# Data Hiding in Secured Images Using Reversible Image Transformation

*B.Dinesh*[1] *A.Karna Rao*[2] *V.Srinivas*[3]

[1]*P.G.Scholar, Department of ECE, Swarnandhra Institute of Engineering Technology, Narasapuram, Andhra Pradesh*

[2,3] *Department of ECE, Swarnandhra Institute of Engineering Technology, Narasapuram, Andhra Pradesh*

**b.dinesh777@gmail.com**

***Abstract*—**. *A texture synthesis process re-samples a smaller texture image which synthesizes a new texture image with a similar local appearance and arbitrary size. This concept proposes a novel framework for RDH-EI based on reversible image transformation (RIT). Different from all previous encryption-based frameworks, in which the cipher texts may attract the notation of the curiouscloud, RIT-based framework allows the user to transform the content of original image into the content of another targetimage with the same size. We weave the texture synthesis process into steganography to conceal secret messages. In contrast to using an existing cover image to hide messages, our algorithm conceals the source texture image and embeds secret messages through the process of texture synthesis. This allows us to extract secret messages and the source texture from a stego synthetic texture. Our approach offers three distinct advantages. First, our scheme offers the embedding capacity that is proportional to the size of the stego texture image. Second, a steganalytic algorithm is not likely to defeat our steganographic approach. Third, the reversible capability inherited from our scheme provides functionality which allows recovery of the source texture. Images along with text hiding is a main enhancement for this concept.*

***Index Terms*—reversible data hiding, image encryption and decryption, reversible image transformation, privacy protection.**

## I. INTRODUCTION

Now a days outsourced storage by cloud becomes a more and more popular service, especially for multimedia files, such images or videos, which need large storage space. To manage the outsourced images, the cloud server may embed some additional data into the images, such as image category and notation information, and use such data to identify the ownership [1] or verify the integrity of images. Obviously, the cloud service provider has no right to introduce permanent distortion during data embedding into the outsourced images. Therefore, reversible data hiding (RDH) technology is needed, by which the original image can be losslessly recovered after the embedded message is extracted. This technique is also widely used in medical imagery [2], military imagery and law forensics, where no distortion of the original cover is allowed.

So far, many RDH methods on images have been proposed. In essence, all these methods can be viewed as a process of semantic lossless compression [3], [4], in which some space is saved for embedding extra data by losslessly compressing the image. Herein, “semantic compression” means that the compressed image should be close to the original image, and thus one can get a marked image with good visual quality. Because the residual part of images, e.g., the prediction errors (PE), has small entropy and can be easily compressed, almost all recent RDH methods first generate PEs as the host sequence [5]–[7], and then reversibly embed the message into the host sequence by modifying its histogram with methods like histogram shifting (HS) [8] or difference expansion (DE) [9]. Recently, Zhang et al. proposed the optimal histogram modification algorithm [4], [10] for RDH by estimating the optimal modification probability [11], [12].

On the other hand, cloud service for outsourced storage makes it challenging to protect the privacy of image contents. For instance, recently many private photos of Hollywood actress leaked from iCloud [13]. Although RDH is helpful for managing the outsourced images, it cannot protect the image content. Encryption is the most popular technique for protecting privacy. So it is interesting to implement RDH in encrypted images (RDH-EI), by which the cloud server can reversibly embed data into the image but can not get any knowledge about the image contents. Inspired by the needs of privacy protection, many methods have been presented to extend RDH methods to encryption domain. From the viewpoint of compression, these methods on RDH-EI belong to the next two frameworks [14]: Framework I “vacating room after encryption (VRAE)” and Framework II “reserving room before encryption (RRBE) ”.

In the framework “vacating room after encryption (VRAE)”, the cloud server embeds data by losslessly vacating room from the encrypted images by using the idea of compressing encrypted images [15], [16]. Compression of encrypted data can be formulated as source coding with side information at the decoder [15]. Usually the side information is the correlation of plaintexts that is exploited for decompression by the decoder. In [17], Zhang divided the encrypted image into several blocks. By flipping 3 LSBs (Least Significant Bits) of the half of pixels in each block, room can be vacated for the embedded bit. The data extraction and image recovery proceed by finding which part has been flipped in one block. This process can be realized with the help of spatial correlation in the decrypted image. Hong et al. [18] ameliorated Zhang's method at the decoder side by further exploiting the spatial correlation using a different estimation equation and side match technique. For both methods in [17] and [18], decrypting image and extracting data must be jointly executed. Recently, Zhou et al. [19] proposed a novel RDH-EI method for joint decryption and extraction, in which the correlation of plaintexts is further exploited by distinguishing the encrypted and non-encrypted pixel blocks with a two-class SVM classifier. To separate the data extraction from image decryption, Zhang [20] emptied out space for data embedding by directly using the typical manner of ciphertext compression, that is, compressing the encrypted pixels in a lossless manner by using the syndromes of parity-check matrix of channel codes. Qian et al. [21] improved the method of [20] by adopting LDPC (low density parity check) based Slepian-Wolf encoder which is also one of the most efficient methods for ciphertext compression.

In the framework "reserving room before encryption (R-RBE)", the image owner first empties out room by using RDH method in the plain images. After that, the image is encrypted and outsourced to the cloud and the cloud server can freely embed data into the reserved room of the encrypted image. The first method under RRBE framework was presented in [14], which reserves room by embedding LSBs of some pixels into other pixels with a traditional RDH method and then encrypts the image, so the positions of these LSBs in the encrypted image can be used to embed data. The method in [14] implies that the purpose of RDH in encrypted image can also be realized by RDH for plaintext images. Following this idea, Zhang et al. [22] reserve room in images by generating prediction errors (PE) and modifying the histogram of PE, which is the most popular technique used in RDH for plaintext images. To protect confidentiality, a special encryption scheme is designed in [22] to encrypt the PEs. Cao et al. [23] improved the methods of [14], [22] by patch-level sparse representation which can yield PEs with smaller entropy and thus result in a large hiding room.

For both frameworks, VRAE and RRBE, the image owner will send a ciphertext-formed image to the cloud. However, the ciphertexts with the special form of messy codes are easy to cause the attention of the cloud server who may try to dig out information on the encryption users. In fact, the cloud server is assumed to be curious to collect information from the outsourced files [24], and obviously the encrypted images are more attractive to a curious cloud server. Therefore, the fact, that the user is outsourcing encrypted images, itself is also a kind of privacy of the user, which should be protected.

In this paper, we propose a novel framework for RDH-EI by using reversible image transformation (RIT). RIT transfers the semantic (content) of the original image $I$ into the semantic of another image $J$, and "reversibility" means that $I$ can be losslessly restored from the transformed image. Therefore RIT can be viewed as a special encryption scheme, called "Semantic Transfer Encryption (STE)". In other words, the resultant transformed image which is also the encrypted image $E(I)$ will look similar with $J$. The image $J$ is selected to be irrelevant with $I$ but has the same size of $I$, and thus the content of the image $I$ is protected. Because the "encrypted image" is in a form of plaintext, it will avoid the notation of the cloud server, and the cloud server can easily embed data into the "encrypted image" with traditional RDH methods for plaintext images.

The rest of the paper is organized as follows. In Sec-tion II, we compare the RIT-based framework with previous frameworks and summarize the main contributions of the novel framework. A method of RIT is elaborated in Section III, and two kinds of RDH methods on transformed images are proposed in Section IV. The paper is concluded with a discussion in Section V.

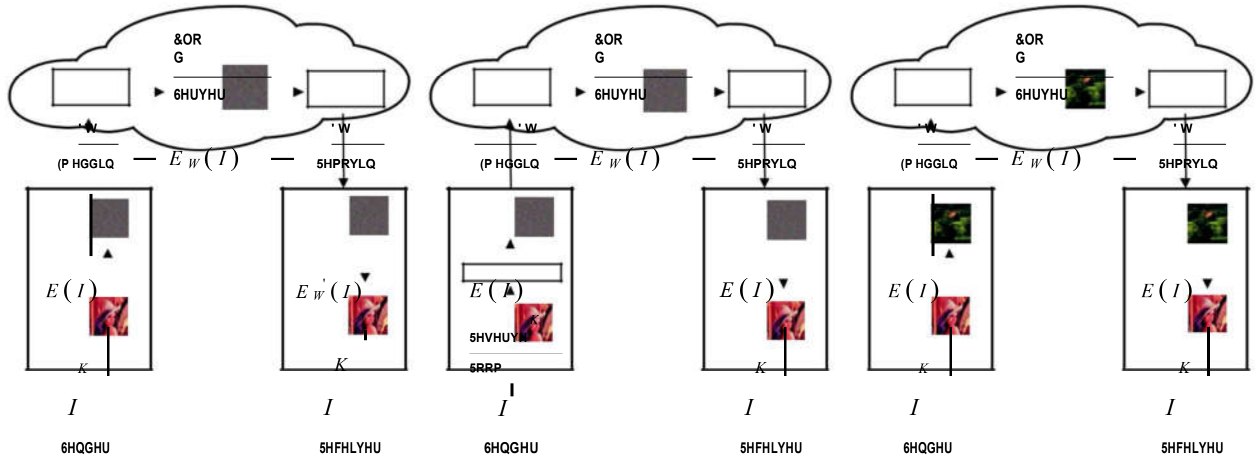
## II. Comparison between three frameworks

Fig. 1 depicts the differences between the the novel frame-work and previous frameworks, which shows that, by frame-works VRAE and RRBE, the user's images are stored in the form of ciphertext in the cloud account, while by the the RIT-based framework, the image is stored in a form of plaintext.

In the framework VRAE shown in Fig.1(a), such as schemes in [17] and [18], the image owner (the sender) encrypts the image $I$ into $E(I)$ with a key $K$. The cloud server embeds data by compressing the encrypted image $E(I)$ and generates $E_w(I)$ that is stored in the cloud. When getting a retrieval request, the cloud server returns $E_w'(I)$ to the recevier, maybe an authorized third party, who generates $I$ through a process of joint decompression and decryption with the key $K$. Herein, $E_w'(I)$ may be just $E_w(I)$ or a modified version obtained by removing the embedded data. Note that the cloud server cannot restore $E(I)$ from $E_w(I)$, since decompression should be joined with decryption with the help of $K$. In this framework, the complexity is taken on by the receiver who must join the process of decompression and decryption to get the original image. In other words, the compression-based RDH method used by the cloud server should be specified together with the receiver, i.e., the RDH method is receiver-related.

In the framework RRBE shown in Fig.1(b), such as schemes in [14], [22], the image owner (the sender) reserves room from the image $I$ and encrypts it into $E(I)$ with a key $K$, and then sends it to the cloud server who embeds data into the reserved room and generates $E_w(I)$. $E_w(I)$ is stored in the cloud, from which the cloud server can extract the data that is used for management. When an authorized user (the receiver) wants to retrieve the image, the cloud server can restore $E(I)$ from $E_w(I)$ and send $E(I)$ to the user who can decrypt $E(I)$ and get $I$ with the key $K$. In the framework RRBE, the complexity is borne by the sender who should reserve room for RDH by exploiting the redundancy within the image and thus the RDH method used by the cloud should be specified with the sender, that is, the RDH method used by cloud is sender-related.

In the RIT based framework depicted in Fig.1(c), the image $I$ is "encrypted" into another plaintext image $E(I)$ with a key $K$, so all images of the users, encrypted or not, will be stored in the cloud in the form of plaintexts. The cloud server can embed/extract data into/from $E(I)$ with any classical RDH method for plaintext images. And $E(I)$ can be recovered from the watermarked image $E_w(I)$ by the cloud and sent back to the authorized user who anti-transforms it to get the original image $I$ with the key $K$. The main contributions of this novel framework include:



(a) Framework VRAE (b) Framework RRBE (c) RIT-based Framework

Fig. 1. Comparison between three frameworks of RDH-EI.

- The idea of RIT is exploited for RDH-EI, by which the user can outsource the encrypted image to the cloud in a form of plaintext and thus it will avoid the attention of the curious cloud.
- In the RIT based framework, the cloud server can easily embed data into the "encrypted image" by arbitrarily selecting RDH methods for plaintext images such as those in [4], [6], [7], [10]. In other words, the RDH used by the cloud is irrelevant with both the sender and receiver, which is called a client-free RDH-EI scheme by us. "Client free" is important for the scenarios of public clouds, in which it is hard for the cloud server to ask the clients how to encrypt or decrypt their data, because the cloud is thought to be only semi-honest [24].

## III. An Example on Reversible Image Transformation

In this section, we propose a method of RIT to encrypt spatial images, which is inspired by the technique of image transformation proposed by Lee et al. [26]. Lee et al.'s method can transform the original image to a freely-selected target image with the same size, yielding a secret-fragment-visible mosaic image defined in [25]. But the original image cannot be restored in a lossless way. It is not reversible, so it is not suitable for the scenario of RDH-EI. We will modify Lee et al.'s method to be reversible and obtain an encrypted image which looks like the target image.

For color images, we transform the color channel R, G, and B respectively in the same manner. So we just take gray images (one channel) as an example to describe the method. For an original image $I$, we randomly select a target image $J$ having the same size with $I$ from an image database.

Firstly, we divide the original image $I$ and the target image $J$ into $N$ non-overlapping blocks respectively, and then pair the blocks of $I$ and $J$ as a sequence such that $(B_1, T_1), \ldots, (B_N, T_N)$, where $B_i$ is an original block of $I$ and $T_i$ is the corresponding target block of $J$, $1 \leq i \leq N$. We will transform $B_i$ toward $T_i$ and generate a $T_i'$ similar to $T_i$. After that, we replace each $T_i$ with $T_i'$ in the target image $J$ to get the transformed image $J'$. Finally we embed some accessorial information into $J'$ with an RDH method and generate the ultimate "encrypted image" $E(I)$. These accessorial information is necessary for recovering $I$ from $J'$. Before being embedded, these accessorial information will be compressed and encrypted with a key $K$ shared with the receiver, so only a receiver having $K$ can decrypt $E(I)$.

The proposed transformation process consists of three steps: block pairing, block transformation and accessorial information embedding. We will mainly elaborate the first two steps in the subsections and the third step can be implemented by any traditional RDH method.

### A. Block Pairing

To make the transformed image $J'$ look like target image $J$, we hope, after transformation, each transformed block will have close mean and standard deviation (SD) with the target block. So we first compute the mean and SD of each block of $I$ and $J$ respectively. Let a block $B$ be a set of pixels such that $B = \{p_1, p_2, \cdots, p_n\}$, and then the mean and SD of this block is calculated as follows.

$$u = \frac{1}{n}\sum_{i=1}^{n} p_i. \tag{1}$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - u)^2}. \tag{2}$$

When matching blocks between original image and target image, we hope two blocks with closest SDs to be a pair. In Lee et al.'s method, the blocks of original image and target image are sorted in ascending order according to their SDs respectively, and then each original block is paired up with a corresponding target tile in turn according to the order. To recover the original image from the transformed image, the positions of the original blocks should be recorded and embedded into the transformed image with an RDH method. If the image is divided into $N$ blocks, $N\lceil \log N \rceil$ bits are needed to record block indexes. Obviously, the smaller the block size is, the better the quality of transformed image will be, but which will result in a large $N$. Therefore, the amount of information used to record the index for each block may be these information into the transformed image. In fact there may not exist enough redundant space to store these additional information. For instance, if we divide a $1024 \times 1024$ image into $4 \times 4$ blocks, $2^{16} \times 16$ bits are needed to record the positions of blocks.

To compress the block indexes, we first classify the blocks according to their SD values before pairing them up. In fact, we found that the SD values of most blocks concentrate in a small range close to zero and the frequency quickly drops down with the increase of the SD value as displayed in Fig. 2, which is depicted from various sizes of 10000 images from the BossBase image database [27]. Therefore, we divide the blocks into two classes with unequal proportions: class 0 for blocks with smaller SDs, and class 1 for blocks with larger SDs, and pair up the blocks belonging to the same class. By assigning the majority of blocks to the class 0, we can avoid the large deviation of SDs between a pair of blocks and efficiently compress the indexes at the same time.

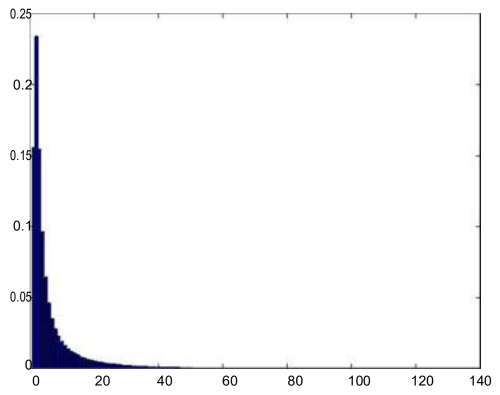


Fig. 2. The distribution of SDs of $4 \times 4$ block for various sizes of natural images

In the present paper, we propose to divide both the original and target images into non-overlapping $4 \times 4$ blocks and calculate the SDs of each block. We first divide the blocks of original image $I$ into 2 classes according to the quantile of SDs. Denote that the $\%\alpha$ quantile of SDs by $N_\alpha$. We assign the blocks with SDs $\in [0, N_\alpha]$ to "Class 0", and blocks with SDs $\in (N_\alpha, N_{100}]$ to "Class 1". And then we will scan the blocks in the raster order, i.e., from left to right and from top to bottom, and assign a class label, 0 or 1, to each block.

Next, we label the blocks of target image based on the classes' volumes of original image. Assuming that the ith class in the original image includes $n_i$ blocks for $i = 0$ or $1$, we scan the target image in the raster order, and label the first $n_0$ blocks with the smallest SDs as Class 0, and the rest $n_1$ blocks as Class 1. As a result, each class in the target image includes the same number of blocks as the corresponding class in the original image. We pair the original block up with target block in the following manner. Scan the original image and target image in raster order respectively and pair the jth block of the class $i$ in the original image up with the jth block of the class $i$ in the target image for $i = 0, 1$ and $j = 1, \ldots, n$

A simple example on the proposed block pairing method is shown in Fig. 3, in which the image only consists of 10 blocks. By setting $\alpha = 70$, we assign 7 blocks with smallest SDs into class 0, and the rest 3 blocks into class 1 in the original image. In the target image, although the 8th and 9th block have the same SD value 5, the 8th block is assigned to class 0 but the 9th block is assigned to the class 1, because class 0 can only include 7 blocks as determined by the class 0 of the original image. After labeling the class indexes, we get a class index table (CIT) for original image and target image respectively, which will be helpful for understanding the procedure of block pairing.

According to the pairing rule, the first block of the original image is paired up with the forth block of the target image, because both of them is the first block of class 1 as shown in the CIT; the second block of original image is paired up with the ninth block of target image, because both of them is the second block of class 1, and so on. The pairing result is listed in Table I, which can be generated according to the CIT of original image and the CIT of the target image.

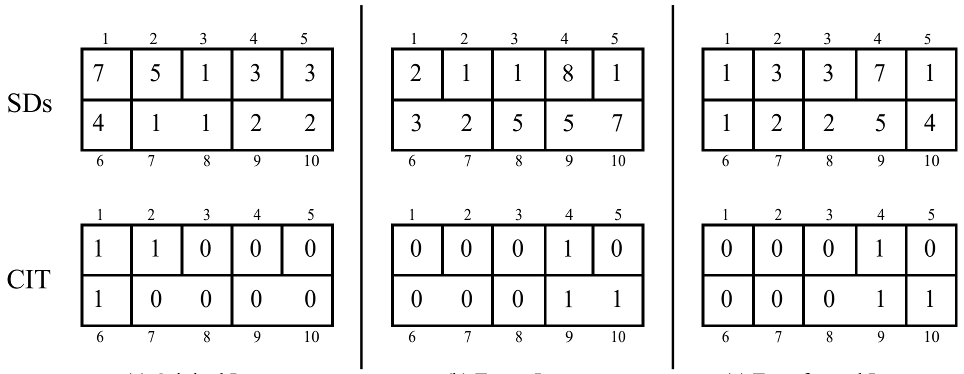
For each pair of blocks $(B, T)$, as we will see in the next section, the original block $B$ will be transformed to target block $T$ by mean shifting and block rotation, yielding $T'$. By replacing each $T$ with $T'$ in the target image, the sender will generate the transformed image. Note that both operations of mean shifting and block rotation will not change the SD value, so $T'$ has the same SD as $B$. Therefore, the SDs in transformed image is only a permutation of those in original image. When classifying the blocks of transformed image according to $\%\alpha$ quantile of SDs, the receiver can get a CIT that is same with the CIT of target image as shown in Fig. (b) and Fig. (c) in Fig. 3.

Therefore, to restore the original image from the transformed image, the receiver only needs to know the CIT of the original image. In fact, by CIT of original image and the CIT of transformed image (which is also the CIT of target image), the receiver can reconstruct Table I perfectly. Then according to the table he will know how to rearrange the transformed blocks to restore the original blocks. In the example of Fig. 3, the first block of the transformed image should be put back to position 3, and the second block should be put back to position 4 as indicated in Table I.

Note that CIT can be efficiently compressed because the ratio of 0 and 1 is bias. If the image is divided into $N$ blocks, and these blocks are divided into two classes with $\%\alpha$ quantile of SDs, we need $N \cdot H(\alpha/100)$ bits to record $S$, where $H$ is the binary entropy function. For instance, if we set $\alpha = 75$ and divide a $1024 \times 1024$ image into $4 \times 4$ blocks, we only need $2^{16} \times H(0.75) \approx 2^{16} \times 0.81$ bits to record the positions of blocks, which is much less than $2^{16} \times 16$ bits used by the method in [26]. The compressed CIT will be encrypted and embedded into the transformed image as a part of accessorial information (AI).

## B. Block Transformation

By the block pairing method described above, in each pair $(B, T)$, the two blocks have close SD values. Therefore, when



(a) Original Image (b) Target Image (c) Transformed Image

Fig. 3. An example of block pairing.

TABLE I
BLOCK PAI RI NG RES ULT OF THE EXAM P LE I N FI G. 3

| Block index of original image | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Block index of target image | 4 | 9 | 1 | 2 | 3 | 10 | 5 | 6 | 7 | 8 |

transforming B towards T, we only need a mean shifting transformation that is reversible. However, the transformation used in Lee et al.'s method [26] is not reversible because it changes the mean and SD at the same time.

Let the original block $B = \{p_1, p_2, \cdots, p_n\}$, and the corresponding target block $T = \{p_1, p_2, \cdots, p_n\}$. With Eq. (1), we calculate the means of B and T and denote them by $u_B$ and $u_T$ respectively.

The transformed block $T = \{p_1, p_2, \cdots, p_n\}$ is generated by the mean shifting as follows.

$$p''_i = p_i + u_T - u_B, \quad (3)$$

where $(u_T - u_B)$ is the difference between the means of target block and original block. We want to shift each pixel value of original block by amplitude $(u_T - u_B)$ and thus the transformed block has the same mean with the corresponding target block. However, because the pixel value $p''_i$ should be an integer, to keep the transformation reversible, we round the difference to be the closest integer as Eq. (4)

$$u = \mathrm{round}(u_T - u_B), \quad (4)$$

and shift the pixel value by u, namely, each $p''_i$ is gotten by

$$p_i = p_i + u, \quad (5)$$

Note that the pixel value $p''_i$ should be an integer between 0 and 255, so the transformation (5) may result in some over-flow/underflow pixel values. To avoid such transformed blocks abstained by Eq. (5), we assume that the maximum overflow pixel value is $OV_{max}$ for $u \geq 0$ or the minimum underflow pixel value is $UN_{min}$ for $u < 0$. If overflow/underflow occurs in some blocks, we eliminate them by modifying u

$$u + 255 - OV_{max}, \quad \text{if} \quad u \geq 0$$

We use the modified u to shift the pixels of block B, and thus all the pixels' values are controlled into the range of [0, 255]. However the range of u's value is still very large, which cannot be efficiently compressed. Thus we further modify u as

$$u = \begin{cases} \lambda \times \mathrm{round}(\frac{u}{\lambda}), & \text{if} \quad u \geq 0 \\ \lambda \times \mathrm{floor}(\frac{u}{\lambda}) + \frac{\lambda}{2}, & \text{if} \quad u < 0 \end{cases}, \quad (7)$$

in which the quantization step, λ, is an even parameter. Then it just needs to record $u' = 2|u|/\lambda$, by which it has the advantage of not to record the sign of u. Because when $u'$ is an even number it means $u \geq 0$ and when $u'$ is an odd number it means $u < 0$. Since when λ is large the amount of information recording $u'$ will be small but the offset between the modified u and the original u will be large, a trade-off must made by choosing λ. We set λ = 8 in the following experiments.

Finally, to maintain the similarity between the transformed image and target image as much as possible, we further rotate the shifted block into one of the four directions $0^o$, $90^o$, $180^o$ or $270^o$. The optimal direction is chosen for minimizing the root mean square error (RMSE) between the rotated block and the target block.

After shifting transformation and rotation, we get a new block $T'$. With these new blocks, we replace the corresponding blocks in the target image and generate the transformed image J. The parameters, u and rotation directions, will be compressed, encrypted and then embedded into the trans-formed image $J'$ as accessorial information (AI) to output the "encrypted image" E(I) called in this paper image.

The transform and anti-transformation procedures of the

**Algorithm 1** Procedure of Transformation

**Input:** An original image I and a secret key K.
**Output:** The encrypted image E(I).

1) Select a target image J having the same size as I from an image database.
2) Divide both I and J into several non-overlapping $4 \times 4$ blocks. Assuming that each image consists of N blocks, calculate the mean and SD of each block.
3) Classify the blocks with %α quantile of SDs and generate CITs for I and J respectively. Pair up blocks of I with blocks of J according the CITs as described in subsection III-A.
4) For each block pair $(B_i, T_i)$ $(1 \le i \le N)$, compute the mean difference $u_i$. Add $u_i$ to each pixel of $B_i$ and then rotate the block into the optimal direction $\theta_i$ ($\theta_i \in \{0^o, 90^o, 180^o, 270^o\}$), which yields a transformed block $T_i$.
5) In the target image J, replace each block $T_i$ with the corresponding transformed block $T_i$ for $1 \le i \le N$ and generate the transformed image J.
6) Collect $u_i$'s and $\theta_i$'s for all block pairs, and compress them together with the CIT of I. Encrypt the compressed sequence and the parameter α by a standard encryption scheme such as AES with the key K.
7) Take the encrypted sequence as accessorial information (AI), and embed AI into the transformed image $J'$ with an RDH method such as the one in [7], and output the encrypted image E(I).

**Algorithm 2** Procedure of Anti-transformation

**Input:** The encrypted image E(I) and the key K.
**Output:** The original image I.

1) Extract AI and restore the transformed image $J'$ from E(I) with the RDH scheme in [7].
2) Decrypt AI by AES scheme with the key K, and then decompress the sequence to obtain CIT of I, $u_i$, $\theta_i$ $(1 \le i \le N)$ and α.
3) Divide J into non-overlapping N blocks with size of $4 \times 4$. Calculate the SDs of blocks, and then generate the CIT of J according to the %α quantile of SDs.
4) According to the CITs of $J'$ and I, rearrange the blocks of $J'$ as described in Subsection III-A.
5) For each block $T_i$ of J for $1 \le i \le N$, rotate $T_i'$ in the anti-direction of $\theta_i$, and then subtract $u_i$ from each pixel of $T_i$, and finally output the original image I.

## C. Experimental Results on RIT

In this section experimental results on the proposed RIT method are presented. 100 pairs of images are randomly chosen as our test images from the BossBase image database [27]. Firstly all the images are preprocessed to get the same size of $1024 \times 1024$ pixels.

Since in the RIT method the parameter α has an effect on the AI payload, we give the experiment to select a better α to improve the overall performance. The result is depicted in Fig. 4. The smaller the space occupied by AI is, the better the encrypted images visual quality will be. For α in the range of [0.05,0.95], the variation of AI payload seems to be not large. And it can be seen that when α is 0.75, the AI payload reaches the valley value. So in the following experiments, α is set 0.75.

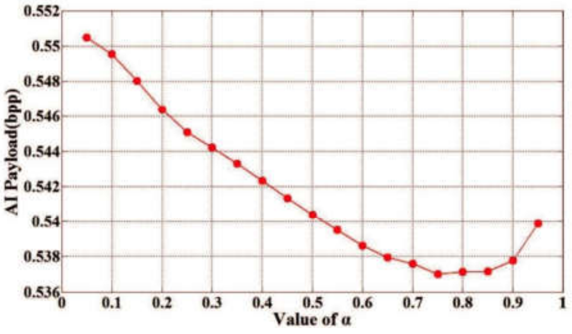


Fig. 4. The effect of AI payload for different α values.

To illustrate the visual effect of the RIT method, experimental results of five pairs of test images labeled as A, B, C, D and E are given from Fig. 5 to Fig. 8. In Experiment A, we list the "decrypted images" with the right key and the wrong key respectively. Because the original image can be losslessly restored with the correct key, we did not list the "decrypted images" in the rest experiments. We also list the marked images with RDH in experiment D and E, which will be further discussed in the next section.

The encrypted images E(I) obtained by RIT look like mosaic images with their appearance similar to the target images. Since the difference between the encrypted image and the target image is small, such visual effect will meet the requirement of camouflage, which means that the original image content is totally covered by a target image content. Even if the attacker recognizes the camouflage, without the secret key K of AES, it is also unfeasible to decrypt the accessorial information that is necessary for restoring the original image. And thus the attacker only gets a meaningless image as shown in Fig. 5(e).

TABLE II
SPACE OCCUPIED BY AI AND PSNRS OF THE ENCRYPTED IMAGES

| Experiment | A | B | C | D | E |
|---|---|---|---|---|---|
| AI(bpp) | 0.523 | 0.499 | 0.521 | 0.554 | 0.508 |
| PSNR(dB) | 30.68 | 30.72 | 30.95 | 30.09 | 30.83 |

The quality of the encrypted image E(I) is measured by the peak-signal-to-noise ratio (PSNR) defined as $PSNR = 10 \times \log_{10} \frac{255^2}{MSE}$, where the MSE (mean-square error) for an $m \times n$ image is computed by formula $MSE = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} - y_{ij})^2$ in which $x_{ij}$ and $y_{ij}$ denote the pixel values of the target image and encrypted image E(I), respectively. The result of five pairs of images listed is displayed in Table II. It can been seen that accessorial information (AI) occupies about 0.521 bits per pixel (bpp) on average. Such large overhead cause large distortion to some extent, but the encrypted image E(I) still can keep an acceptable quality with the PSNR value about equal to 30 dB, which is an accepted visual effect. Besides we also give the average payload of AI and PSNR value of 100 randomly selected tested images, 0.529 bpp and 27.2 dB respectively.

## IV. RDH in Encrypted Image

RIT generates an encrypted image E(I), which has the advantage of keeping a meaningful form of the image com-pared to traditional encryption methods. Therefore, it is free for the cloud server to employ any classical RDH on the encrypted image. Selecting what kind of RDH method depends on whether to keep the image quality or not. In this section we simply adopt two RDH methods, one is a traditional RDH that keeps the quality of images and the other is a unified data embedding and scrambling method that may greatly degrades image structures for embedding large payload.

### *A. Traditional RDH on the encrypted image*

It should be noted that any one of classical RDH method for plaintext image can be implemented to embed and extract watermark in the encrypted image E(I) in the RIT based scheme. As an example, we select the method proposed by Dragoi et al. [7] to embed watermark into the encrypted image. Dragoi et al.'s method is a typical PEE (predicted error expansion) based RDH method, in which a new local least square (LLS) predictor with high prediction accuracy is predicted. Obviously the smaller the PE is, the higher the quality of marked image will be. The scheme of Dragoi et al. is briefly described as follows. For each pixel, a least square predictor is computed on a square block centered on the pixel, which can adaptively make use of every neighbor pixel's distinction in a local region. The most interesting aspect of the approach is the fact that the same predictor can be realized at the receiver side, avoiding the need of embedding a large amount of additional information. Having predicted the current pixel, the predicted error (PE) will be shifted for vacating room or be expanded for embedding one message bit. For more details please refer to [7].

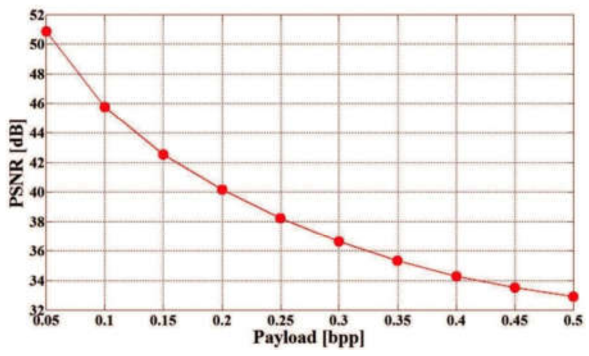


Fig. 9. Average PSNRs between encrypted images and marked images with different embedding payloads for 100 pairs of test images by applying RDH.

We depicted the average PSNR results of 100 test images between the marked image and the encrypted image given different embedding payloads from 0.05 bpp to 0.5 bpp in Fig. 9. It can be viewed that the average PSNR of 100 encrypted images maintains a high value. In Fig. 7 and Fig. 8, marked images of test image D and E after embedding 0.1 and 0.5 bpp payloads are displayed respectively. For both experiments it is hard to distinguish the marked image from the encrypted image.

### *B. Unified embedding and scrambling (UES) on the encrypted image*

From the result of Table II in Section III-C, the amount of accessorial information is already large. So it is hard for traditional RDH methods to earn large embedding capacity while still keeping high visual quality. To meet the demand of large payloads, the cloud server can insert watermark with a unified embedding and scrambling method called UES [28], which deliberately degrades image structure. In such way a marked image with meaningless form may be produced just like the way of traditional encryption based RDH-EI schemes. In fact, in some application cases, the cloud server does not need to consider the quality of marked image as done in all previous RDH-EI schemes [14], [17]–[23], only if the cloud server can losslessly restore the encrypted image E(I) and send it back to the users.

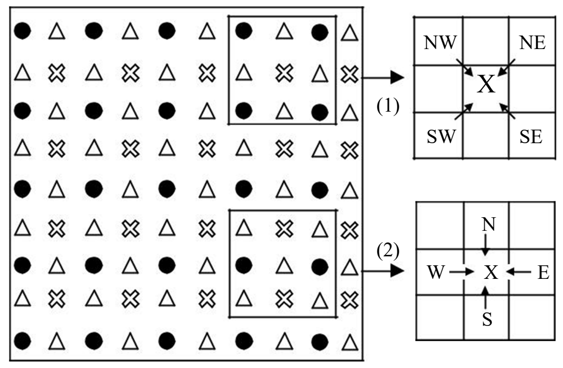
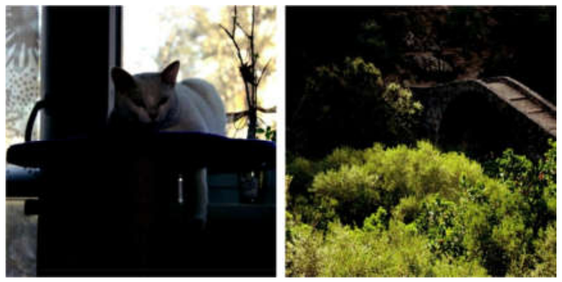


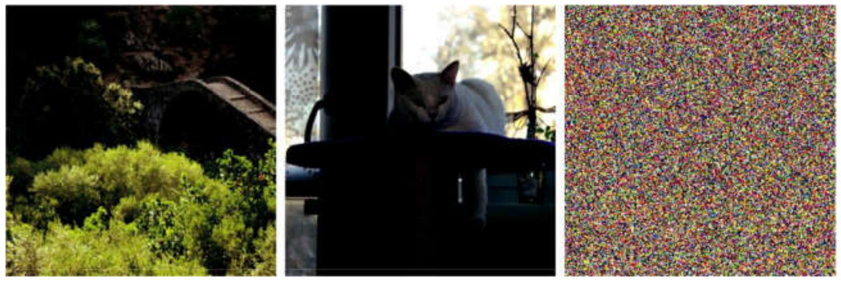
Fig. 10. Illustration of CBP

The UES scheme consists of checkerboard based prediction (CBP) tailored for compress coding algorithm. As shown in Fig. 10, all the pixels are firstly divided into three sets: the Circle set, the Cross set and the Triangle set. The Circle set will not be changed, which is reserved as reference points to predict the pixels belonging to the Cross set and Triangle set. The way of pixel prediction contains two steps, shown in Fig. 10 (1) and (2). In the first step the Cross set is predicted by rounding the result of Eq. (8) and in the second step the Triangle set is predicted by rounding the result of Eq. (9).

$$X = \begin{cases} (NW + SE)/2, & \text{if } \|NE - SW\| > \|NW - SE\| \\ (NE + SW)/2, & \text{if } \|NE - SW\| < \|NW - SE\| \\ (NW + NE + SW + SE)/4, & \text{otherwise} \end{cases} \tag{8}$$

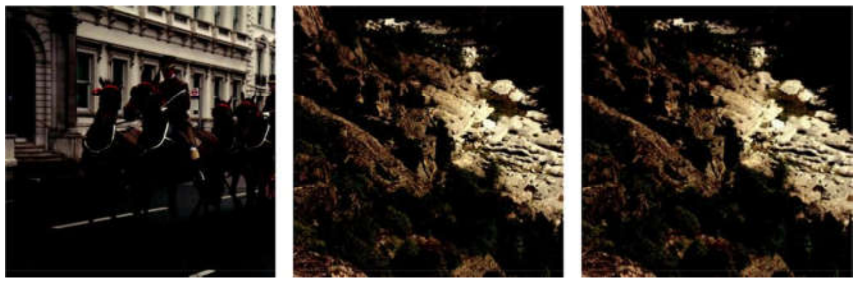


(a) Original image (b) Target image

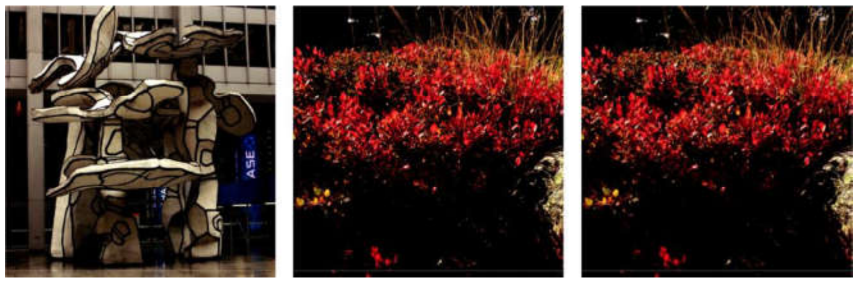


(c) Encrypted image (d) Decrypted image (right key) (e) Decrypted image (wrong key)

Fig. 5. Experimental results of test images in Experiment A.



(a) Original image (b) Target image (c) Encrypted image



(d) Original image (e) Target image (f) Encrypted image

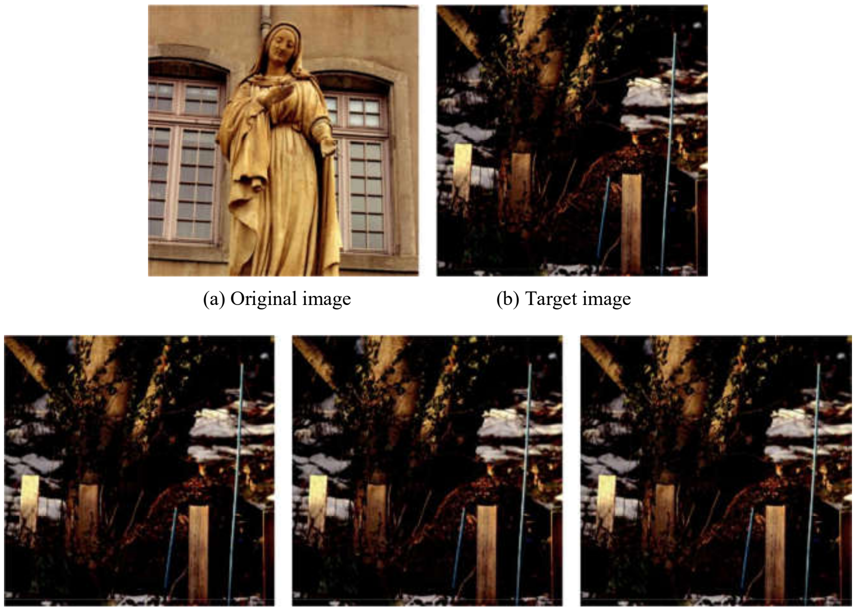
Fig. 6. Experiment results of test images in Experiment B (top row) and Experiment C (bottom row).

$$X = \begin{cases} (W + E)/2, & \text{if } kN - Sk > kW - Ek \\ (N + S)/2, & \text{if } kN - Sk < kW - Ek \\ (W + N + E + S)/4, & \text{otherwise} \end{cases} \qquad (9)$$

After prediction, data embedding can be executed as follows.

1) Compute the prediction errors.
2) Compress prediction errors using run-length and Huffman coding.
3) Directly insert the compressed predicted errors and the watermark into the predicted locations by replacing the predicted pixels.

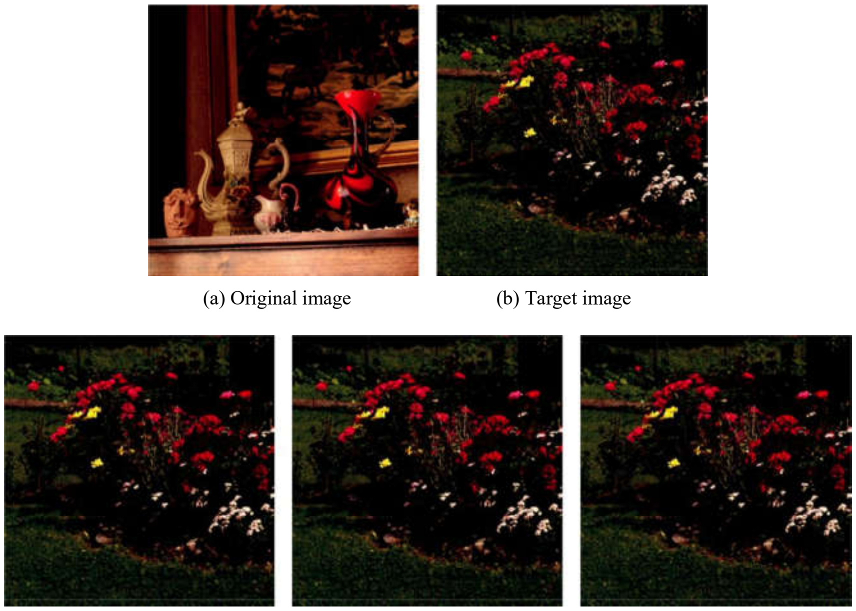
At the receiver side, after extracting the watermark, the decoder needs to decompress the prediction error and add it to predicted pixel value by CBP, which will losslessly generate the pixel value. Note that the original UES method in [28] is not reversible, because, to enlarge payloads, it replaces the



(a) Original image (b) Target image

(c) Encrypted image (d) Marked image (0.1bpp) (e) Marked image (0.5bpp)

Fig. 7. Experimental results of test images in Experiment D.



(a) Original image (b) Target image

(c) Encrypted image (d) Marked image (0.1bpp) (e) Marked image (0.5bpp)

Fig. 8. Experimental results of test images in Experiment E.

prediction error $e_{ij}$ with a truncated value $\tilde{e}_{ij}$ . Herein, we modify UES to be reversible by using $e_{ij}$ directly.

Since the reference pixels (Circle set) remains unchanged both at embedding and extraction, they can be extracted to make up a sub-sampled image. Then the sub-sampled image can be employed by UES again to further embed more data, which is called two-level embedding. In fact, such process can be repeated and realize multi-level embedding as shown in Fig. 11, which will further degrade the visual quality of the image.

Note that the embedding payload of UES can be controlled by two parameters, the prediction error threshold $T$ determining which pixel can be replaced by external message and the embedding level $L$. Obviously the payload increases with the growth of $T$ and $L$ because more locations can be used for data embedding. We evaluate the quality of image structure subjectively by visual inspection and objectively measured by SSIM [29] between marked image and encrypted image. Fig.

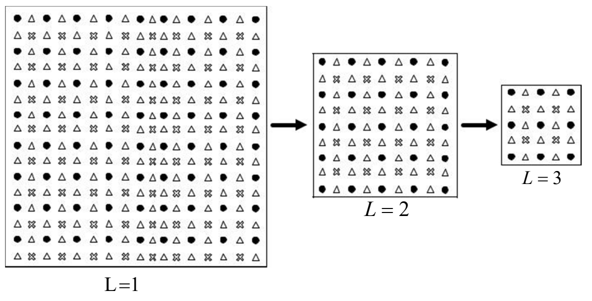


Fig. 11. Illustration of multi-level embedding by UES

12 shows that the average SSIM values for all test images are gradually decreasing with increasing payloads. In addition in Fig. 13 we use test images in Experiment A as an example to show the distortion level for various embedding payloads. With the increase of embedding payload, the image become more and more blurred. It is verified that UES can greatly expand the embedding capacity available for the cloud server.

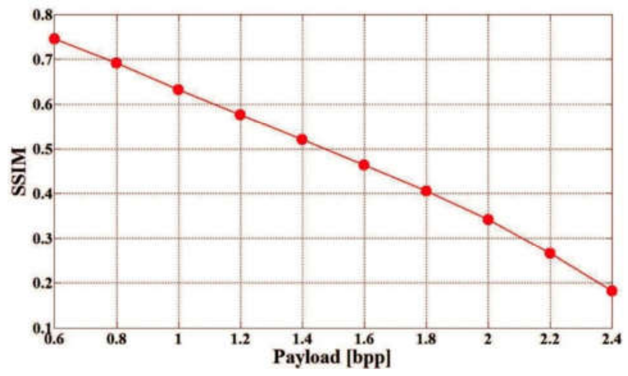


Fig. 12. Average SSIM between encrypted images and marked images with different embedding payloads for 100 pairs of test images by applying UES

## V. Conclusion and Future work

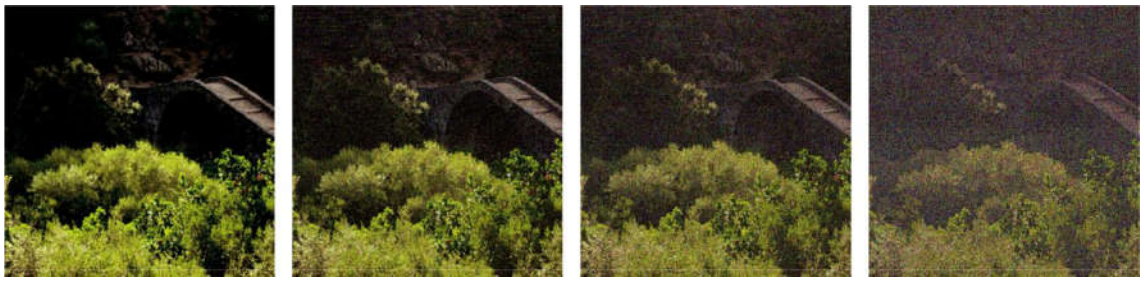
In this paper we propose a novel framework for reversible data hiding in encrypted image (RDH-EI) based on reversible image transformation (RIT). Different from previous frame-works which encrypt a plaintext image into a ciphertext form, RIT-based RDH-EI shifts the semantic of original image to the semantic of another image and thus protect the privacy of the original image. Because the encrypted image has the form of a plaintext image, it will avoid the notation of the curious cloud server and it is free for the cloud sever to choose any one of RDH methods for plaintext images to embed watermark.

We realize an RIT based method by improving the image transformation technique in [26] to be reversible. By RIT, we can transform the original image to an arbitrary selected target image with the same size, and restore the original image from the encrypted image in a lossless way. Two RDH methods including PEE-based RDH and UES are adopted to embed watermark in the encrypted image to satisfy different needs on image quality and embedding capacity.

Several interesting problems can be considered in the future, including how to improve the quality of the encrypted image and how to extend idea of RIT to audio and video.

## References

[1] K. Hwang, D. Li, “Trusted cloud computing with secure resources and data coloring,” IEEE Internet Computing, vol. 14, no. 5, pp. 14-22, Sept.-Oct. 2010.
[2] F. Bao, R. H. Deng, B. C. Ooi, et al., “Tailored reversible watermarking schemes for authentication of electronic clinical atlas,” IEEE Trans. on Information Technology in Biomedicine, vol. 9, no. 4, pp. 554-563, Dec. 2005.
[3] F. Willems, D. Maas, and T. Kalker, “Semantic lossless source coding,” 42nd Annual Allerton Conference on Communication, Control and Computing, Monticello, Illinois, USA, pp. 1411-1418, 2004.
[4] W. Zhang, X. Hu, N. Yu, et al. “Recursive histogram modification: establishing equivalency between reversible data hiding and lossless data compression,” IEEE Trans. on Image Processing, vol. 22, no. 7, pp. 2775-2785, Jul. 2013.
[5] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, “Reversible watermarking algorithm using sorting and prediction,” IEEE Trans. on Circuits and Systems for Video Technology, vol.19, no.7, pp. 989-999, Jul. 2009.
[6] B.ou, X. Li, Y. Zhao, R. Ni, Y. Shi, “Pairwise prediction-error expansion for efficient reversible data hiding,” IEEE Trans. on Image Processing, vol. 22, no.12, pp. 5010-5021, Dec. 2013.
[7] Ioan-Catalin Dragoi, Dinu Coltuc, “Local-prediction-based difference expansion reversible watermarking,” IEEE Trans. on Image Processing, vol. 23, no. 4, pp. 1779-1790, Apr. 2014.
[8] Z. Ni, Y. Shi, N. Ansari, and S. Wei, “Reversible data hiding,” IEEE Trans. on Circuits and Systems for Video Technology, vol. 16, no. 3, pp. 354-362, Mar. 2006.
[9] J. Tian, “Reversible data embedding using a difference expansion,” IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, no.8, pp. 890-896, Aug. 2003.
[10] X. Hu, W. Zhang, X. Li, N. Yu, “Minimum rate prediction and optimized histograms modification for reversible data hiding,” IEEE Trans. on Information Forensics and Security, vol. 10, no. 3, 653-664, Mar. 2015.
[11] X. Hu, W. Zhang, X. Hu, N. Yu, X. Zhao, F. Li, “Fast estimation of optimal marked-signal distribution for reversible data hiding,” IEEE Trans. on Information Forensics and Security, vol. 8, no. 5, pp. 779-788, May. 2013.
[12] W. Zhang, X. Hu, N. Yu, “Optimal transition probability of reversible data hiding for general distortion metrics and its applications,” IEEE Trans. on Image Processing, vol. 24, no. 1, pp. 294-304, Jan. 2015.
[13] 2014 celebrity photo hack, [Online]. Available: http://en.wikipedia.org/wiki/2014_celebrity_photo_hack
[14] K. Ma, W. Zhang, X. Zhao, N. Yu, F. Li, “Reversible data hiding in encrypted images by reserving room before encryption,” IEEE Trans. on Information Forensics and Security, vol. 8, no. 3, pp. 553-562, Mar. 2013.
[15] M. Johnson, P. Ishwar, V. M. Prabhakaran, D. Schonberg, and K. Ramchandran, “On compressing encrypted data,” IEEE Trans. on Signal Processing, vol. 52, no. 10, pp. 2992-3006, Oct. 2004.
[16] W. Liu, W. Zeng, L. Dong, and Q. Yao, “Efficient compression of encrypted grayscale images,” IEEE Trans. on Image Processing, vol. 19, no. 4, pp. 1097-1102, Apr. 2010.
[17] X. Zhang, “Reversible data hiding in encrypted images,” IEEE Signal Processing Letters, vol. 18, no. 4, pp. 255-258, Apr. 2011.
[18] W. Hong, T. Chen, H. Wu, “An improved reversible data hiding in encrypted images using side match,” IEEE Signal Processing Letters, vol. 19, no. 4, pp. 199-202, Apr. 2012.
[19] J. Zhou, W. Sun, L. Dong, et al., “Secure reversible image data hiding over encrypted domain via key modulation,” IEEE Trans. on Circuits and Systems for Video Technology, vol. 26, no. 3, pp. 441-452, Mar. 2016.
[20] X. Zhang, “Separable reversible data hiding in encrypted image,” IEEE Trans. on Information Forensics and Security, vol. 7, no. 2, pp. 826-832, Apr. 2012.
[21] Z. Qian, and X. Zhang, “Reversible data hiding in encrypted image with distributed source encoding,” IEEE Trans. on Circuits and Systems for Video Technology, vol. 26, no. 4, pp. 636-646, Apr. 2016.
[22] W. Zhang, K. Ma and N. Yu, “Reversibility improved data hiding in encrypted images,” Signal Processing, vol. 94, pp. 118-127, Jan. 2014.



(a) Encrypted image (b) Marked image (1.0bpp) (c) Marked image (1.7bpp) (d) Marked image (2.4bpp)

Fig. 13. Distortion of the encrypted image with various payloads by applying UES in the test images of Experiment A.

[23] X. Cao, L. Du, X. Wei, et al., "High capacity reversible data hiding in encrypted images by patch-level sparse representation," IEEE Trans. on Cybernetics, vol. 46, no. 5, pp. 1132-1143, May. 2016.

[24] S. Yu, C. Wang and K. Ren,"Achieving secure, scalable, and finegrained data access control in cloud computing," IEEE Proceeding of INFOCOM 2010, pp. 1-9, Mar. 2010.

[25] I. Lai and Wen. Tsai,"Secret-fragment-visible mosaic image-a new computer art and its application to information hiding," IEEE Trans. on Information Forensics and Security, vol. 6, no. 3, pp. 936-945, Sept. 2011.

[26] Y. Lee and W. Tsai,"A new secure image transmission technique via secret-fragment-visible mosaic images by nearly reversible color transformation," IEEE Trans. on Circuits and Systems for Video Technology, vol. 24, no. 4, pp. 695-703, Apr. 2014.

[27] BossBase image database, [Online]. Available: http://agents.fel.cvut.cz/stegodata/RAWS

[28] R. Rad, K. Wong and J. Guo, "An unified data embedding and scrambling method," IEEE Trans. on Image Processing, vol. 23, no. 4 , pp. 1463-1475, Apr. 2014.

[29] Z. Wang, A. Bovik, H. Sheikh and E. Simoncelli, "Image quality assessment: from error measurement to structural similarity," IEEE Trans. on Image Processing, vol. 13, no. 4, pp. 600-612, Apr. 2004.