Noise Removal in Embedded Image With Bit Approximation

Xianquan Zhang, Xuelong Li, *Fellow, IEEE*, Zhenjun Tang, *Member, IEEE*, Shichao Zhang, *Senior Member, IEEE*, and Shaomin Xie

Abstract—Stego-images are often contaminated by interchannel noise or active noise attack when communicating on the Web. And it is challenging to restore embedded image from corrupted stego-image. This paper studies a *k*NN-bit approximation algorithm to remove noises in embedded image. The proposed algorithm distinguishes reliable bits from extracted bits, and estimates pixel values by keeping reliable bits unchanged and correcting unreliable bits. Specifically, the 8th (highest) unreliable bit of a pixel can be approximated with its nearest neighbor pixels. And then, if an unreliable bit locates at any one of the $5^{th} \sim 7^{th}$ bits of a pixel, it is adjusted with two nearest neighbors of the pixel, where the pixel is in-between these two nearest neighbors. Finally, for other unreliable bits, each one is approximated by the maximum and minimum possible values of nearest neighbors of its pixel. We conduct experiments for illustrating the efficiency, and demonstrate that the proposed algorithm can recover the embedded images with good visual quality from corrupted stego-images.

Index Terms—Embedded image, data hiding, noise removal, bit approximation, *k* nearest neighbors (*k*NN)

1 Introduction

In the digital era, people must pay much attention to privacy protection when communicating on the Web. Consequently, information security has become an important issue. Data hiding [1],[2],[3] is one of information security techniques, which has widely been used in, such as copyright protection [4], covert communication, image forensics, content authentication, and data binding. Image-based data hiding is an important kind of data hiding algorithms, where secret message and cover signal are both digital images. This kind of techniques finds many useful applications. For example, a sender can achieve secure transmission of a military map from one place to another place by embedding it into a landscape photograph. After data embedding, stego-image can be transmitted through the Internet. At the receiver's side, the military map can be restored from stego-image. As digital images are easily corrupted by interchannel noise [5], [6] or active noise attack, visual quality of the secret image extracted/decrypted from corrupted stego-image is inevitably hurt. To overcome this problem, a possible strategy is to use filtering algorithms [7].

The existing filtering algorithms have shown good performance in noise removal for those images directly contaminated, but they are ineffective in recovering/restoring embedded image from corrupted stego-image due to the following fact. When stego-image is contaminated by impulse noise, some bits per pixel in the decrypted (extracted embedded) image are altered, whereas the other bits can be kept unchanged. In other words, for embedded image extracted from corrupted stego-image, a dirty/noisy pixel can contain both reliable and unreliable bits together. This indicates that noise removal in embedded image should be done by bit approximation instead of pixel replacement. In general, conventional filtering algorithms improve visual quality of the contaminated images by identifying noisy pixels and replacing their values regardless of the actual bit values. As pixel replacement strategy does not utilize those reliable bits of a pixel, it is difficult to guarantee visual quality of embedded image.

To address this actual yet challenging issue, we propose a *k*NN-bit approximation algorithm to remove noises in embedded image, referred to bit-approximation algorithm. Different from the existing filtering algorithms, our algorithm distinguishes reliable bits from the extracted bits, and estimates pixel values by keeping reliable bits unchanged and correcting unreliable bits. Our contributions are summarized as follows.

(1) Reliable bits of a pixel are kept unchanged during pixel restoration. We exploit an efficient noise detection method to find noisy pixels in stego-image, and classify the extracted bits into reliable bits and unreliable bits. During bit approximation, reliable bits are kept unchanged. This is helpful to make the restored pixel close to its original value.

(2) Unreliable bits of a pixel are approximated by the corresponding bits of its *k*NN pixels. As high bits are important than low bits in pixel value, the highest unreliable bit (i.e., the 8th bit) is first corrected, the $5^{th} \sim 7^{th}$ unreliable bits are then verified, and other unreliable bits are finally approximated.

- *X. Zhang, Z. Tang, S. Zhang, and S. Xie are with the Guangxi Key Lab of Multi-Source Information Mining & Security, Department of Computer Science, Guangxi Normal University, Guilin 541004, P.R. China. E-mail: {zxq6622, tangzj230}@163.com, zhangsc@gxnu.edu.cn, 568413949@qq.com.*
- *X. Li is with the School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, P.R. China. E-mail: xuelong_li@nwpu.edu.cn.*

Manuscript received 2 Aug. 2018; revised 25 Mar. 2020; accepted 30 Apr. 2020. Date of publication 14 May 2020; date of current version 3 Feb. 2022. (Corresponding authors: Zhenjun Tang, Shichao Zhang.) Recommended for acceptance by J. Tang. Digital Object Identifier no. 10.1109/TKDE.2020.2992572

The corrected unreliable bits and reliable bits jointly contribute efficient pixel restoration.

We conduct performance comparisons with some well-known filtering algorithms for illustrating the efficiency. Our experiments demonstrate that our algorithm can restore embedded images with good visual quality, and outperforms the compared algorithms.

The rest of this paper is organized as follows. Section 2 review the related work. Section 3 presents our algorithm. Section 4 describes experimental results. Conclusions are finally given in Section 5.

2 Related Work

Many researchers have devoted to developing high performance filtering methods. For example, Sun and Neuvo [8] proposed a median and weighted median based impulse detector to remove impulse noises in digital images. Alajlan *et al.*[9] exploited a recursive minimum-maximum method to build a variation of peak-and-valley filter. This filter can recover dirty images with high impulsive noise contamination. Luo[10] introduced a two-phase method. It first detects impulse noise and then removes impulse noise. This method does not require a training step. Ibrahim *et al.* [11] exploited two median filters (adaptive filter and switching filter) to remove impulsive noise. In this scheme, threshold parameter is not needed. Chen and Lien[12] presented a method without previous training. In this method, noisy pixels are determined by a detector and restored by an edge-preserving filter.

In another study, Kang and Wang [13] removed image noises by a modified switching median filter. Wang and Wu [14] proposed a fast algorithm preserving image details. This algorithm is based on convolution results of Laplacian operators. Wan *et al.*[15] developed a robust method for noise density estimation and applied it to iterative denoising process. Zhang[16] proposed an iterative algorithm for recovering noisy images. In this algorithm, noisy pixels are found by an adaptive center-weighted median filter and restored by iteratively using a median filter. This algorithm cannot reach fast speed due to iterative computation. In another work, Fabijańska and Sankowski[17] exploited local intensity extreme to identify noisy pixels and corrected them by median filter. This method has good performance in image recovery under a high noise density. However, for the cases of low and medium noise densities, its restoration results are worse than standard median filter.

Recently, Zhang and Li[18] presented an adaptive weighted mean filter. This filter finds noisy pixel with adaptive window size and replaces its value with the weighted mean of the window. Ahmed and Das[19] proposed two-stage method for removing salt-and-pepper noise. This method uses an adaptive fuzzy filter to detect noisy pixel and restores pixel value by a weighted mean filtering operation on nearby uncorrupt pixels. In another work, Bai *et al.* [20] replaced median filter with continued fractions interpolation filter for salt and pepper noise removal. Pyatykh and Hesser[21] designed a method for removing salt and pepper noise in binary images. This method requires computation of block prior probabilities from training noise-free images. Sun *et al.*[22] achieved salt and pepper noise removal by introducing shearlet transform to the filtering stage. Zhang *et al.*[23] took noisy pixels as missing data for inpainting, and adaptively selected convolution mask for iterative filtering in terms of local image texture. Gupta *et al.*[24] introduced a novel de-noising scheme based on adaptive dual thresholds and simple median filter. Varatharajan *et al.*[25] proposed a high density salt and pepper noise removal method based on kriging interpolation. The noise pixel is replaced by a value which is interpolated with the weights calculated using semi-variance between the noise pixel and the non-noisy pixels. Veerakumar *et al.*[26] adopted an edge preserving contextual model to restore the noisy pixels, in which Gaussian kernel is used for edge preserving of the processing pixel. Li *et al.*[27] presented a novel filter based on local and global image information. The local image information is used to estimate noise density and the global information is exploited to guide noise detection rectification.

In recent years, learning based methods including sparse representation and deep learning draw much attention for their outstanding image restoration performance[28], [29]. For example, Chen *et al.*[30] proposed a weighted couple sparse representation model for removing impulse noise. A weighted rank-one minimization problem is addressed to train a dictionary on the noisy raw data, which enables to acquire more original data features. Huang *et al.*[31] utilized Laplacian scale mixture (LSM) modeling and nonlocal low-rank regularization to remove mixture noise efficiently. In [32], a deep convolutional neural network (CNN) architecture and particle swarm optimization are adopted to detect impulse noise pixels accurately, and the median filter is used to clean noise pixels. Wang *et al.*[33] introduced a low-rank prior in small oriented noise-free image patches, then integrated low-rank and sparse matrix recovery to detect and remove nonpointwise random-valued impulse noise simultaneously. Wang *et al.*[34] proposed a variational method to automatically classify noise according to different statistical parameters and integrated the CNN regularizer to improve quality of the restored images significantly. Xing *et al.*[35] employed CNN with the multi-layer structure to remove salt and pepper noise. In [36], a deep CNN (DCNN) is used to remove Gaussian noises, where residual learning is employed to separate noise from noisy observation. The batch normalization and residual learning are integrated to accelerate the training process as well as boost the denoising performance. To improve the flexibility for dealing with spatially variant noise, Zhang *et al.* [37] proposed a fast and flexible denoising CNN, namely FFDNet. The FFDNet is able to handle noise with different levels and drive outstanding restoration performance on both synthetic noisy images corrupted by additive white Gaussian noise and real-world noisy images. In another work, Dong *et al.* [38] presented a new deep-learning framework for removing noises in three-dimensional hyperspectral images and achieved excellent denoising performance.

3 *k*NN-Bit Approximation

Our algorithm first finds those unreliable bits of every pixel extracted from the stego-image and then conducts bit approximation with those bits of its k nearest neighbor (kNN) [39],[40],[41] pixels. The kNN imputation method is an efficient technique for recovering missing numerical data

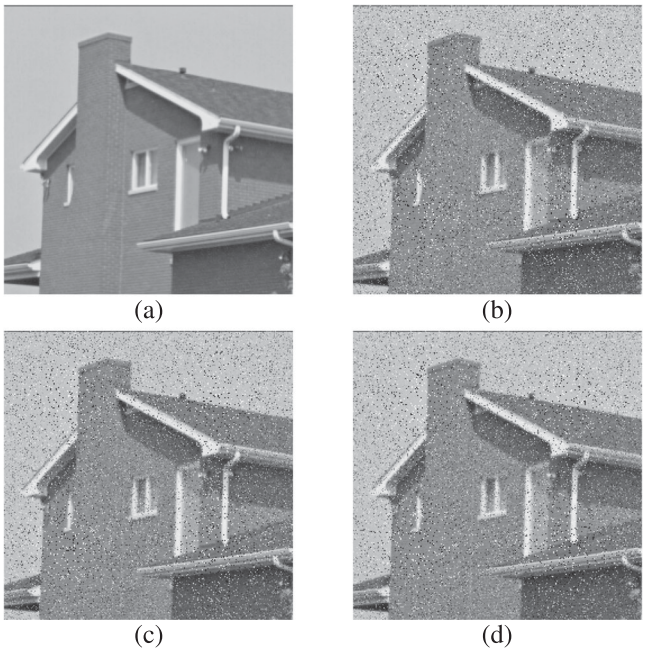


Fig. 1. Cover image and corrupted stego-images generated by different methods. (a)Cover image; (b)Corrupted stego-image generated by LSB substitution; (c)Corrupted stego-image generated by [42]; (d)Corrupted stego-image generated by [43].

and has been successfully applied to many real applications [39]. Here, the kNN method is used to approximate unreliable bits of noisy pixel. Let p be a noisy pixel. Take p as the center of a window sized $(2D+1)\times(2D+1)$. Thus, the pixels in the window (except p itself) are the kNN pixels of p. Suppose that $p_1, p_2, \ldots, p_k$ represent the kNN pixels of p, where $k=(2D+1)(2D+1)-1$. In this study, the unreliable bits of p are approximated by the corresponding bits of its kNN pixels. The detailed procedures of unreliable bit identification and bit correction are described as follows.

3.1 Identifying Unreliable Bits

Image-based data hiding algorithm includes two parts: data embedding and data extraction. In data embedding, the algorithm converts a secret image into a binary stream and embeds these bits into one or several lowest bit-planes of the cover image. Data extraction is an inverse procedure of data embedding. Embedded bits are first extracted from the used bit-planes and the extracted binary stream is then exploited to reconstruct the secret image. It is clear that if the stego-image is corrupted by noises, the used bit-planes of the stego-image are changed and therefore the decrypted image is also corrupted. Fig. 1a is a gray image of size 512×512 and Fig. 2a is a gray image of size 256×256. We

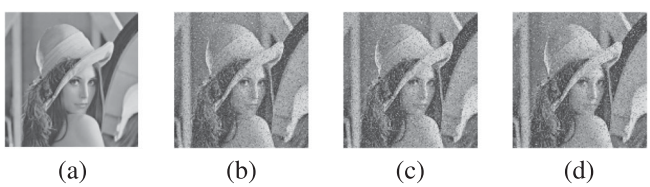


Fig. 2. Secret image and extracted results from different corrupted stego-images. (a) Secret image; (b) Result of Fig. 1b; (c) Result of Fig. 1c; (d) Result of Fig. 1d.

TABLE 1
Noise Density Comparisons Between the Stego-Images and the Decrypted Images

Stego-images	0.1	0.2	0.3	0.4	0.5
Extracted results of LSB substitution	0.3337	0.5684	0.7245	0.8335	0.9002
Extracted results of [42]	0.2787	0.4922	0.6546	0.7758	0.8623
Extracted results of [43]	0.2763	0.4898	0.6518	0.7781	0.8608

embedded Fig. 2a into Fig. 1a by least-significant-bit (LSB) substitution using two lowest bit-planes and the methods [42], [43], and added different salt and pepper noises into the stego-images. Figs. 1b, 1c, and 1d are the corrupted stego-images generated by LSB substitution and the methods [42], [43] under the salt and pepper noise with 0.1 densities, respectively. Figs. 2b, 2c, and 2d are the extracted results of the Figs. 1b, 1c, and 1d, respectively. As each pixel of the secret image is concealed in several pixels of the stego-image, noise density in the decrypted image will be bigger than that in the corrupted stego-image. We calculated noise densities in the decrypted images when the stego-images are corrupted by different densities. The results are listed in Table 1. From the results, we observe that, as the noise density in the stego-image reaches 0.5, the density in the decrypted images is about 0.86, leading to badly degraded images. Therefore, in the following experiments, we only consider the cases that noise densities in the stego-image are not bigger than 0.5.

To identify corrupted bits of the decrypted image, we exploit the noise detection method [44] to find noisy pixels in the stego-image. Thus, we mark all bits during data extraction. Let $b(p,n)$ be the nth bit of the pixel p, and $c(p,n)$ be the sign of the nth bit of p, where $n=1,2,3,4,5,6,7,8$. Thus, we have

$$c(p,n)=\begin{cases}1 & \text{if } b(p,n) \text{ is extracted from a noisy pixel.}\\ 0 & \text{if } b(p,n) \text{ is not extracted from a noisy pixel.}\end{cases} \quad (1)$$

Clearly, the nth bit of p is unreliable when $c(p,n)=1$. If $c(p,n)=0$, the nth bit of p is reliable.

As the pixel value of p is calculated by $\sum_{n=1}^{8}2^{n-1}\times b(p,n)$, restoration is achieved when all unreliable bits of p are corrected. Clearly, those high bits are important than those low bits in pixel value. Therefore, we first correct the highest bit (i.e., the 8th bit), verify those the $5^{th}\sim7^{th}$ bits, and finally approximate other unreliable bits using kNN bits.

3.2 Correcting the 8th Bit

Let a_p be the original pixel value of p. Thus, we can correct the 8th bit of p by using the 8th bits of its kNN pixels. Let cv_1 and cv_2 be two signs for approximate estimation, whose initial values are both 0. We perform the 8th bit approximation in terms of the following conditions.

(1) Case 1:$c(p,7)=0$.

We correct the 8th bit of p by using its kNN pixels whose 7th and 8th bits are both reliable. This is achieved by two schemes as follows.

Scheme 1: Correct the 8th bit of p by exploiting those kNN pixels whose 7th bits are equal to the 7th bit of p.

TABLE 2
Bit Values With Minimum Difference

Bit no.	$a^{(0)}_{p_{i_j}}$	$a^{(1)}_{p_{i_1}}$
8	1	0
7	*	*
6	0	1
5	0	1
4	0	1
3	0	1
2	0	1
1	0	1

To do so, we find the pixels satisfying $\begin{cases} c(p_i,8)=0 \\ c(p_i,7)=0 \\ b(p_i,7)=b(p,7) \end{cases}$ $(i=1,2,\ldots,k)$ from p_1, p_2,...,p_k and label them as p_{i_1}, p_{i_2},...,p_{i_r} , where $r=\sum_{i=1}^{k}[1-c(p_i,8)][1-c(p_i,7)][1-|b(p_i,7)-b(p_i,7)|]$. Let $a_{p_{i_1}}, a_{p_{i_2}}, \ldots, a_{p_{i_r}}$ be the original values of $p_{i_1}, p_{i_2}, \ldots, p_{i_r}$, respectively. Thus, the 8th bit of p can be determined as follows.

If $b(p_{i_j},8)\neq b(p_{i_1},8)(2\leq j\leq r)$, we have $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_1},8)=0 \end{cases}$ or $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_1},8)=1 \end{cases}$. If $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_1},8)=0 \end{cases}$, let the $1^{st}\sim 6^{th}$ bits of p_{i_j} and p_{i_1} be 0 and 1, respectively. Then, suppose that the values of p_{i_j} and p_{i_1} are $a^{(0)}_{p_{i_j}}$ and $a^{(1)}_{p_{i_1}}$. As the 7th bits of $p_{i_1}, p_{i_2}, \ldots, p_{i_r}$ are equal and reliable, their values will not affect the minimum difference between $a^{(0)}_{p_{i_j}}$ and $a^{(1)}_{p_{i_1}}$. Table 2 is an example of the detailed bit values with minimum difference, where '*' represents 0 or 1. In this case, we have $|a_{p_{i_j}}-a_{p_{i_1}}|\geq|a^{(0)}_{p_{i_j}}-a^{(1)}_{p_{i_1}}|>2^6$. Similarly, if $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_1},8)=1 \end{cases}$, let the $1^{st}\sim 6^{th}$ bits of p_{i_j} and p_{i_1} be 1 and 0, respectively. Suppose that the values of p_{i_j} and p_{i_1} are $a^{(1)}_{p_{i_j}}$ and $a^{(0)}_{p_{i_1}}$. Thus, we also have $|a_{p_{i_j}}-a_{p_{i_1}}|\geq|a^{(1)}_{p_{i_j}}-a^{(0)}_{p_{i_1}}|>2^6$. From the above analysis, we find that pixel difference in neighbor is large when $b(p_{i_j},8)\neq b(p_{i_1},8)$. So we don't correct the 8th bit of p.

If $b(p_{i_1},8)=b(p_{i_2},8)=...=b(p_{i_r},8)$, let $v_1=b(p_{i_1},8)$, where $v_1\in\{0,1\}$. Suppose that $b(p,8)\neq v_1$. Thus, we have $\begin{cases} b(p_{i_j},8)=1 \\ b(p,8)=0 \end{cases}$ or $\begin{cases} b(p_{i_j},8)=0 \\ b(p,8)=1 \end{cases}$. Therefore, we can conclude $|a_{p_{i_j}}-a_p|>2^6$ with similar analysis presented in the last paragraph. Then, v_1 is the probable value of $b(p,8)$. So let $cv_1=1$.

Scheme 2: Correct the 8th bit of p by using those kNN pixels whose 7th bits are not equal to the 7th bit of p.

For those kNN pixels whose 7th and 8th bits are both reliable, if their 7th bits are not equal to the 7th bit of p, these bits will have the same value. If their 8th bit values are different, $b(p,8)$ cannot be corrected in terms of Scheme 1. If their 8th bits and the 7th bits are not equal, $b(p,8)$ cannot be determined. Therefore, we exploit those neighbor pixels, whose $7^{th}\sim 8^{th}$ bits are reliable and equal, to estimate $b(p,8)$. Detailed steps are as follows.

Calculate the pixels p_1, p_2,...,p_k satisfying $\begin{cases} c(p_i,8)=0 \\ c(p_i,7)=0 \\ b(p_i,7)\neq b(p,7) \\ b(p_i,7)=b(p,8) \end{cases}$ $(i=1,2,\ldots,k)$ and label them as p_{i_1}, p_{i_2},...,p_{i_s}, where $s=\sum_{i=1}^{k}[1-\ c(p_i,8)][1-c(p_i,7)][|b(p_i,7)-b(p,7)|][1-|b(p_i,7)-b(p_i,8)|]$. Let $a_{p_{i_1}}, a_{p_{i_2}}, \ldots, a_{p_{i_s}}$ be their original pixel values, i.e., the values without noise contamination. Thus, we use their high bits to estimate $b(p,8)$.

Clearly, we have $b(p_{i_1},7)=b(p_{i_2},7)=...=b(p_{i_s},7)$, $b(p_{i_1},8)=b(p_{i_2},8)=...=b(p_{i_s},8)$, and $b(p_i,8)=b(p_i,7)$, where $i=1,2,\ldots,s$. Let $v_2=b(p_{i_1},8)$, where $v_2\in\{0,1\}$. Assume that $b(p,8)\neq v_2$. Then, we have $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_j},7)=1 \\ b(p,8)=0 \\ b(p,7)=0 \end{cases}$ or $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_j},7)=0 \\ b(p,8)=1 \\ b(p,7)=1 \end{cases}$. If $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_j},7)=1 \\ b(p,8)=0 \\ b(p,7)=0 \end{cases}$, let the $1^{st}\sim 6^{th}$ bits of p_{i_j} and p be 0 and 1, and the results of p_{i_j} and p be labeled as $a^{(0)}_{p_{i_j}}$ and $a^{(1)}_{p}$, respectively. Thus, $|a_{p_{i_j}}-a_p|\geq|a^{(0)}_{p_{i_j}}-a^{(1)}_{p}|>2^7$. Similarly, if $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_j},7)=0 \\ b(p,8)=1 \\ b(p,7)=1 \end{cases}$, let the $1^{st}\sim 6^{th}$ bits of p_{i_j} and p be 1 and 0, and the results of p_{i_j} and p be labeled as $a^{(1)}_{p_{i_j}}$ and $a^{(0)}_{p}$, respectively. Thus,$|a_{p_{i_j}}-a_p|\geq|a^{(1)}_{p_{i_j}}-a^{(0)}_{p}|>2^7$ is also available. Therefore, let $cv_2=1$, and v_2 is the probable value of $b(p,8)$.

In summary, when $c(p,7)=0$, the 8th bit of p can be approximated as follows.

If $cv_1=0$ and $cv_2=0$, pixel difference in neighbor is large, indicating that p is probably in a textural or edge region. So the 8th bit estimation is not needed.

If $cv_1=1$ and $cv_2=1$, the 8th bit approximation is also not needed when $v_1\neq v_2$. This is because no matter which value the 8th bit is, large pixel difference in neighbor still exists, meaning that this region is a textural or edge region. When $v_1=v_2$, let $b(p,8)=v_1$ and $c(p,8)=0$. This operation can make a small difference between p and its neighbor pixels.

If $cv_1=1, cv_2=0$ and $r\geq 2$, let $b(p,8)=v_1$ and $c(p,8)=0$.
If $cv_1=0, cv_2=1$ and $s\geq 2$, let $b(p,8)=v_2$ and $c(p,8)=0$.

During the approximation, calculation is performed in the 3×3 window. If the approximation is not achieved, we repeat the calculation in 5×5 window. If the conditions are still not satisfied, the 8th bit estimation is discarded.

(2) Case 2: $c(p,7)=1$.

If the 7th and 8th bits of neighbor pixels are all reliable but different and the 7th and 8th bits of p are both unreliable, the actual value of $b(p,8)$ cannot be determined. So the 8th bit of p is not corrected. If the 7th and 8th bits of some neighbor pixels are reliable and equal, we use them to approximate the 8th bit of p as follows.

Calculate the pixels p_1, p_2,...,p_k satisfying $\begin{cases} c(p_i,8)=0 \\ c(p_i,7)=0 \\ b(p_i,7)=b(p_i,8) \end{cases}$ $(i=1,2,\ldots,k)$ and label them as p_{i_1}, p_{i_2},..., p_{i_t}, where $t=\sum_{i=1}^{k}[1-c\ (p_i,8)][1-c(p_i,7)][1-|b(p_i,7)-\ b(p_i,8)|]$. Approximation is done in terms of the 8th bit values of these pixels.

If $b(p_{i_j},8)\neq b(p_{i_1},8)(2\leq j\leq t)$,we have $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_j},7)=1 \\ b(p_{i_1},8)=0 \\ b(p_{i_1},7)=0 \end{cases}$ or $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_j},7)=0 \\ b(p_{i_1},8)=1 \\ b(p_{i_1},7)=1 \end{cases}$. In both cases, we have $|a_{p_{i_j}}-a_{p_{i_1}}|>2^7$.

TABLE 3
Restoration Ratios Under Different Noise Densities

Noise density	0.1	0.2	0.3	0.4	0.5
N_{right}	5870	10921	14909	17753	19178
N_{total}	5878	10938	14930	17803	19234
R	99.86%	99.84%	99.86%	99.72%	99.71%

Thus, if $b(p_{i_j},8) \neq b(p_{i_1},8)$, the difference of neighbor pixels is large. This means that the pixel is probably in a textural or edge region. So the 8th bit estimation is not needed.

If $b(p_{i_1},8)=b(p_{i_2},8)=...=b(p_{i_t},8)$, we have $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_j},7)=1 \\ b(p,7)=0 \end{cases}$ or $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_j},7)=0 \\ b(p,7)=1 \end{cases}$ under the assumption that $b(p,8) \neq b(p_{i_1},8)$. If $\begin{cases} b(p_{i_j},8)=1 \\ b(p_{i_j},7)=1 \\ b(p,8)=0 \end{cases}$, let the $1^{st} \sim 6^{th}$ bits of p_{i_j} and p be 0 and 1, respectively. Suppose that the values of p_{i_j} and p are $a_{p_{i_j}}^{(0)}$ and $a_p^{(1)}$. Thus,$|a_{p_{i_j}} - a_p| > 2^6$. Similarly, if $\begin{cases} b(p_{i_j},8)=0 \\ b(p_{i_j},7)=0 \\ b(p,8)=1 \end{cases}$, let the $1^{st} \sim 6^{th}$ bits of p_{i_j} and p be 1 and 0, respectively. Suppose that the values of p_{i_j} and p are $a_{p_{i_j}}^{(1)}$ and $a_p^{(0)}$. Thus,$|a_{p_{i_j}} - a_p| > 2^6$. Therefore, when $c(p,7) \neq 0$, we let $b(p,8)=b(p_{i_1},8)$ and $c(p,8)=0$ if $b(p_{i_1},8)=b(p_{i_2},8)=...=b(p_{i_t},8)$, where $t \geq 2$.

During the bit approximation, computation is only performed in 3×3 window. Similarly, if the restoration is not achieved, we repeat the computation in the 5×5 window. If restoration conditions are still not satisfied, bit approximation is discarded.

To validate the performance of the 8th bit approximation, we took different cover images and secret images as test images, exploited LSB substitution to perform data embedding, and obtained a series of stego-images. To produce corrupted stego-images, we added salt and pepper noises with different densities to the stego-images. During the operation, we recorded the positions of noisy pixels. Next, we exploited our approach to perform bit restoration and counted the right bit number. To measure the estimation performance, a restoration ratio is defined as

$$R = \frac{N_{\text{right}}}{N_{\text{total}}} \times 100\%, \tag{2}$$

TABLE 4
Correct Unreliable Bits Between Two Reliable Bits

Bit no.	p	p_{i_1}
...		
n_{j+1}	*	*
...		
n	$b(p,n)$	$b(p_{i_1},n)$
...		
n_j	*	*
...		

TABLE 5
Restoration Ratios of Bit Estimation Between Two Reliable Bits

Noise density	0.1	0.2	0.3	0.4	0.5
N_{right}	10028	17297	21463	21528	17865
N_{total}	10111	17443	21690	21783	18062
R	99.18%	99.16%	98.95%	98.83%	98.91%

where N_{right} is the number of those corrupted bits correctly restored and N_{total} is the total number of corrupted bits. For each noise density, we calculated the R values of different corrupted stego-images to find the average value. The results are listed in Table 3. We observe that all values are bigger than 99 percent, indicating a good restoration performance. In the experiments, we have used other data hiding methods [42], [43] to embed secret images and found that different methods have similar results.

3.3 Correcting the $5^{th} \sim 7^{th}$ Bits

A pixel value is mainly determined by its high bits. As the weights of low bits are small, the low bit values have small effect on the pixel value. Therefore, in this paper, we correct those unreliable values in the $5^{th} \sim 8^{th}$ bits and discard direct approximation on the $1^{st} \sim 4^{th}$ bits.

Let $n_1, n_2, \ldots, n_e (5 \leq n_1 < n_2 < ... < n_e \leq 8)$ be the reliable bit indices of p. If $|n_j - n_{j+1}| > 1 (j = 1, 2, \ldots, e-1)$, there is unreliable bit between the n_j-th bit and the n_{j+1}-th bit, i.e., $c(p,n)=1$ $(n_j < n < n_{j+1})$. Thus, we can correct the nth bit as follows.

We select the pixels from $p_1, p_2, \ldots, p_k$, which satisfy two conditions: (1) Their n_j-th, nth and n_{j+1}-th bits are all reliable. (2) Their n_j-th and n_{j+1}-th bits are equal to those of p. These pixels are then labeled as $p_{i_1}, p_{i_2}, ..., p_{i_w}$, where $w = \sum_{i=1}^{k} [1 - c(p_i, n_j)] [1 - |b(p_i, n_j) - b(p, n_j)|] [1 - c(p_i, n_{j+1})] [1 - |b(p_i, n_{j+1}) - b(p, n_{j+1})|] [1 - c(p_i, n)]$. We use the nth bits to correct the corresponding bit of p.

If $w \geq 3$ and $b(p_{i_1}, n) = b(p_{i_2}, n) = , \ldots, = b(p_{i_n}, n)$, there are some neighbor pixels whose high bits are the same with that of p. In this case, neighbor pixels have small difference. So let $b(p,n) = b(p_{i_1}, n)$ and $c(p,n)=0$. Table 4 is an example of unreliable bit approximation between two reliable bits, where '*' represents those bits with the same values.

If $w < 3$ or there exists $p_{i_j} (j = 2, 3, \ldots, w)$ whose nth bit is not equal to that of p_{i_1}, this means that p is not in a smooth region. So the approximation is not needed.

During the above restoration, calculation is performed in the 3×3 window. If the restoration is not achieved, we repeat the computation in the 5×5 window. If the conditions are still not satisfied, the estimation is discarded.

To test the performance of our strategy, we used different images to produce stego-images, added different salt and pepper noises to the stego-images, and then calculated restoration ratios of different corrupted stego-images to find the average ratio under different noise densities. The results are illustrated in Table 5. We observe that all R values are bigger than 98 percent, indicating good restoration performance.

3.4 Approximating Other Unreliable Bits

Generally, pixels in a local region are similar. Therefore, we calculate the maximum and the minimum possible values of

kNN pixels of p, and use them to approximate the probable value of p. To do so, for the kNN pixels of p in the 3×3 window, we set their unreliable bits to 1 and use m_{11}, m_{21},...,m_{81} to represent their values. Similarly, we set their unreliable bits to 0 and use m_{10}, m_{20},...,m_{80} to represent their values. Next, we use $M_1 = \min\{m_{11}, m_{21}, \ldots, m_{81}\}$ and $M_2 = \max\{m_{10}, m_{20}, \ldots, m_{80}\}$ to represent the minimum and the maximum values of $\{m_{11}, m_{21}$,...,$m_{81}\}$ and $\{ m_{10}, m_{20}$,...,$m_{80} \}$, respectively. If $M_1 \geq M_2$, let $h = M_1$ and $l = M_2$. Otherwise, we sort $\{m_{11}, m_{21}$,...,$m_{81} \}$ and $\{ m_{10}, m_{20}$,...,$m_{80} \}$. Let $\{m_{11}^{(1)}, m_{21}^{(1)}$,...,$m_{81}^{(1)} \}$ and $\{m_{10}^{(1)}, m_{20}^{(1)}$,...,$m_{80}^{(1)}\}$ be the sorted versions of $\{m_{11}, m_{21}$,...,$m_{81}\}$ and $\{m_{10}, m_{20}$,...,$m_{80}\}$ in ascending order, respectively. Also, let $i = 1$ and $j = 8$. If $m_{i1}^{(1)} \geq m_{j0}^{(1)}$, let $h = m_{i1}^{(1)}$ and $l = m_{j0}^{(1)}$. Otherwise, we increase the i value, i.e., $i = i + 1$, and judge the condition again. If $m_{i1}^{(1)} \geq m_{j0}^{(1)}$ is not satisfied, we decrease the j value, i.e., $j = j - 1$, and verify the condition again. We repeat the above operations until the condition is satisfied or $i = 8$ and $j = 1$.

Suppose that there are d bits of p for approximation. As each bit has two possible values (i.e., 1 or 0), the p value has $u = 2^d$ possible values. Let m_1, m_2,...,m_u be these u values in ascending order. We determine the p value in terms of the following conditions.

(1) If there are two values m_g and m_{g+1} satisfying $m_g < l$ and $m_{g+1} > h$, we choose the p value as follows. If $|m_g - l| < |m_{g+1} - h|$, let $p = m_g$. Otherwise, let $p = m_{g+1}$.

(2) If $m_1 > h$, let $p = m_1$.

(3) If $m_u > h$, let $p = m_u$.

(4) If there are two values m_g and m_v satisfying $m_g \geq l$ and $m_v \leq h$, we select the p value from $\{m_g, m_{g+1}$,..., $m_v\}$. Let the total number of possible values of p_i $(i = 1, 2, \ldots, 8)$ be q_i, and these values be f_{i1}, f_{i2},..., f_{iq_i}.Thus, we calculate $x_z = \sum_{i=1}^{8} \min_{1 \leq j \leq q_i}(|m_z - f_{ij}|^2)(z = g, g+1, \ldots, v)$. If there is only one minimum value $x_{\min}$ in $\{x_g$,..., $x_v\}$, its corresponding value m_i is the target value of p. Otherwise, we sort the corresponding values m_i and choose the middle one as the target value. Finally, we adjust those unreliable bits of p to approximate the target value by exhaustive search. The adjusted result with minimum difference between itself and the target value is viewed as the p value.

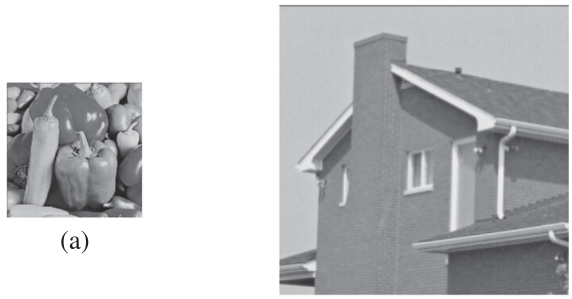
3.5 Detailed Steps

Detailed steps of our algorithm are as follows.

Bit-approximation algorithm

Input: A corrupted stego-image
Output: The recovered image extracted from the stego-image.

1. Identify the noisy pixels in the corrupted stego-image by the detection method [44], mark all extracted bits, and then reconstruct the embedded image.
2. For each pixel of the embedded image, we exploit our approach presented in Section 3.2 to correct its unreliable 8th bit.
3. For each pixel, we use our scheme given in Section 3.3 to recover its unreliable $5^{th} \sim 7^{th}$ bits between two reliable bits.
4. We apply our strategy introduced in Section 3.4 to estimating the values of noisy pixels in the embedded image.



(a) (b)

Fig. 3. Secret image and stego-image. (a)Secret image; (b)Stego-image.

4 Experimental Results

To view our performance, we validate efficiency of each step of the proposed algorithm in Section 4.1 and compare the proposed algorithm with some notable algorithms in Section 4.2. In these experiments, we take peak signal-to-noise ratio (PSNR) as the metric for objective evaluation, which is defined as

$$\mathrm{PSNR} = 10 \log_{10} \frac{L^2}{\mathrm{MSE}}, \tag{3}$$

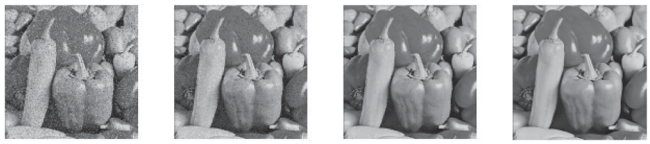
where L is the maximum possible pixel value of the image, equaling 255 for gray images, and MSE is the mean squared error (MSE) calculated by

$$\mathrm{MSE} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [I(i,j) - J(i,j)]^2, \tag{4}$$

where $I(i,j)$ and $J(i,j)$ are the pixel values in the ith row and the jth column of the original and the restored images sized $M \times N$, respectively. Since the aim of our algorithm is to recover the embedded image from corrupted stego-image, we select the well-known LSB substitution from diverse data hiding algorithms to conduct data embedding. In the experiments, the secret image is converted into a secure bit sequence for data embedding. More specifically, the pixels of secret image are scanned from left to right and top to bottom. For each pixel, it is decomposed into 8 bits, i.e., the 1st bit, 2nd bit, ..., 8th bit. Then, a bit sequence is obtained by concatenating the bits of all pixels. Finally, a pseudorandom generator controlled by a secret key is exploited to generate a secure bit sequence by scrambling the bit sequence. During data embedding, the secure bit sequence is concealed in the cover image by the LSB substitution. Therefore, direct extraction can reconstruct the embedded image by directly retrieving the used LSBs and re-scrambling them.

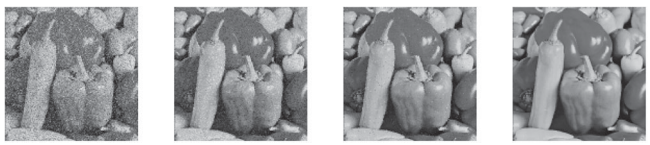
4.1 Step validation

To validate efficiency of each step of the proposed algorithm, we embed Fig. 3a sized 256×256 into Fig. 1a by using the two lowest bit-planes, and obtain the stego-image as shown in Fig. 3b. We add salt and pepper noise with different densities to Fig. 3b, and use the direction extraction and our schemes with different steps to restore the embedded images. Figs. 4, 5, 6, 7, and 8 are the recovery result comparisons among direction extraction and our schemes when noise densities are $0.1 \sim 0.5$, respectively. Table 6 presents



(a) (b) (c) (d)

Fig. 4. Recovery result comparisons among direction extraction and our schemes with different steps when the noise density is 0.1. (a)Direct extraction; (b)Steps $1 \sim 2$; (c)Steps $1 \sim 3$; (d)Steps $1 \sim 4$.

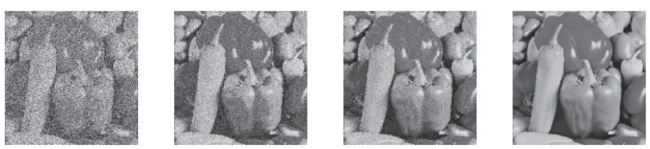


(a) (b) (c) (d)

Fig. 5. Recovery result comparisons among direction extraction and our schemes with different steps when the noise density is 0.2. (a) Direct extraction; (b) Steps $1 \sim 2$; (c) Steps $1 \sim 3$; (d)Steps $1 \sim 4$.

(a) (b) (c) (d)

Fig. 6. Recovery result comparisons among direction extraction and our schemes with different steps when the noise density is 0.3. (a)Direct extraction; (b)Steps $1 \sim 2$; (c)Steps $1 \sim 3$; (d)Steps $1 \sim 4$.



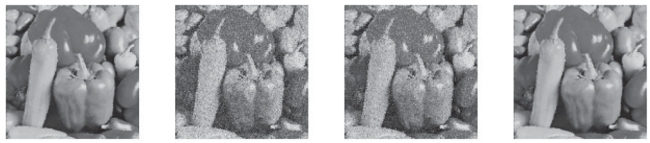
(a) (b) (c) (d)

Fig. 7. Recovery result comparisons among direction extraction and our schemes with different steps when the noise density is 0.4. (a)Direct extraction; (b)Steps $1 \sim 2$; (c)Steps $1 \sim 3$; (d)Steps $1 \sim 4$.

PSNR comparisons under different noise densities. From these results, we find that our schemes can produce better visual qualities of the restored images than the direct extraction, and further, more steps used will produce better visual quality. The benefits of our steps are more obvious when the noise density becomes high, such as 0.3, 0.4 and 0.5. These results illustrate the efficiency of our steps.

4.2 Performance Comparisons

To show advantage, we compare the proposed algorithm with some notable algorithms, i.e., direction extraction method, 3×3 median filter, and the methods [8], [9], [10], [11], [13], [14], [16], [17], [18], [24], [25], [26]. In the experiments, we embed different secret images into various cover images, add salt and pepper noise with different densities to the stego-images, and recover the embedded images from the corrupted stego-images. We find that visual qualities of the restored results produced by our algorithm are all better than those generated by compared algorithms. For space limitation, typical examples are given here. Fig. 1a is the cover image and Fig. 9 presents the used secret images,

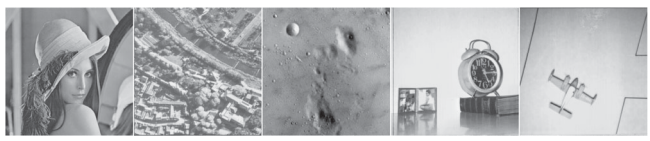


(a) (b) (c) (d)

Fig. 8. Recovery result comparisons among direction extraction and our schemes with different steps when the noise density is 0.5. (a)Direct extraction; (b)Steps $1 \sim 2$; (c)Steps $1 \sim 3$; (d)Steps $1 \sim 4$.

TABLE 6
PSNR Comparisons Among Direct Extraction and Our Schemes With Different Steps (Unit: dB)

Noise density	0.1	0.2	0.3	0.4	0.5
Direct extraction	17.8378	14.9415	13.2904	12.0760	11.2575
Steps $1 \sim 2$	23.1534	20.2246	18.4869	16.8903	15.6055
Steps $1 \sim 3$	27.7484	23.9190	21.3374	18.8154	16.7645
Steps $1 \sim 4$	34.8604	31.4969	29.4026	27.5101	26.3836



(a) (b) (c) (d) (e)

Fig. 9. Our used secret images sized 256×256. (a)Lena; (b)Aerial; (c) Moon surface; (d)Clock; (e)Airplane.

where Lena is the standard benchmark image, and other four images are downloaded from an open database [45]. Figs. 10, 11, and 12 illustrate recovery result comparisons when the noise densities are 0.1, 0.3 and 0.5, respectively. It is observed that, as noise density increases, visual qualities of the recovery results produced by the assessed methods significantly decrease. However, degradation of our algorithm is much slower than those of the compared algorithms. For example, our algorithm can make the decrypted image good visual quality even if the noise density is 0.5. Clearly, our algorithm is better than the compared methods in recovering embedded images from the corrupted stego-images.

To make quantitative comparisons, we calculate the PSNR and the normalized correlation (NC) between the original secret images and the recovered results by different methods, and obtain the average PSNR and NC under different noise densities. Tables 7 and 8 are the average PSNR comparisons and the average NC comparisons in recovering the five secret images, respectively. From Table 7, we find that the average PSNR values of our algorithm are all bigger than those of other methods. For example, as the noise density is 0.5, our PSNR value can reach 27.1229 dB, indicating acceptable visual quality. However, the biggest value of the compared methods is 19.9164 dB, which is much smaller than our value. Fig. 13 illustrates our incremental values of average PSNRs (i.e., our average PSNR subtracts the average PSNR of compared algorithm) under different noise densities. Similarly, it is observed from Table 8 that our average NC values are also bigger than those of the compared methods. Fig. 14 presents our incremental values of average NCs under different noise densities. Clearly, our algorithm has

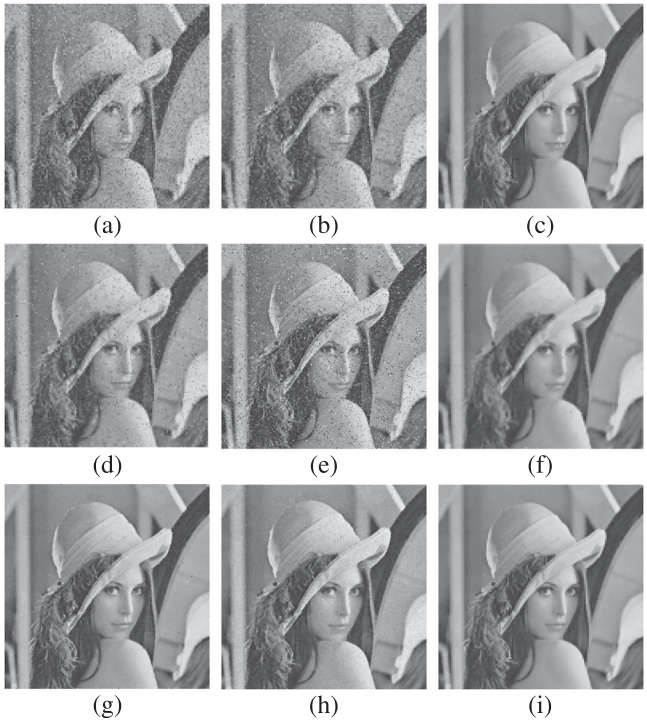


Fig. 10. Recovery result comparisons when the noise density is 0.1. (a) Direct extraction; (b)Wang and Wu [14]; (c)Zhang [16]; (d)Fabijańska *et al.* [17]; (e)Zhang and Li [18];(f)Gupta *et al.* [24]; (g)Varatharajan *et al.* [25];(h) Thangaraj *et al.* [26]; (i)Our algorithm.

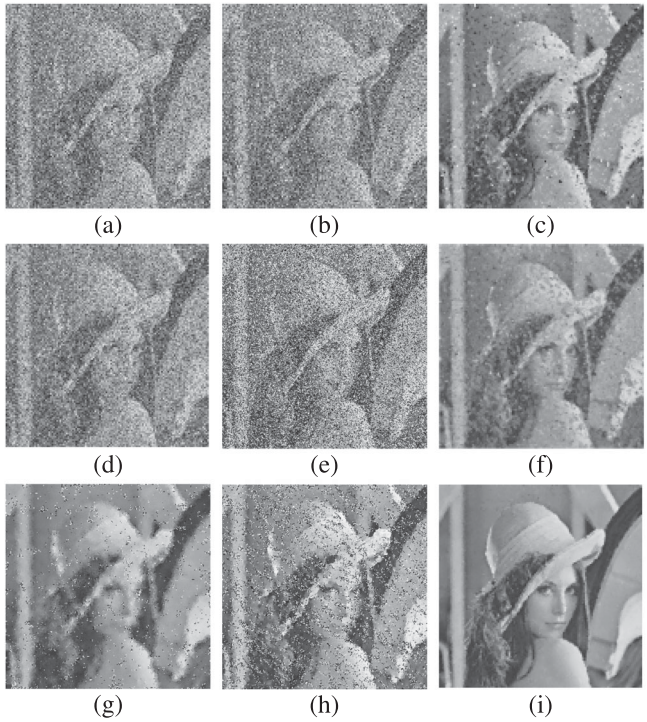


Fig. 12. Recovery result comparisons when the noise density is 0.5. (a) Direct extraction; (b)Wang and Wu [14]; (c)Zhang [16]; (d)Fabijańska *et al.* [17]; (e)Zhang and Li [18];(f)Gupta *et al.* [24]; (g)Varatharajan *et al.* [25];(h) Thangaraj *et al.* [26]; (i)Our algorithm.

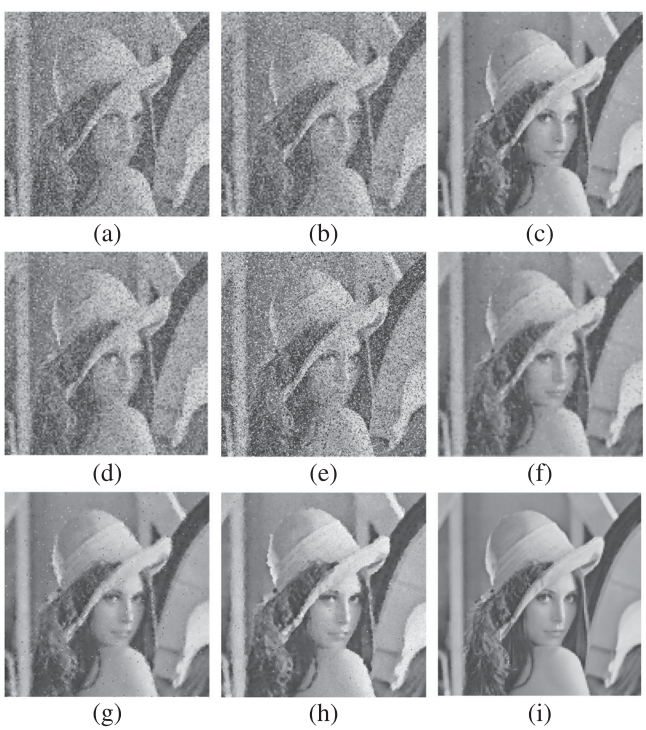


Fig. 11. Recovery result comparisons when the noise density is 0.3. (a) Direct extraction; (b)Wang and Wu [14]; (c)Zhang [16]; (d)Fabijańska *et al.* [17]; (e)Zhang and Li [18];(f)Gupta *et al.* [24]; (g)Varatharajan *et al.* [25];(h) Thangaraj *et al.* [26]; (i)Our algorithm.

TABLE 7
Average PSNR Comparisons in Recovering the Five Secret Images (unit: dB)

Noise density	0.1	0.2	0.3	0.4	0.5
Direct extraction	17.8811	14.9016	13.2278	12.1046	11.1921
3×3 median filter	28.8918	25.8369	22.6722	19.9648	17.6830
Sun and Neuvo [8]	27.3141	24.5912	22.6075	20.6667	18.6987
Alajlan *et al.* [9]	27.0077	22.6817	19.6944	17.4294	15.7217
Luo [10]	29.8004	26.6993	23.5272	20.7604	18.4147
Ibrahim *et al.* [11]	17.8929	14.9460	13.2860	12.1687	11.2519
Kang and Wang [13]	28.7329	26.8516	24.7315	22.2952	19.9164
Wang and Wu [14]	17.9263	14.9546	13.2896	12.1692	11.2515
Zhang [16]	30.1085	27.1530	24.4977	21.7742	19.1087
Fabijańska *et al.* [17]	21.4403	17.0836	14.8687	13.4471	12.3101
Zhang and Li [18]	19.9298	16.1917	14.2178	12.9200	11.8884
Gupta *et al.* [24]	27.1776	23.8251	21.0332	18.7895	16.8660
Varatharajan *et al.* [25]	31.4062	26.7968	24.4724	22.6173	19.0042
Thangaraj *et al.* [26]	26.8764	24.0064	22.2065	18.9441	14.3557
Our algorithm	36.1034	32.6837	30.5443	28.8304	27.1229

significant improvement on visual quality with respect to PSNR and NC. Our algorithm outperforms the compared algorithms. This can be understood as follows. When the stego-image is contaminated, some bits per pixel in the embedded image are altered and other bits are still preserved. In our algorithm, these preserved bits are exploited to estimate the values of the dirty pixels. They give us more details to recover the actual values and thus make good visual quality of the restored image. For the compared algorithms, the dirty pixels are simply replaced without the use of these preserved bits and consequently the visual quality of the recovered image is not good enough.

Computational time of the assessed algorithms is also evaluated. All the algorithms are implemented with MATLAB R2011a, and run on a desktop computer with 3.40 GHz Intel Core i5-3570 CPU and 16 GB RAM. The operating

TABLE 8
Average NC Comparisons Among Different Methods

Noise density	0.1	0.2	0.3	0.4	0.5
Direct extraction	0.7461	0.5934	0.4795	0.3897	0.3082
3×3 median filter	0.9719	0.9463	0.8910	0.8059	0.6939
Sun and Neuvo [8]	0.9604	0.9295	0.8916	0.8346	0.7514
Alajlan *et al.* [9]	0.9578	0.8903	0.7943	0.6788	0.5615
Luo [10]	0.9810	0.9550	0.9078	0.8334	0.7284
Ibrahim *et al.* [11]	0.7420	0.5910	0.4795	0.3904	0.3092
Kang and Wang [13]	0.9706	0.9555	0.9287	0.8783	0.7990
Wang and Wu [14]	0.7474	0.5937	0.4812	0.3913	0.3096
Zhang [16]	0.9792	0.9591	0.9259	0.8679	0.7698
Fabijańska *et al.* [17]	0.8646	0.6993	0.5694	0.4628	0.3685
Zhang and Li [18]	0.8162	0.6517	0.5279	0.4287	0.3396
Gupta *et al.* [24]	0.9442	0.8826	0.7947	0.6912	0.5662
Varatharajan *et al.* [25]	0.9846	0.9551	0.9233	0.8796	0.7502
Thangaraj *et al.* [26]	0.9638	0.9476	0.9236	0.8122	0.8475
Our algorithm	0.9950	0.9878	0.9826	0.9744	0.9625

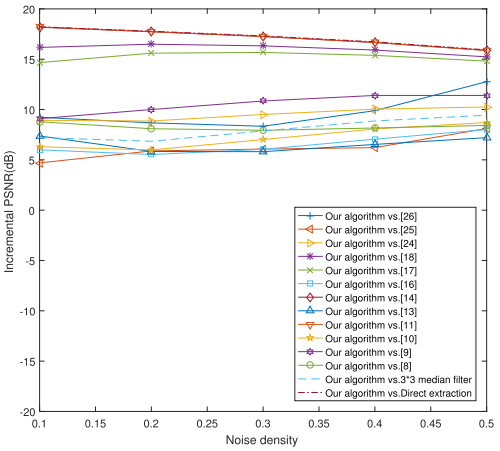


Fig. 13. Our incremental values of average PSNRs under different noise densities.

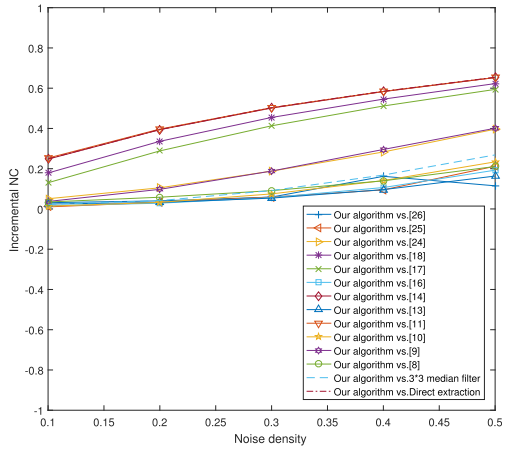


Fig. 14. Our incremental values of average NCs under different noise densities.

TABLE 9
Time Comparisons Among Different Methods (unit: second)

Noise density	0.1	0.2	0.3	0.4	0.5
Direct extraction	2.991	2.915	2.883	2.897	2.914
3×3 median filter	2.952	2.923	2.868	2.934	2.917
Sun and Neuvo [8]	4.132	4.114	4.139	4.144	4.167
Alajlan *et al.* [9]	3.669	3.693	3.688	3.670	3.675
Luo [10]	6.316	6.320	6.432	6.365	6.405
Ibrahim *et al.* [11]	3.136	3.149	3.181	3.134	3.173
Kang and Wang [13]	4.859	4.738	4.755	4.736	4.706
Wang and Wu [14]	3.068	3.071	3.091	3.101	3.122
Zhang [16]	12.996	13.310	14.145	21.836	16.198
Fabijańska *et al.* [17]	10.359	10.127	10.106	10.204	10.240
Zhang and Li [18]	12.459	13.567	13.789	14.522	14.632
Gupta *et al.* [24]	4.786	4.799	4.804	4.754	4.758
Varatharajan *et al.* [25]	4.4625	6.7773	10.6944	14.5010	16.7766
Thangaraj *et al.* [26]	6.7620	10.0925	12.6117	14.4545	14.9750
Our algorithm	14.117	29.016	63.806	159.534	427.367

system installed on the desktop computer is Windows 10. We record the processing time of the assessed algorithms for recovering Airplane from the corrupted stego-images, and obtain the results as shown in Table 9. It is observed that our speed is a little slower than those of the compared algorithms (such as [16], [17], and [18]) when the noise density is 0.1. As the noise density increases, our algorithm becomes very slow. The low speed of our algorithm can be understood as follows. (1) When bit approximation is not achieved, iterative computation in a large window is needed. (2) There are several loops in our implementation code, which is time-consuming since MATLAB is inefficient in processing loop. Note that visual quality of the recovered image is the first performance of the assessed algorithms and computational time is the second one. In practice, the time-consumed problem can be overcome by high performance computing (HPC).

5 Conclusion

In this paper, we have proposed an efficient algorithm, called bit-approximation algorithm, for noise removal in embedded image. The extracted bits of each pixel are first marked by the detection method. The unreliable bits are corrected with three different strategies according to their locations in pixel. Different from existing pixel estimation methods, the proposed bit-approximation efficiently utilizes reliable bits of pixels, so as to improve the visual quality of decrypted images. Sets of experiments have been conducted to validate the performance of our algorithm, and showed that the proposed approach has better performance than some well-known filtering methods in embedded image recovery from corrupted stego-images. Research on embedded image recovery is still under way. In the future, we will focus on embedded image recovery when secret images are embedded by different data hiding methods (such as pixel-value differencing based method and EMD (exploiting modification direction) based method), embedded image recovery under other attacks (such as filtering and smoothing), and so on.

Acknowledgments

The authors would like to thank the reviewers and the editor for their helpful comments and suggestions. This work was partially supported by the National Natural Science Foundation of China (61836016, 61762017, 61962008, 61871470, U1801262), Guangxi "Bagui Scholar" Team for Innovation and Research, Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing, and the Guangxi Talent Highland Project of Big Data Intelligence and Application.

REFERENCES

[1] J. Wang, J. Ni, X. Zhang, and Y. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 315–326, Feb. 2017.

[2] D. Hu, D. Zhao, and S. Zheng, "A new robust approach for reversible database watermarking with distortion control," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1024–1037, Jun. 2019.

[3] Z. Tang, Q. Lu, H. Lao, C. Yu, and X. Zhang, "Error-free reversible data hiding with high capacity in encrypted image," *Optik - Int. J. Light Electron Optics*, vol. 157, pp. 750–760, 2018.

[4] L. An, X. Gao, X. Li, D. Tao, C. Deng, and J. Li, "Robust reversible watermarking via clustering and enhanced pixel-wise masking," *IEEE Trans. Image Process.*, vol. 21, no. 8, pp. 3598–3611, Aug. 2012.

[5] H. Duan and X. Wang, "Echo state networks with orthogonal pigeon-inspired optimization for image restoration," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2413–2425, Nov. 2016.

[6] J. Jiang, L. Zhang, and J. Yang, "Mixed noise removal by weighted encoding with sparse nonlocal regularization," *IEEE Trans. Image Process.*, vol. 23, no. 6, pp. 2651–2662, Jun. 2014.

[7] D. H. Dini and D. P. Mandic, "Class of widely linear complex Kalman filters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 775–786, May 2012.

[8] T. Sun and Y. Neuvo, "Detail-preserving median based filters in image processing," *Pattern Recognit. Lett.*, vol. 15, no. 4, pp. 341–347, 1994.

[9] N. Alajlan, M. Kamel, and Ed Jernigan, "Detail preserving impulsive noise removal," *Signal Process.: Image Commun.*, vol. 19, no. 10, pp. 993–1003, 2004.

[10] W. Luo, "An efficient algorithm for the removal of impulse noise from corrupted images," *AEU-Int. J. Electron. Commun.*, vol. 61, no. 8, pp. 551–555, 2007.

[11] H. Ibrahim, N. S. P. Kong, and T. F. Ng, "Simple adaptive median filter for the removal of impulse noise from highly corrupted Images," *IEEE Trans. Consum. Electron.*, vol. 54, no. 4, pp. 1920–1927, Nov. 2008.

[12] P. Chen and C. Lien, "An efficient edge-preserving algorithm for removal of salt-and-pepper noise," *IEEE Signal Process. Lett.*, vol. 15, no. 12, pp. 833–836, Dec. 2008.

[13] C. Kang and W. Wang, "Modified switching median filter with one more noise detector for impulse noise removal," *AEU-Int. J. Electron. Commun.*, vol. 63, no. 11, pp. 998–1004, 2009.

[14] S. Wang and C. Wu. "A new impulse detection and filtering method for removal of wide range impulse noises," *Pattern Recognit.*, vol. 42, no. 9, pp. 2194–2202, 2009.

[15] Y. Wan, Q. Chen, and Y. Yang, "Robust impulse noise variance estimation based on image histogram," *IEEE Signal Process. Lett.*, vol. 17, no. 5, pp. 485–488, May 2010.

[16] J. Zhang, "An efficient median filter based method for removing random-valued impulse noise," *Digital Signal Process.*, vol. 20, no. 4, pp. 1010–1018, 2010.

[17] A. Fabijańska and D. Sankowski, "Noise adaptive switching median-based filter for impulse noise removal from extremely corrupted images," *IET Image Process.*, vol. 5, no. 5, pp. 472–480, 2011.

[18] P. Zhang and F. Li, "A new adaptive weighted mean filter for removing salt-and-pepper noise," *IEEE Signal Process. Lett.*, vol. 21, no. 10, pp. 1280–1283, Oct. 2014.

[19] F. Ahmed and S. Das, "Removal of high-density salt-and-pepper noise in images with an iterative adaptive fuzzy filter using Alpha-trimmed mean," *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 5, pp. 1352–1358, Oct. 2014.

[20] T. Bai, J. Tan, M. Hu, and Y. Wang, "A novel algorithm for removal of salt and pepper noise using continued fractions interpolation," *Signal Process.*, vol. 10, no. 2, pp. 247–255, 2014.

[21] S. Pyatykh and J. Hesser, "Salt and pepper noise removal in binary images using image block prior probabilities," *J. Vis. Commun. Image Representation*, vol. 25, no. 5, pp. 748–754, 2014.

[22] C. Sun, C. Tang, X. Zhu, X. Li, and L. Wang, "An efficient method for salt-and-pepper noise removal based on shearlet transform and noise detection," *AEU-Int. J. Electron. Commun.*, vol. 69, no. 12, pp. 1823–1832, 2015.

[23] X. Zhang, F. Ding, Z. Tang, and C. Yu, "Salt and pepper noise removal with image inpainting," *AEU-Int. J. Electron. Commun.*, vol. 69, no. 1, pp. 307–313, 2015.

[24] V. Gupta, V. Chaurasia, and M. Shandilya, "Random-valued impulse noise removal using adaptive dual threshold median filter," *J. Vis. Commun. Image Representation*, vol. 2, no. 6, pp. 296–304, 2015.

[25] R. Varatharajan, K. Vasanth, M. Gunasekaran, M. Priyan, and X. Gao, "An adaptive decision based kriging interpolation algorithm for the removal of high density salt and pepper noise in images," *Comput. Elect. Eng.*, vol. 70, no. 1, pp. 447–461, 2018.

[26] V. Thangaraj, N. Badri, E. Sankaralingam, and K. Prasanta, "Context model based edge preservation filter for impulse noise removal," *Expert Syst. Appl.*, vol. 88, no. 2, pp. 29–44, 2017.

[27] Z. Li, Y. Cheng, K. Tang, Y. Xu, and D. Zhang, "A salt & pepper noise filter based on local and global image information," *Neurocomputing*, vol. 15, no. 9, pp. 172–185, 2015.

[28] J. Pan *et al.*, "Learning dual convolutional neural networks for low-level vision," in *Proc. IEEE & CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3070–3079.

[29] J. Pan *et al.*, "Physics-based generative adversarial models for image restoration and beyond," *IEEE Trans. Pattern Anal. Mach. Intell.*, [Online]. Available: ,https://ieeexplore.ieee.org/document/8968618

[30] C. Chen, L. Liu, L. Chen, Y. Tang, and Y. Zhou, "Weighted couple sparse representation with classified regularization for impulse noise removal," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4014–4026, Nov. 2015.

[31] T. Huang, W. Dong, X. Xie, G. Shi, and X. Bai, "Mixed noise removal via Laplacian scale mixture modeling and nonlocal low-rank approximation," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3171–3186, Jul. 2017.

[32] H. Khaw, F. Soon, J. Chuah, and C. Chow, "High-density impulse noise detection and removal using deep convolutional neural network with particle swarm optimisation," *IET Image Process.*, vol. 13, no. 2, pp. 365–374, 2018.

[33] R. Wang, M. Pakleppa, and E. Trucco, "Low-rank prior in single patches for nonpointwise impulse noise removal," *IEEE Trans. Image Process.*, vol. 24, no. 5, pp. 1485–1496, May 2015.

[34] F. Wang, H. Huang, and J. Liu, "Variational-based mixed noise removal with CNN deep learning regularization," *IEEE Trans. Image Process.*, vol. 29, pp. 1246–1258, 2020.

[35] Y. Xing, J. Xu, J. Tan, D. Li, and W. Zha, "Deep CNN for removal of salt and pepper noise," *IET Image Process.*, vol. 13, no. 5, pp. 1550–1560, 2019.

[36] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[37] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4608–4622, Sep. 2018.

[38] W. Dong, H. Wang, F. Wu, G. Shi, and X. Li, "Deep spatial-pectral representation learning for hyperspectral image denoising," *IEEE Trans. Comput. Imaging*, vol. 5, no. 4, pp. 635–648, Dec. 2019.

[39] S. Zhang, "Nearest neighbor selection for iteratively kNN imputation," *J. Syst. Softw.*, vol. 85, no. 11, pp. 2541–2552, 2012.

[40] Q. Liu, and C. Liu, "A novel locally linear KNN method with applications to visual recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 2010–2021, Sep. 2017.

[41] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient kNN classification with different numbers of nearest neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1774–1785, May 2018.

[42] C. K. Chan and L. M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognit.*, vol. 37, no. 3, pp. 469–474, 2004.

[43] C. F. Lee and H. L. Chen, "A novel data hiding scheme based on modulus function," *J. Syst. Softw.*, vol. 83, no. 5, pp. 832–843, 2010.

[44] X. Wang, X. Zhao, F. Guo, and J. Ma, "Impulsive noise detection by double noise detector and removal using adaptive neural-fuzzy inference system," *AEU-Int. J. Electron. Commun.*, vol. 65, no. 5, pp. 429–434, 2011.

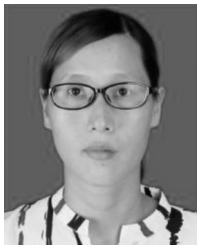
[45] USC-SIPI Image Database. 2017. [Online]. Available: Available: http://sipi.usc.edu/database/

Xianquan Zhang received the MEng degree from Chongqing University, Chongqing, P. R. China. He is currently a professor with the Department of Computer Science, Guangxi Normal University. His research interests include image processing and data hiding. He has contributed more than 100 papers.

Xuelong Li (Fellow, IEEE) is a full professor with the School of Computer Science and Center for Optical Imagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, P.R. China.

Zhenjun Tang (Member, IEEE) received the BS and MEng degrees from Guangxi Normal University, Guilin, P. R. China, in 2003 and 2006, respectively, and the PhD degree from Shanghai University, Shanghai, P. R. China, in 2010. He is now a professor with the Department of Computer Science, Guangxi Normal University. His research interests include image processing and multimedia security. He has contributed more than 60 international journal papers. He is a reviewer of more than 30 SCI journals, such as IEEE/IET journals, Elsevier journals and Springer journals.

Shichao Zhang (Senior Member, IEEE) received the PhD degree in computer science from Deakin University, Geelong, VIC, Australia. He is currently a distinguished professor with the Department of Computer Science, Guangxi Normal University, Guilin, P.R. China. He has authored more than 60 international journal papers and 70 international conference papers. His current research interests include data quality and pattern discovery. He is a member of the association for Computing Machinery. As a chief investigator, he has won four Australian Large ARC Grants, three China 863 Programs, two China 973 Programs, and five NSFs of China Grants. He served/serves as an associate editor of the *IEEE Transactions on Knowledge and Data Engineering, Knowledge and Information Systems*, and the *IEEE Intelligent Informatics Bulletin*. He served as a PC chair or Conference chair for six international conferences.



Shaomin Xie received the MEng degree from Guangxi Normal University, Guilin, P.R. China, in 2012. Her research interests include image processing and data hiding.