tag{8}$$

However, since the prediction values are the same, embedding would be the same as double-embedding around 0, which is known to not perform well; thus, it is not analyzed in this paper.

Furthermore, to avoid double embedding around 0, the equation below is used to ensure that the pair of prediction values are never equal:

$$\hat{p}_l = \begin{cases} \hat{p}_l - 1 & \text{if } \hat{p}_l = \hat{p}_h \\ \hat{p}_l & \text{else} \end{cases} \tag{9}$$

Note that using Eq. (9) has no effect on the reversibility as it is applied after the prediction and before the embedding. The analysis of the predictors and Eq. (9) is shown in the last section.

A short example of embedding using Predictor 1 is shown in Fig. 3 for embedding $m = 0$ and $m = 1$ in $p = 125$. During PHS, m cannot be embedded, since the prediction error $(p - \hat{p}_h)$ is not 0; however, during NHS, m can be embedded, since the prediction error $(p - \hat{p}_l)$ is 0.

D. Local complexity functions

In literature, a technique called pixel selection is used to further reduce the distortion. The main idea of pixel selection is choosing less distortion-causing pixels first for embedding. Local complexity values are used to rank the pixels by examining their likelihood of being embeddable. Generally, a smaller value indicates that it is a smooth pixel, *i.e.*, more embeddable. Therefore, pixels with smaller local complexity values are embedded first. In this subsection, two techniques from the literature and one proposed technique are presented: the four-pixel version, the pairwise version, and the proposed extended version.

Fig. 4a shows the context of the four-pixel version, which uses the four direct neighbors to determine the local complexity value. Fig. 4b shows the context of the pairwise version [23], which uses pairs of pixels' neighbors. Fig. 4c shows the context of the proposed scheme, which uses a similar idea of incorporating more neighbors. The following gives a description of the relevant functions:

1) Four-pixel version (see Fig. 4a):
   $C(p) = |N - W| + |W - S| + |S - E| + |N - E| + |W - E| + |N - S|$
2) Pairwise version (see Fig. 4b):
   $C_{pair}(p_1) = |N_1 - W_1| + |W_1 - S_1| + |S_1 - E_1| + |N_1 - E_1| + |S_1 - S_2| + |S_2 - E_2| + |E_2 - E_1| + |E_1 - F_1| + |S_1 - F_2|$
3) Extended version (see Fig. 4c):
   $\tilde{C}(p) = C(p) + C(NW) + C(NE) + C(SE) + C(SW)$

IV. Encoder and Decoder.

In this section, the encoder and decoder are presented. Before presenting the encoder and the decoder, the side information needed for perfect recovery is discussed.

A. Side information

The decoder requires side information for data extraction and original image recovery. Each part of the side information is described as following:

Preprocessing and Location Map: A preprocessing step and location map are necessary to avoid an underflow and overflow problem. Embedding a pixel can result in a watermarked pixel with under and overflow values -1 and 256; but this can be solved if pixel values 0 and 255 are preprocessed to 1 and 254, respectively. The general procedure is to modify the image prior to embedding (preprocessing) and embed the modification history (location map) as part of the payload. The location map $M = \{M_1, ..., M_i, ..., M_n\}$ is generated and preprocessing from the original pixel p_o to the preprocessed pixel p is implemented in the following way:

1) Scan the image in raster scanning order.

$$M_i = \begin{cases} 0 & \text{if } p_o = 1 \text{ or } 254 \\ 1 & \text{if } p_o = 0 \text{ or } 255 \end{cases} \tag{10}$$

$$p = \begin{cases} 254 & \text{if } p_o = 255 \\ 1 & \text{if } p_o = 0 \\ p_o & \text{else} \end{cases} \tag{11}$$

2) Repeat the above steps until the whole image is scanned.

During the decoding stage, recovery of the pixel values 0 and 255 using M is trivial:

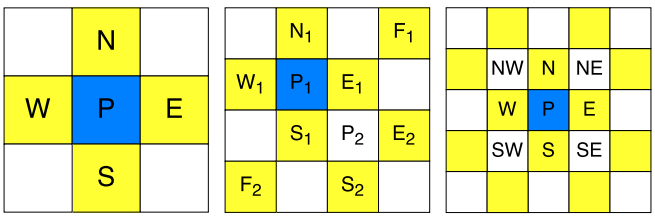
1) Scan the image in raster scanning order.

$$p_o = \begin{cases} 0 & \text{if } p = 1 \text{ and } M_i = 1 \\ 255 & \text{if } p = 254 \text{ and } M_i = 1 \end{cases} \tag{12}$$

2) Repeat the above steps until the whole image is scanned.

Finally, the location map is losslessly compressed using arithmetic coding and appended to the front of the payload, along with its size. Given an image size of $n \times m$, the upper limit to represent the size of the location map is $t = \lceil log_2(n \times m) \rceil$ bits. For an image size of 512×512, this is 18 bits.

Payload length: The payload length is recorded in the LSBs of the border pixels using the LSB replacement method, and the original LSBs are appended in front of the payload. It should be sufficient to limit the $t = \lceil log_2(n \times m) \rceil$ bits to represent the size of the payload. For an image size of 512×512, this is 18 bits. The payload length is used to know when to stop decoding.



(a) Four-pixel context (b) Pairwise context (c) Extended context

Fig. 4: Context used for different local complexity functions. Yellow pixels represent the context, and the blue pixels are the target pixels.

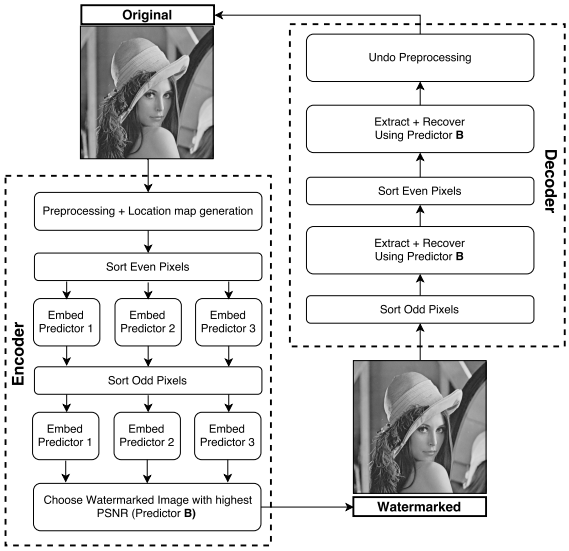


Fig. 5: Encoder and decoder

Predictor number: All three predictors are used for embedding, and the watermarked image with the highest PSNR is chosen. The predictor information is recorded using the LSB replacement method in the border pixels' LSB. The predictor number can be represented using two bits.

LSBs of the border pixels: Payload length (t bits) and predictor number (2 bits) are recorded in the border pixel's LSBs, and the $t + 2$ border pixel LSBs are appended to the front of the payload and used during recovery.

B. Encoder

Given an image of size $n \times m$, the embedding is done using the following steps:

1) Preprocess and generate the location map.
2) Append the location map's size, the location map, and the LSBs of the $t + 2$ border pixels to the payload.
3) For each predictor, perform Steps 4)-8).
4) Replace the $t + 2$ border pixel LSBs with the payload length and predictor number.
5) Sort the even pixels using $\tilde{C}(p)$.
6) Use PHS, and then NHS to embed the first half of the payload.
7) Sort the odd pixels using $\tilde{C}(p'')$.
8) Use PHS, and then NHS to embed the second half of the payload.
9) Calculate the PSNR between the original and each of the watermarked images.
10) Save the watermarked image with the highest PSNR.

C. Decoder

Extraction and recovery are done using the following steps:

1) Read the $t + 2$ border pixel LSBs and extract the predictor number and the payload length.
2) Sort the odd pixels using $\tilde{C}(p'')$.
3) Undo NHS, then PHS, and recover and extract the second part of the payload.
4) Sort the even pixels using $\tilde{C}(p)$.
5) Undo NHS, then PHS, and recover and extract the first part of the payload.
6) Replace the first $t + 2$ border pixel's LSBs with the original LSBs.
7) Uncompress the location map, and undo the preprocessing.

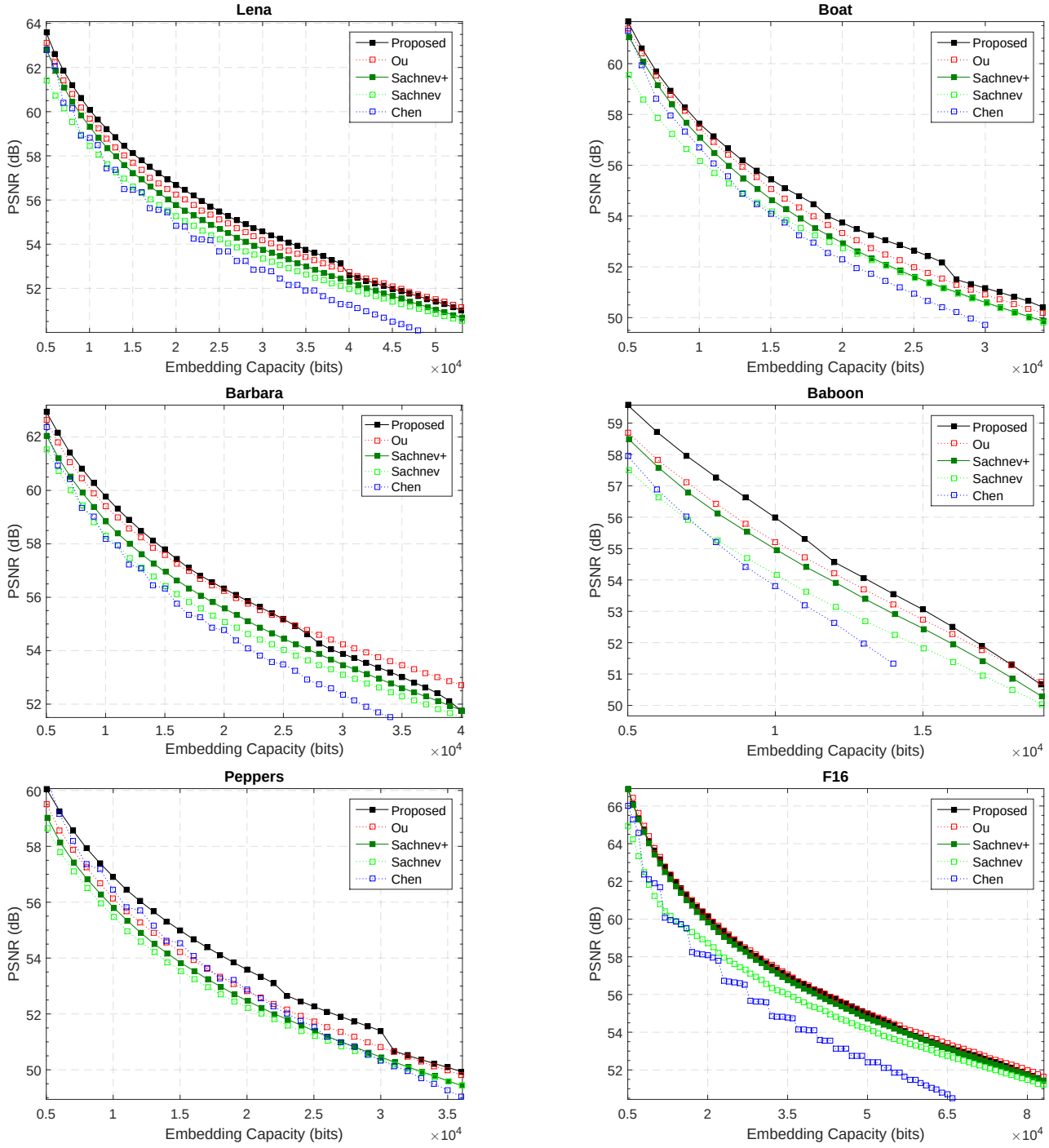


Fig. 6: PSNR comparison for the USC-SIPI image database. 'Ou' is [23], 'Sachnev+' is [6]+, 'Sachnev' is [6], and 'Chen' is [60].

The graphical figure for encoder and decoder is provided in Fig. 5.

V. Experimental Result and Analysis

This section compares the proposed method using three image databases: USC-SIPI[1], Kodak[2], and BOSS v1-1[3]. The USC-SIPI database is used for detailed analysis, whereas Kodak and BOSS databases are used to collect the statistics of the comparison results[4]. The goal of the comparison here is to compare the proposed scheme with the other basic schemes without peak optimization to provide clear analysis.

A. USC-SIPI database

USC-SIPI is a popular image database for testing various image processing techniques, and contains several 512×512

[1]http://sipi.usc.edu/database/
[2]http://www.r0k.us/graphics/kodak/
[3]http://agents.fel.cvut.cz/stegodata/RAWs/BossBase-1.0-raw-1.tar

[4]Our code will be available at https://github.com/suahnkim/skewed_histogram_shifting after publication

sized color and grey-scaled images.

The proposed method is compared with Sachnev et al. [6], updated version of [6] (denoted as [6]+ or Sachnev+), Ou et al. [23], and Chen et al. [60].

- [6] is one of the most well-known benchmark algorithms for reversible data hiding.
- [6]+ is updated with the proposed location map and the extended local complexity function.
- [23] is one of the most recent state of the art reversible data hiding techniques which utilizes the 2D histogram shifting, which performs strictly better than existing methods without embedding peak optimization.
- [60] is the histogram shifting using directed-prediction scheme, which utilizes two variants of gradient adjusted predictors (GAP) to produce two histograms. This is similar to the proposed algorithm; it uses a pair of predictions to generate two prediction error histograms, but the difference in prediction values exists only for the edges: edge prediction values are assigned constant values of either 0 or 255, depending on the histogram.

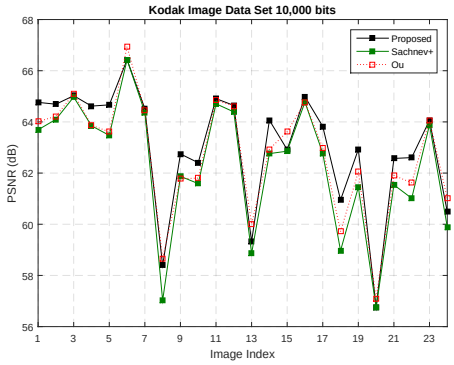
Payloads starting from 5,000 bits, increasing in the order of 1,000 bits, are tested and corresponding PSNR values are compared.

Fig. 6 shows that in general, Chen et al.'s method [60] does not perform well, except for the image Peppers. For image Peppers, it performs better or the same as [23] for payload between 5,000 and 20,000 bits. More importantly, it performs strictly worse than the proposed method for all images and payloads.

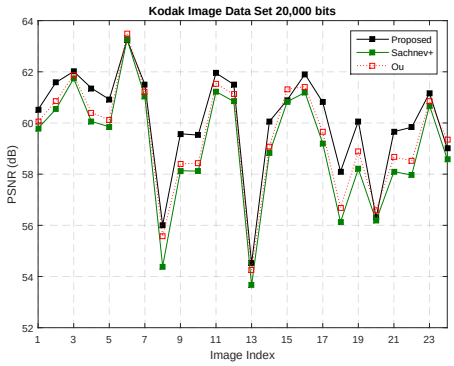
On the other hand, [6]+ noticeably performs better than [6] in lower payload areas, especially for image F16 (the performance improvement is a direct result of the updated local complexity function).

The proposed method performs better or close to [23] across most payloads. However, it performs worse for image Barbara, near the maximum embedding capacity (rough areas), because the rough areas of image Barbara have many prediction errors equal to 1 and -2 and only [23] can possibly embed in those prediction errors. Thus, for this specific case, [23] embeds with smaller distortion compared to the proposed method and [6]. Overall, the proposed method performs better than [23] in noisy images such as Boat and Peppers.

In conclusion, the proposed method performs very well with noisy images such as Boat and Peppers, because prediction error histograms tend to be more skewed, leading to a smaller number of pixels that are shifted. For smooth images such as F16, the proposed method lacks the advantage, because the rhombus predictor's prediction value and the proposed predictor's prediction values are more likely to be similar, *i.e.*, the generated prediction error histograms are barely skewed.



(a) 10,000 bits



(b) 20,000 bits

Fig. 7: PSNR comparison for the Kodak image database.

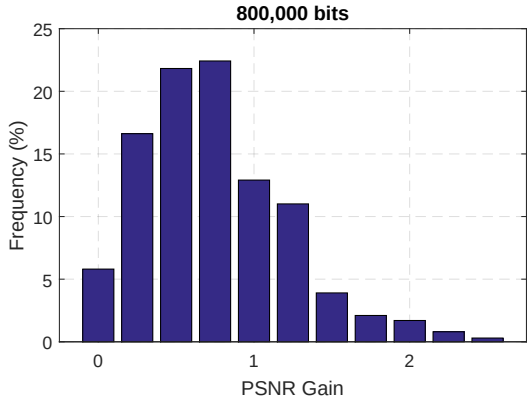
	10,000 bits			20,000 bits		
#	[6]+	[23]	Proposed	[6]+	[23]	Proposed
1	63.69	64.01	**64.76**	59.79	60.05	**60.52**
2	64.09	64.22	**64.69**	60.53	60.85	**61.58**
3	64.97	**65.09**	65.02	61.73	61.86	**62.01**
4	63.84	63.86	**64.61**	60.05	60.40	**61.35**
5	63.47	63.61	**64.66**	59.85	60.11	**60.92**
6	66.42	**66.94**	66.42	63.27	**63.48**	63.24
7	64.37	64.44	**64.51**	61.02	61.22	**61.50**
8	57.02	**58.66**	58.39	54.39	55.56	**56.01**
9	61.87	61.77	**62.74**	58.13	58.40	**59.57**
10	61.58	61.80	**62.40**	58.11	58.42	**59.52**
11	64.70	64.84	**64.90**	61.22	61.53	**61.95**
12	64.39	64.62	**64.64**	60.84	61.12	**61.50**
13	58.85	**60.01**	59.32	53.66	54.24	**54.52**
14	62.76	62.93	**64.06**	58.81	59.07	**60.05**
15	62.85	**63.61**	62.91	60.82	**61.31**	60.89
16	64.77	64.77	**64.98**	61.19	61.41	**61.90**
17	62.77	62.97	**63.82**	59.20	59.65	**60.80**
18	58.96	59.71	**60.94**	56.12	56.68	**58.10**
19	61.44	62.04	**62.92**	58.20	58.89	**60.06**
20	56.73	**57.09**	56.75	56.18	**56.58**	56.30
21	61.53	61.90	**62.57**	58.09	58.66	**59.66**
22	61.01	61.62	**62.60**	57.96	58.52	**59.83**
23	63.88	64.05	**64.06**	60.68	60.84	**61.16**
24	59.89	**61.00**	60.48	58.58	**59.34**	58.99
Average	62.33	62.73	**63.05**	59.10	59.51	**60.08**

TABLE II: PSNR comparison for the Kodak database.

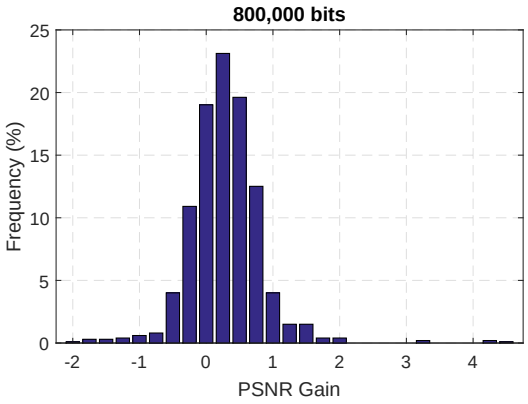
B. Kodak database

Extended experiments were performed using the Kodak database. This database has 24 color images with size of 512 $\times$ 768. The images are first converted to gray-scaled images using the built-in MATLAB function *rgb2gray()*.

Table II and Fig. 7 show the PSNR values for 10,000 and 20,000 bits, where the table entry is in bold if it is the highest. The proposed method has the highest PSNR value for 17 and 20 images for 10,000 and 20,000 bits, respectively. The proposed algorithm has the highest average PSNR value



(a) Proposed vs Sachnev+



(b) Proposed vs [23]

Fig. 8: PSNR comparison for the BOSS database.

as well. Compared to [6]+, the proposed method has higher PSNR value by 0.72 dB and 0.98 dB for 10,000 and 20,000 bits, respectively. Compared to [23], the proposed method has higher PSNR value by 0.32 dB and 0.57 dB for 10,000 and 20,000 bits, respectively.

Overall, the proposed method performs better than [6]+ and [23], and the PSNR gain for 20,000 bits is larger than for 10,000 bits. The next subsection describes additional experiments using a larger database with larger images.

C. Boss database

To verify the experiments at larger scale, Break Our Steganography System (BOSS) database is used. For the experiment, 1,000 raw color images of size $2,602 \times 3,906$ in CR2 format are used. To use the raw pixels, the images are first converted to PGM format using DCRaw[5] and then converted to gray-scaled images.

Instead of testing the payload size of 2×10^4 bits, the payload size of 8×10^5 bits is tested, since when compared to the USC-SIPI images, the BOSS images have about 40 times the number of pixels. Image 229 has been excluded from the result, because only the proposed method successfully embedded the payload, due to the large location map size.

Fig. 8 summarizes the gain in the PSNR result. The proposed algorithm has the highest average PSNR value; it performs better than [6]+ by 0.77 dB and better than [23] by 0.29 dB, and it has higher PSNR values than [23] for 74.1% of the images.

[5]http://www.cybercom.net/~dcoffin/dcraw/

VI. Analysis of the Extreme Predictors

In this section, the proposed predictors are analyzed by their embedding and distortion capability. The initial analysis focuses on the result of each predictor when Eq. (9) is not used (indicated by an asterisk '*' after the predictor number), and the improvement resulting from Eq. (9) is discussed in the last subsection.

A. Pixel classification

For analysis of the extreme predictors, the following Table III is used to classify each pixel,

	$L_{ec} = 0$	$L_{ec} = 1$	$L_{ec} = 2$
$L_d = 0$	Skipped	Embedded	Twice Embedded
$L_d = 1$	Shifted	Embedded	Twice Embedded

TABLE III: Pixel Classification.

where, L_d represents the local distortion, the absolute difference between the watermarked pixel p'' and the original pixel p, and L_{ec} represents the local embedding, the number of bits embedded in p''.

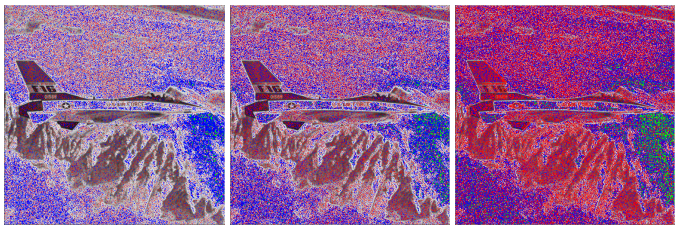
Fig. 9 shows a graphical representation of the distribution to further examine each predictors.

1) Fig. 9a has the most skipped pixels, whereas Fig. 9c has the least skipped pixels.
2) Fig. 9a has the least embedded pixels, whereas Fig. 9c has the most embedded pixels.
3) Fig. 9a has almost no twice embedded pixels, whereas for Figs. 9b and c, a few trails of twice embedded pixels can be observed near the head of the airplane.

In summary, predictors that incorporate more neighboring information have more embedded and twice embedded pixels, whereas predictors that incorporate least neighboring information have more skipped pixels.

B. Notes on skipped pixels

Skipped pixels are the main advantage of skewed histogram shifting because they do not cause distortion.



(a) Predictor 1* (b) Predictor 2* (c) Predictor 3*

Fig. 9: F16 embedded using skewed histogram shifting without using Eq. (9). Blue pixels are embedded, red pixels are shifted, green pixels are twice embedded, and gray-scaled pixels are skipped.

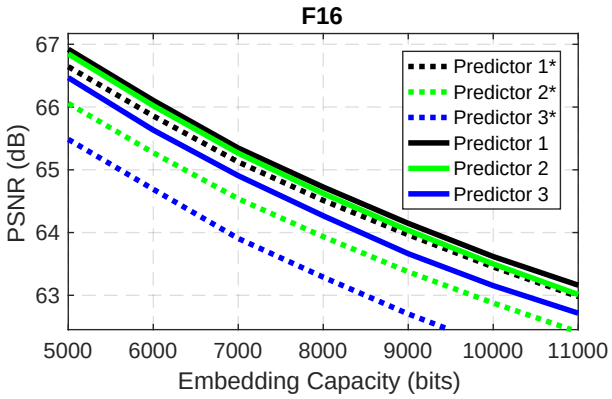


Fig. 10: Improved performance due to Eq. (9). '*' denotes the results without using Eq. (9).

The conditions for a pixel to be skipped are as follows:

1) Skipped during PHS
   implies $p - \hat{p}_h < 0$
   then $p < \hat{p}_h$
2) Skipped during NHS,
   implies $p' = p$ *and* $p' - \hat{p}_l > 0$
   then $\hat{p}_l < p'$

From conditions 1) and 2), a pixel is skipped if $\hat{p}_l < p < \hat{p}_h$ is established, *i.e.,* the target pixel value is strictly between the predicted extreme values. To summarize, skipped pixels occur in cases where pixels are difficult to predict.

C. Notes on twice embedded pixels and Eq. (9)

Twice embedded pixels are the main drawback to skewed histogram shifting. Although these pixels embed two bits at the cost of one or zero distortion, they decrease the total embedding capacity. This is because of the part of the condition of twice embedded hinges on the equality between the high estimate and the low estimate ($\hat{p}_h = \hat{p}_l$), which results in double embedding for prediction errors equal to 0. The conditions for a pixel to be twice embedded are as follows:

1) Embedded during PHS
   implies $p - \hat{p}_h = 0$
   then $p = \hat{p}_h$
2) Embedded during NHS
   implies $p' - \hat{p}_l = 0$
   then $p' = \hat{p}_l$

From conditions 1) and 2), a pixel is twice embedded if $\hat{p}_h = p = p' = \hat{p}_l$, *i.e.,* high estimate, target pixel, first watermarked pixel, and low estimate are all the same.

The aforementioned Eq. (9) is used to eliminate the chance of double embedding. In the case of $\hat{p}_h = \hat{p}_l$, Eq. (9) artificially decreases $\hat{p}_l$ by -1 such that embedding is possible when $p - \hat{p}_h = 0$ and $p' - \hat{p}_l - 1 = 0$ (here, $\hat{p}_l$ is the original value without Eq. (9). Upon close inspection, this modification is equivalent to embedding in not just 0, but also in -1 when $\hat{p}_h = \hat{p}_l$ is established

Fig 10 shows the performance gain when Eq. (9) is used for image F16, and '*' indicates the results without using Eq. (9). Significant improvement ranging from 0.25 to 1 dB can be observed. Notice that predictors that incorporate more neighboring information end up with the most improvement. This is because when more neighbors are incorporated, it is more likely that $\hat{p}_h = \hat{p}_l$ cases occur.

Another point to be made is that the improvement is more noticeable for embedding in smooth images such as image F16, as $\hat{p}_h = \hat{p}_l$ cases occur most often for smooth areas.

VII. Future work in reversible data hiding with image content analysis

Image content analysis is becoming a hot topic, as it provides an efficient way to search images using texts and match similar images based on its content [61]. Since it is becoming popular, we want to share how the proposed method and other reversible data hiding techniques can be applicable in image content analysis.

First, we want to discuss the performance impact when image search techniques are used on watermarked images. Existing image search techniques based on probabilistic image hashing, such as locality-sensitive hashing [62] and Grassman hashing [63, 64], hash the image such that there is a higher chance of collision for similar looking images than the ones that are not. Therefore, smaller distortion causing reversible data hiding methods are less likely to affect the search performance than the ones that cause larger distortion, *i.e.,* PSNR will be a good estimator for determining whether a particular technique will affect the image search performance. Furthermore, since the proposed method prioritizes embedding in smooth areas before complex areas, an image hashing technique that is trained to be robust against noise in the smooth areas will have smaller impact on the performance.

The proposed method can also be used to embed the metadata resulting from image content analysis directly into the image. When implemented in this way, it provides an additional protection against accidental deletion and corruption of metadata. In addition to the protection, the embedded metadata can be compared with the original metadata for integrity verification.

VIII. Conclusion

In this paper, we propose a novel reversible data hiding scheme that uses skewed histograms with a pair of extreme predictions. Unlike the traditional prediction error histogram shifting, where each pixel is predicted once using a good predictor, the proposed method predicts each pixel twice using a pair of extreme prediction values to produce two skewed histograms, where their asymmetry is used to reduce distortion. Analysis shows that the proposed method performs well. Further research on the topic should bring interesting results.

IX. Acknowledgment

The authors would like to thank all the anonymous reviewers for their constructive feedback to improve the paper, especially the suggestion to discuss the application of the proposed method in image content analysis.

REFERENCES

[1] K.-L. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 553–562, 2013.

[2] H.-T. Wu, J.-L. Dugelay, and Y.-Q. Shi, "Reversible image data hiding with contrast enhancement," *IEEE Signal Processing Letters*, vol. 22, no. 1, pp. 81–85, 2015.

[3] G. Gao and Y.-Q. Shi, "Reversible data hiding using controlled contrast enhancement and integer wavelet transform," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 2078–2082, 2015.

[4] H.-T. Wu, J. Huang, and Y.-Q. Shi, "A reversible data hiding method with contrast enhancement for medical images," *Journal of Visual Communication and Image Representation*, vol. 31, pp. 146–153, 2015.

[5] S. Kim, R. Lussi, X. Qu, and H. J. Kim, "Automatic contrast enhancement using reversible data hiding," in *IEEE International Workshop on Information Forensics and Security*, 2015, pp. 1–5.

[6] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989–999, 2009.

[7] J. Fridrich, M. Goljan, and R. Du, "Lossless data embeddingnew paradigm in digital watermarking," *EURASIP Journal on Advances in Signal Processing*, no. 2, pp. 185–196, 2002.

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

[9] M. U. Celik, G. Sharma, and A. M. Tekalp, "Lossless watermarking for image authentication: a new framework and an implementation," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 1042–1049, 2006.

[10] J. Fridrich, M. Goljan, and R. Du, "Invertible authentication," in *SPIE Security and Watermarking of Multimedia Contents III*, 2001, pp. 197–208.

[11] G. Xuan, J. Zhu, J. Chen, Y. Q. Shi, Z. Ni, and W. Su, "Distortionless data hiding based on integer wavelet transform," *Electronics Letters*, vol. 38, no. 25, pp. 1646–1648, 2002.

[12] H. J. Hwang, H. J. Kim, V. Sachnev, and S. H. Joo, "Reversible watermarking method using optimal histogram pair shifting based on prediction and sorting," *KSII Transactions on Internet and Information Systems*, vol. 4, no. 4, pp. 655–670, 2010.

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

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

[15] L. Kamstra and H. J. Heijmans, "Reversible data embedding into images using wavelet techniques and sorting," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2082–2090, 2005.

[16] H. J. Kim, V. Sachnev, Y. Q. Shi, J. Nam, and H.-G. Choo, "A novel difference expansion transform for reversible data embedding," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 456–465, 2008.

[17] I.-C. Dragoi and D. Coltuc, "Adaptive pairing reversible watermarking," *IEEE Transactions on Image Processing*, vol. 25, no. 5, pp. 2420–2422, 2016.

[18] W.-L. Tai, C.-M. Yeh, and C.-C. Chang, "Reversible data hiding based on histogram modification of pixel differences," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 6, pp. 906–910, 2009.

[19] G. Coatrieux, W. Pan, N. Cuppens-Boulahia, F. Cuppens, and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 111–120, 2013.

[20] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

[21] S.-K. Lee, Y.-H. Suh, and Y.-S. Ho, "Reversiblee image authentication based on watermarking," in *IEEE International Conference on Multimedia and Expo*, 2006, pp. 1321–1324.

[22] J. Hwang, J. Kim, and J. Choi, "A reversible watermarking based on histogram shifting," in *Lecture Notes in Computer Sciences*, 2006, vol. 4283, pp. 348–361.

[23] B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5010–5021, Dec 2013.

[24] X. Li, W. Zhang, X. Gui, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 7, pp. 1091–1100, July 2013.

[25] B. Ou, X. Li, Y. Zhao, and R. Ni, "Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion," *Signal Processing: Image Communication*, vol. 29, no. 27, pp. 760–772, 2014.

[26] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Processing*, vol. 93, no. 1, pp. 198–205, 2013.

[27] F. Peng, X. Li, and B. Yang, "Improved pvo-based reversible data hiding," *Digital Signal Processing*, vol. 25, pp. 255–265, 2014.

[28] X. Qu and H. J. Kim, "Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding," *Signal Processing*, vol. 111, pp. 249–260, 2015.

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

[30] M. Fallahpour, "Reversible image data hiding based on gradient adjusted prediction," *IEICE Electronics Express*, vol. 5, no. 20, pp. 870–876, 2008.

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

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

[33] W. Hong, "An efficient prediction-and-shifting embedding technique for high quality reversible data hiding," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, pp. 4:1–4:12, 2010.

[34] G. Xuan, Y. Q. Shi, J. Teng, X. Tong, and P. Chai, "Double-threshold reversible data hiding," in *IEEE International Symposium on Circuits and Systems*, 2010, pp. 1129–1132.

[35] X. Gao, L. An, Y. Yuan, D. Tao, and X. Li, "Lossless data embedding using generalized statistical quantity histogram," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 8, pp. 1061–1070, 2011.

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

[37] D. Coltuc, "Improved embedding for prediction-based reversible watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 873–882, 2011.

[38] C. Dragoi and D. Coltuc, "Improved rhombus interpolation for reversible watermarking by difference expansion," in *IEEE Signal Processing Conference*, 2012, pp. 1688–1692.

[39] H.-T. Wu and J. Huang, "Reversible image watermarking on prediction errors by efficient histogram modification," *Signal Processing*, vol. 92, no. 12, pp. 3000–3009, 2012.

[40] C. Qin, C.-C. Chang, Y.-H. Huang, and L.-T. Liao, "An inpainting-assisted reversible steganographic scheme using a histogram shifting mechanism," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 7, pp. 1109–1118, 2013.

[41] D. Coltuc and I.-C. Dragoi, "Context embedding for raster-scan rhombus based reversible watermarking," in *ACM Workshop on Information Hiding and Multimedia Security*, 2013, pp. 215–220.

[42] I.-C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1779–1790, April 2014.

[43] S. Lee, C. D. Yoo, and T. Kalker, "Reversible image watermarking based on integer-to-integer wavelet transform," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 321–330, 2007.

[44] D. Coltuc and J.-M. Chassery, "Very fast watermarking by reversible contrast mapping," *IEEE Signal Processing Letters*, vol. 14, no. 4, pp. 255–258, 2007.

[45] S. Weng, Y. Zhao, J.-S. Pan, and R. Ni, "Reversible watermarking based on invariability and adjustment on pixel pairs," *IEEE Signal Processing Letters*, vol. 15, pp. 721–724, 2008.

[46] C. Wang, X. Li, and B. Yang, "Efficient reversible image watermarking by using dynamical prediction-error expansion," in *IEEE International Conference on Image Processing*, 2010, pp. 3673–3676.

[47] D. Coltuc, "Low distortion transform for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 412–417, 2012.

[48] F. Peng, X. Li, and B. Yang, "Adaptive reversible data hiding scheme based on integer transform," *Signal Processing*, vol. 92, no. 1, pp. 54–62, 2012.

[49] X. Gui, X. Li, and B. Yang, "A novel integer transform for efficient reversible watermarking," in *IEEE International Conference on Pattern Recognition*, 2012, pp. 947–950.

[50] X. Qu, S. Kim, and H. J. Kim, "Reversible watermarking based on compensation," *Journal of Electrical Engineering & Technology*, vol. 10, no. 1, pp. 422–428, 2015.

[51] V. Sachnev, H. Kim, S. Suresh, and Y. Shi, "Reversible watermarking algorithm with distortion compensation," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 1, p. 316820, 2011.

[52] J. Wang and J. Ni, "A ga optimization approach to hs based multiple reversible data hiding," in *IEEE International Workshop on Information Forensics and Security*, 2013, pp. 203–208.

[53] I. Caciula and D. Coltuc, "Improved control for low bit-rate reversible watermarking," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 7425–7429.

[54] L. Dong, J. Zhou, Y. Y. Tang, and X. Liu, "Estimation of capacity parameters for dynamic histogram shifting (dhs)-based reversible image watermarking," in *IEEE International Conference on Multimedia and Expo*, 2014, pp. 1–6.

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

[56] I.-C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1779–1790, 2014.

[57] B. Y. Lee, H. J. Hwang, and H. J. Kim, "Reversible data hiding using a piecewise autoregressive predictor based on two-stage embedding," *Journal of Electrical Engineering and Technology*, vol. 11, 2016.

[58] H. J. Hwang, S. Kim, and H. J. Kim, "Reversible data hiding using least square predictor via the lasso," *EURASIP Journal on Image and Video Processing*, vol. 2016, no. 1, p. 42, 2016.

[59] X. Gui, X. Li, and B. Yang, "Efficient reversible data hiding based on two-dimensional pixel-intensity-histogram modification," in *IEEE International Confer-*

ence on Acoustics, Speech and Signal Processing, 2014, pp. 7420–7424.

[60] X. Chen, X. Sun, H. Sun, L. Xiang, and B. Yang, "Histogram shifting based reversible data hiding method using directed-prediction scheme," *Multimedia Tools and Applications*, vol. 74, no. 15, pp. 5747–5765, 2015.

[61] G. Hua and Q. Tian, "What can visual content analysis do for text based image search?" in *IEEE International Conference on Multimedia and Expo*, 2009, pp. 1480–1483.

[62] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the ACM Annual Symposium on Computational Geometry*, 2004, pp. 253–262.

[63] X. Wang, Z. Li, and D. Tao, "Subspaces indexing model on grassmann manifold for image search," *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2627–2635, 2011.

[64] X. Wang, Z. Li, L. Zhang, and J. Yuan, "Grassmann hashing for approximate nearest neighbor search in high dimensional space," in *IEEE International Conference on Multimedia and Expo*, 2011, pp. 1–6.

Suah Kim Suah Kim (S'15) received the Bachelor of Mathematics from University of Waterloo in 2012, and is working towards the Ph.D degree from Korea University. He was a visiting young scientist at Sun Yat-sen University in 2016. His research interests include reversible data hiding, and image forensics.

Xiaochao Qu Xiaochao Qu is currently a senior engineer at MTlab (Meitu, Inc). He received the PhD degree from the Department of Information Security and Management at Korea University in 2015. Before that, he revieved the BS degree from the Department of Computer Science and Technology in 2009. He mainly focuses on the research topics in the area of Computer Vision, particularly object detection, semantic segmentation, human pose estimation and deep learning-related problems.

Vasily Sachnev Vasily Sachnev received his B.S and M.s degrees in electrical engineering from the Komsomolsk-na-Amure State Technical University, Russia, in 2002 and 2004, PhD degree in Multimedia Security Laboratory at the Center of Information Security and Technology (CIST), Graduate School of Information Management and Security, Korea University, Seoul, Korea at 2009. Since 2010, he is a faculty memeber in the Catholic University of Korea. His research interests include machine learning, digital watermarking, steganography, and image processing.

Hyoung Joong Kim Hyoung Joong Kim (M'09) received the BS, MS and PhD degrees from Seoul National University in 1978, 1986, and 1989, respectively. He was a Professor of Kangwon National University from 1989 to 2006. He is currently a Professor with Korea University. His research interests include distributed computing, machine learning, multimedia security, and cryptocurrency.