Hindawi Publishing Corporation
Advances in Multimedia
Volume 2016, Article ID 4901609, 17 pages
http://dx.doi.org/10.1155/2016/4901609

Research Article

A Novel Printable Watermarking Method in Dithering Halftone Images

Hui-Lung Lee and Ling-Hwei Chen

Department of Computer Science, National Chiao Tung University Hsinchu, Hsinchu 300, Taiwan

Correspondence should be addressed to Ling-Hwei Chen; lhchen@cc.nctu.edu.tw

Received 11 January 2016; Revised 5 April 2016; Accepted 3 May 2016

Academic Editor: Martin Reisslein

Halftone images are commonly printed on books, newspapers, and magazines. How to protect the copyright of these printed halftone images becomes an important issue. Digital watermarking provides a solution for copyright protection. In this paper, we will propose a novel printable watermarking method for dithering halftone images. Based on downsampling and the property of a dispersed dithering screen, the method can resist cropping, tampering, and print-and-scan process attacks. In addition, comparing to Guo et al.'s method, the experimental results show that the proposed method provides higher robustness for the above-mentioned attacks and better visual quality in the high-frequency regions of halftone images.

1. Introduction

Digital halftoning is a method to convert continuous-tone images to two-tone ones; it is widely used in printing newspapers, magazines, books, and so forth. When viewed from a proper distance, halftone images resemble the original grayscale images. Today, many digital halftoning methods [1–3] were developed. Error diffusion, ordered dithering, and iteration-based technique are three common types of halftoning methods. Error diffusion [1] is a single-pass sequential algorithm. The past error is diffused back to the unprocessed neighboring pixels. When processing the current pixel, its gray value will add all the past error and compare with a fixed threshold 128 to determine its output. Ordered dithering [2] is applied to a threshold matrix to convert a gray image to a halftone image. It compares the pixel value with the threshold matrix to determine pixel output. Hence, it has better time efficiency. The iteration-based technique [3] is an iterative algorithm; it proceeds by generating an initial halftone image and then iteratively performs a local search on the halftone space by swapping and toggling to minimize the perceived error. It usually generates a better quality images than the error diffusion and the ordered dithering, but it is time consuming.

Many watermarking techniques have been provided for halftone image copyright protection and authentication; they are divided into three categories: error diffusion based [4–11], ordered dithering based [4, 12–18], and iteration-based [19–21] techniques.

For error diffusion based watermark techniques, in 2002, Fu and Au [4] proposed a few data hiding or watermarking methods in halftone images. When the original multitone image is not available, a data hiding smart pair toggling scheme was presented to hide data in halftone images. When the original multitone image is available and the halftoning method is error diffusion, a modified data hiding error diffusion method was provided to hide data in the halftone images by forced self-toggling with its distortion diffused to the surrounding pixels. This method can be applied to the halftone images generated by ordered dithering. Experimental results show that the proposed methods have high data hiding capacity, low computational complexity, good visual quality, and reasonable resistance toward noise. However, it is not robust to any distortions without applying error correction code, and some artifacts are found in the locations of pair toggling. Later, Fu and Au [5] embedded a single watermark or multiwatermarks in the parity domain of a halftone image

during halftoning. However, it can only embed 1 bit to indicate whether a watermark or one of two watermarks exists. Thus, this method is improper for copyright authentication. In 2004, Fu and Au [6] proposed an improved method to embed a watermark in the local correlation coefficient between the watermark bits and the halftone image. The local correlation coefficient is computed by the exclusive operation between a security code and halftone image. However, if the security code size is small, the visual quality of the watermarked halftone image would be degraded. In 2006, Pei and Guo [7] proposed a data hiding method in several halftone images or color planes using minimal-error bit searching. It employed the gray code to divide code vectors into two groups, and each group corresponds to a watermark bit 0 or 1. According to the watermark bit embedded, most suitable code vectors with better visual quality are chosen to form the watermarked halftone images. However, the quality degrades significantly when capacity increases up to 50%. In 2007, Li et al. [8] proposed a watermarking method for error diffusion halftone images; the method provides a block-overlapping parity check algorithm to reduce the number of pair toggling required in the Fu and Au's method [4]. Experiments show that the method has better visual quality than Fu and Au's method [4]. To treat Pei and Guo's problem [7], Guo and Liu [9] proposed a data hiding method in several halftone images by using overall minimal-error searching and secret sharing. Moreover, the least-mean-square based scheme is also employed to achieve even better quality and edge enhanced embedded results. However, the homogeneous regions of watermarked images maybe have artifacts.

For ordered dithering based watermark techniques, in 2001, Fu and Au [14] proposed a two-phase watermarking method for ordered dithering images. First, one out of every M pseudorandom locations is selected using threshold selection to embed one data bit. Second, screen modification is applied to the local neighborhood of the selected location to change the ordered dithering screen to achieve the desired data embedding. Some quality measures were proposed to evaluate the visual quality of a dithering image. Simulation results show that the method can hide a large amount of data while maintaining good visual quality. However, it is not robust to the print-and-scan process. In 2001, Hel-Or [13] proposed a method to embed a watermark in printed images. First, based on watermark bits, a dithering screen is created by selecting different dither matrices; then the screen is used to produce a printed image. The method is robust under reconstruction errors. However, it is not robust to cropping and has artifacts.

In 2005, Pei et al. [15] presented a method to embed watermarks into dithered halftone images. The method divides a dithered halftone image into several sub-subimages by the bit and sub-subimage interleaving preprocesses, and each watermark bit is embedded into each pair of sub-subimages. The method has low computational complexity and flexible embedding capacity. But the method requires the knowledge of the original watermark to do copyright authentication. To treat Pei et al.'s problem, in 2008, Guo et al. [12] proposed another watermarking method using the blind paired subimage matching ordered dithering (BPSMOD) technique. It does not require the knowledge of the original watermark in the watermark extraction. However, the visual quality in the boundary of the embedded dithered image may be degraded.

In 2010, Bulan et al. [16] proposed a data hiding method that embeds bits through clustered-dot orientations during halftoning process. For extracting the embedding data, a moment-based extracting method is used to detect the clustered-dot orientations. The method is only applicable for clustered-dot halftoning methods, and it relies on the ability to accurately control the printing of the halftone image; this may be restrictive in some applications. In 2013, Feng et al. [17] proposed a halftone watermarking algorithm based on particle swarm optimization. It is robust under smearing and cropping attacks. Unfortunately, it needs mean filtering and median filtering to remove noise from the recovered watermark image. This is only suitable for a watermark with a solid black/white object. Thus, the method is unsuitable for a watermark with a random sequence.

For iteration-based watermark techniques, in 2003, Chun and Ha [19] proposed a watermark technique based on iterative halftoning method. In the embedding stage, a pseudorandom number generator is used to locate the embedding locations and force these pixel values at these locations to be 0 or 1 according to the watermark bits. Then, in the error minimizing stage, for each unembedded pixel, check whether toggling the pixel value or swapping the pixel value with neighbor pixels can reduce the perceived halftoning error. In 2012, Guo et al. [20] proposed a DBS-based orientation modulation watermarking method. In this method, the direction of the point spread function is used to represent different watermark values. To extract the watermark bit, the LMS trained filters and naive Bayes classifier are used to classify the angle. In 2015, Guo et al. [21] proposed a halftoning-based multilayer watermarking method. An efficient direct binary search and lookup table method is applied to embed multiple watermarks. Then, the least mean square and native Bayes classifier are used to extract the watermarks. Although all these methods provide excellent image quality, they are time consuming.

In this paper, we focus on ordered dithering halftone images. And a blind watermarking method will be proposed to treat the disadvantages of the above-mentioned dithering based watermarking methods. First, a grayscale image is transformed into a dithering halftone image according to an $n \times n$ dispersed dithering screen (for convenience of illustration, here, we take $n = 4$; see Figure 1); then the halftone image is divided into several subimages through downsampling. For each subimage, to embed watermark bits, it is first divided into several 4×4 blocks. Then, the number of black pixels, p_j, in all blocks corresponding to the position with the jth smallest value T_j (see Figure 1(c), $j = 3$, $T_3 = 56$) in the 4×4 dispersed dithering screen is counted. Finally, we take (p_u, p_v) as a pair to embed a bit based on the sign of $(p_u - p_v)$, where $(u, v) = (0, 4), (1, 5), (2, 6), (3, 7), (8, 15), (9, 14), (10, 12)$, or $(11, 13)$. If the embedding bit is 0 and $p_u < p_v$ or the embedding bit is 1 and $p_u > p_v$, nothing is done. Otherwise, in each block, the pixel at position with the uth smallest value and the pixel at position with the vth smallest value are swapped. Since $|p_u - p_v|$ is usually larger than $|p_u - p_{u+1}|$, this provides higher robustness than Guo et al.'s method [12]

24	232	40	200
152	88	168	104
56	216	8	248
184	120	136	72

(a)

6	238	62	222	10	226	50	210
134	70	190	126	138	74	178	114
38	198	22	254	42	202	26	242
166	102	150	86	170	106	154	90
14	230	54	214	2	234	58	218
142	78	182	118	130	66	186	122
46	206	30	246	34	194	18	250
174	110	158	94	162	98	146	82

(b)

1 (0, 0)	14 (0, 1)	2 (0, 2)	12 (0, 3)
9 (1, 0)	5 (1, 1)	10 (1, 2)	6 (1, 3)
3 (2, 0)	13 (2, 1)	0 (2, 2)	15 (2, 3)
11 (3, 0)	7 (3, 1)	8 (3, 2)	4 (3, 3)

(c)

1 (0, 0)	59 (0, 1)	15 (0, 2)	55 (0, 3)	2 (0, 4)	56 (0, 5)	12 (0, 6)	52 (0, 7)
33 (1, 0)	17 (1, 1)	47 (1, 2)	31 (1, 3)	34 (1, 4)	18 (1, 5)	44 (1, 6)	28 (1, 7)
9 (2, 0)	49 (2, 1)	5 (2, 2)	63 (2, 3)	10 (2, 4)	50 (2, 5)	6 (2, 6)	60 (2, 7)
41 (3, 0)	25 (3, 1)	37 (3, 2)	21 (3, 3)	42 (3, 4)	26 (3, 5)	38 (3, 6)	22 (3, 7)
3 (4, 0)	57 (4, 1)	13 (4, 2)	53 (4, 3)	0 (4, 4)	58 (4, 5)	14 (4, 6)	54 (4, 7)
35 (5, 0)	19 (5, 1)	45 (5, 2)	29 (5, 3)	32 (5, 4)	16 (5, 5)	46 (5, 6)	30 (5, 7)
11 (6, 0)	51 (6, 1)	7 (6, 2)	61 (6, 3)	8 (6, 4)	48 (6, 5)	4 (6, 6)	62 (6, 7)
43 (7, 0)	27 (7, 1)	39 (7, 2)	23 (7, 3)	40 (7, 4)	24 (7, 5)	36 (7, 6)	20 (7, 7)

(d)

Figure 1: An example of dispersed dithering screens. (a) A 4×4 dispersed dithering screen. (b) An 8×8 dispersed dithering screen. (c) The order and position of each pixel in (a). (d) The order and position of each pixel in (b).

for the print-and-scan process. In addition, since the number of black pixels in each block is not changed, the proposed method also provides higher visual quality in the edge boundary than Guo et al.'s method [12]. Furthermore, the downsampling technique is presented to provide higher robustness than Guo et al.'s method for cropping and tampering. The rest of the paper is organized as follows. Section 2 outlines Guo et al.'s [12, 15] methods. Section 3 describes the proposed method. Section 4 shows the experimental results and comparisons. Section 5 draws conclusions.

2. A Brief Description for Guo et al.'s Methods

As mentioned previously, Guo et al. [12] proposed a watermarking method using BPSMOD in dithering halftone images. At first, a dispersed dithering method is adopted to convert a grayscale image into a dithering halftone image. Then, the bit-interleaving algorithm proposed by Pei et al. [15] is used to arrange the dithering halftone image. After that, the BPSMOD is applied to the arranged image to embed watermark bits. To raise the embedding capacity, a sub-subimage

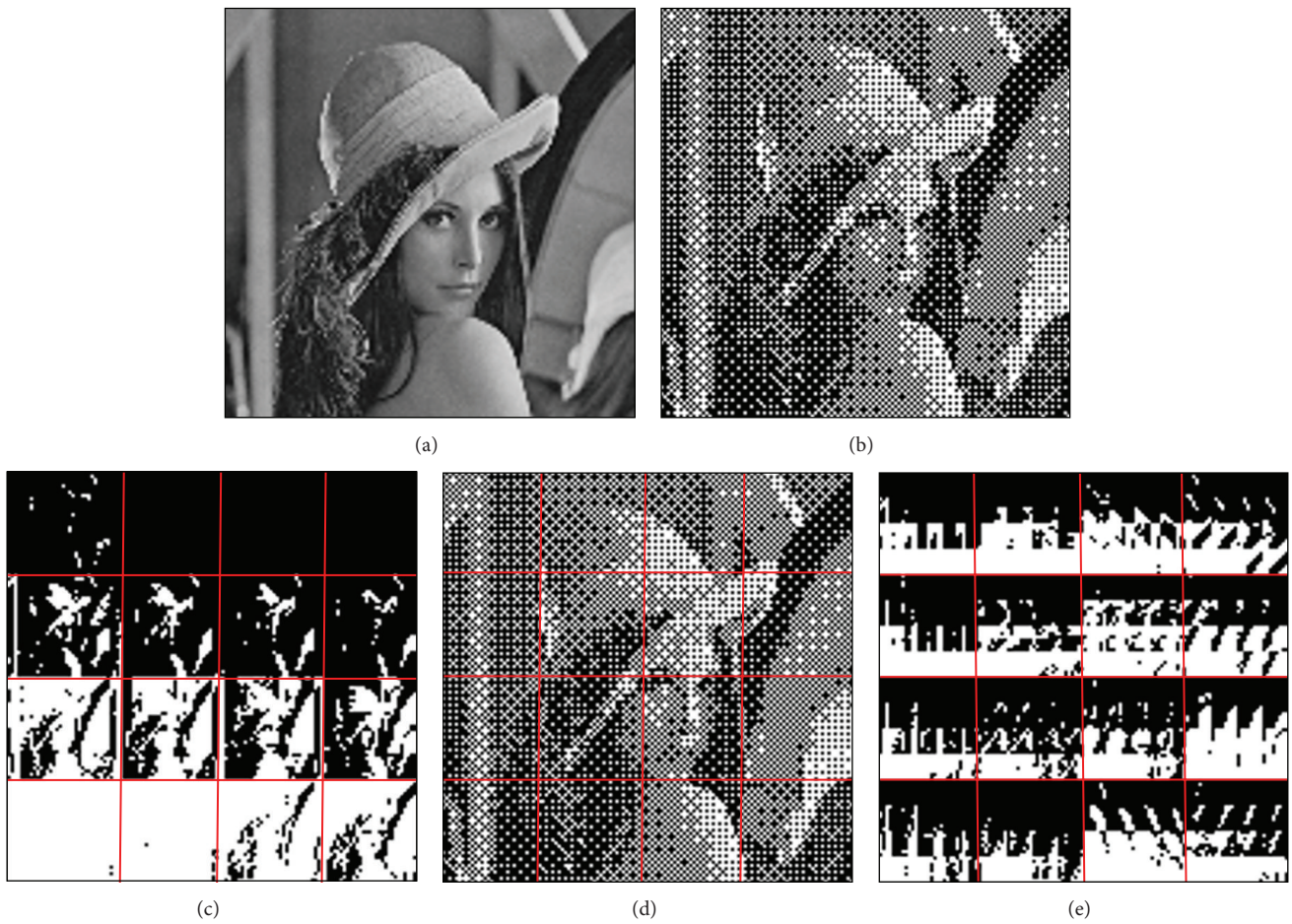


Figure 2: An example of bit-interleaving algorithm [12]. (a) Lena image. (b) Dithering halftone image of (b) using Figure 1(a). (c) Bit-interleaving result of (b). (d) Dithering halftone image divided into 16 subimages. (e) The result of applying bit-interleaving to each subimage in (d).

interleaving algorithm [12, 15] is adopted. The details are described in the following subsections.

2.1. Bit-Interleaving Algorithm in a Dithering Halftone Image. As mentioned above, in Guo et al.'s method [12], an $n \times n$ dispersed dithering screen (DS) [1] is first applied to a $P \times Q$ grayscale image I to result in a dithering halftone image H according to the following equation:

$$H(i,j) = \begin{cases} 0, & x(i,j) < \mathrm{DS}\ (i \bmod n, j \bmod n), \\ 255, & \text{otherwise}, \end{cases} \quad (1)$$

where $x(i, j)$ and $H(i, j)$ are the gray levels of pixel (i, j) in I and H, respectively. $\mathrm{DS}(i, j)$ is the value of the position (i, j) in the dispersed dithering screen. Figure 2(a) shows a 128×128 grayscale image. Figure 2(b) is the resulting dithering halftone image by applying Figure 1(a) to Figure 2(a).

After obtaining the dithering halftone image, all pixels corresponding to the same screen value are then grouped into a subimage; this will result in $n \times n$ subimages, each of which has $P/n \times Q/n$ pixels. Finally, according to the screen values, all subimages will be sorted in ascending order of the screen values and arranged from left to right and bottom to top to result in a binary image RH. Let RH_j be the jth subimage, where $j = 0, 1, \ldots, (n \times n) - 1$, and the bottom-left subimage is RH_0. The above process is called bit-interleaving [12, 15]. Figure 2(c) shows the bit-interleaving result of Figure 2(b), and it contains 16 subimages, and the bottom-left subimage corresponds to screen value 8.

2.2. BPSMOD. In BPSMOD, based on RH, subimages RH_j and RH_{j+1} are considered a pair (RH_j, RH_{j+1}), where $j = 2m$ and $m = 0, 1, \ldots, (n \times n)/2 - 1$. Since the screen value used to form RH_j is smaller than that used to form RH_{j+1}, RH_j will usually have less black pixels than RH_{j+1}. However, RH_j sometimes has black pixels more than (equal to) RH_{j+1}; this kind of pairs is called nonincreased black pixel pair (NIP). Before embedding, if a pair (RH_j, RH_{j+1}) is a NIP, it will be modified by increasing black pixels of RH_{j+1} or decreasing black pixels of RH_j to make the modified RH_j have less black pixels than the modified RH_{j+1}. Then, in embedding, each pair (RH_j, RH_{j+1}) will embed 1 bit. If watermark bit being embedded is 1, RH_j and RH_{j+1} will be swapped. Otherwise, nothing is done.

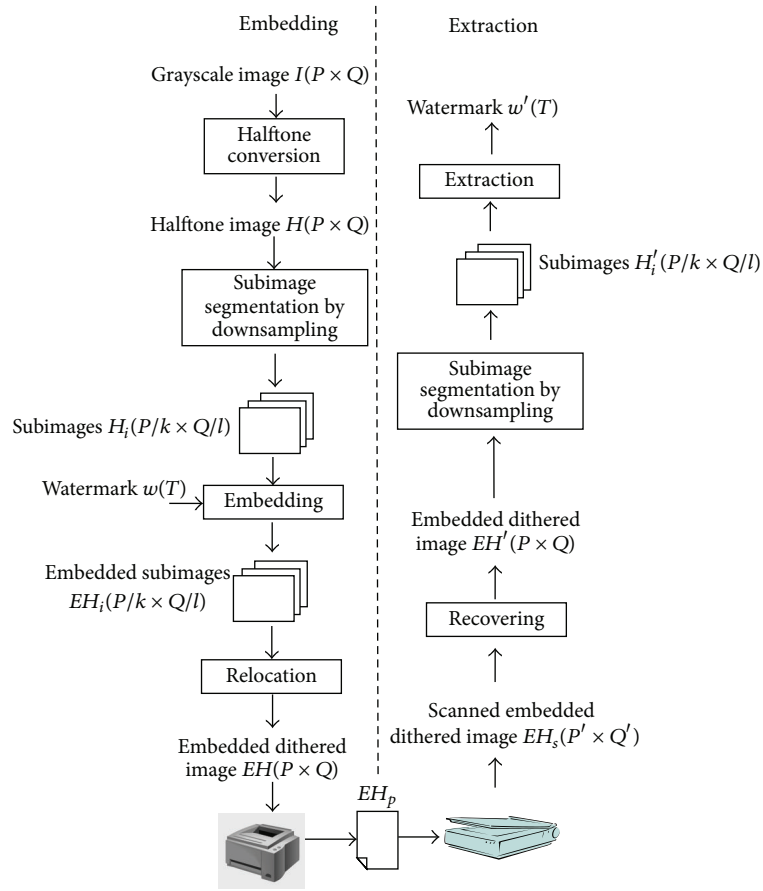


Figure 3: The block diagram of the proposed method.

Since there are $(n \times n)/2$ pairs of subimages in RH and each pair of subimages embeds a bit, the embedding capacity is $(n \times n)/2$. To increase the embedding capacity, Guo et al. [12] first divides the original halftone image H into $k \times l$ subimages H_is. Then, the above-mentioned bit-interleaving method is applied to each H_i to get $n \times n$ sub-subimages $SH_{i,j}$'s to embed $(n \times n)/2$ bits. Hence, the capacity is increased to $(nk \times nl)/2$. Figure 2(d) shows the result of dividing Figure 2(c) into 16 subimages. Figure 2(e) shows the result of applying bit-interleaving to each subimage in Figure 2(d).

Table 1 shows the numbers of black pixels in the sub-subimage pairs $(SH_{0,j}, SH_{0,j+1})$ of the subimage H_0 in the bottom-left of Figure 2(e), where $j = 2m$ and $m = 0, 1, \ldots, 7$. In Table 1, the difference of the numbers of black pixels between $SH_{0,j}$ and $SH_{0,j+1}$ is 0, for $j = 4, 8, 12$. These sub-subimage pairs are NIPs. To eliminate these NIPs, at least one white pixel is chosen and changed into a black pixel in $SH_{0,5}$, $SH_{0,9}$, and $SH_{0,13}$, respectively. This may make the visual quality degraded. Furthermore, from Table 1, we find that the average difference of numbers of black pixels in the eight sub-subimage pairs is 5.25. The low difference of black pixel numbers between $SH_{0,j}$ and $SH_{0,j+1}$ could lead to the sign of the difference being altered, when the embedded dithered image is processed by print-and-scan process. This will make the extracted watermark bit wrong. Here, we will propose a method to treat these disadvantages.

Table 1: The black pixel numbers of eight sub-subimage pairs in the bottom-left subimage of Figure 2(d).

j	# Black pixels		The difference of black pixel numbers between $SH_{0,j}$ and $SH_{0,j+1}$
	$SH_{0,j}$	$SH_{0,j+1}$	
0	0	2	2
2	23	33	10
4	0	0	0
6	18	37	19
8	0	0	0
10	3	6	3
12	0	0	0
14	1	9	8

3. The Proposed Method

The proposed method contains two parts: embedding and extraction. Figure 3 shows the block diagram of the proposed method. In the embedding part, first, a grayscale image I is converted into a halftone image H. Secondly, H is segmented into $k \times l$ subimages H_i through downsampling. Thirdly, watermark bits are embedded into each subimage. Fourthly, all pixels in the embedded subimages EH_i are relocated to their original positions to form the embedded dithered image

EH. Finally, EH can be printed on a paper. In the extraction, after transmission, the printed embedded dithered image EH_p is scanned by a scanner, and then a scanned embedded dithered image EH_s is produced. Since the print-and-scan process could cause distortion, a recovering algorithm proposed by Guo et al. [12] is used to correct the distortion in EH_s. After that, the output EH' will be segmented into several subimages through downsampling. Finally, the watermark w' can be extracted from each subimage.

In this section, we will first introduce the proposed embedding algorithm. Then, the provided extracting algorithm is described.

3.1. Embedding Algorithm. The embedding algorithm contains four stages: halftone conversion, subimage segmentation, embedding, and relocation. They are described in the following.

3.1.1. Halftone Conversion. An $n \times n$ dispersed dithering screen (DS) [1] is first applied to a $P \times Q$ grayscale image I to result in a dithering halftone image H according to (1).

3.1.2. Subimage Segmentation. Suppose that a watermark w with T bits will be embedded. In the subimage segmentation stage, H is segmented into $k \times l$ subimages through downsampling, where $k \times l \geq \lceil T/(n \times n)/2 \rceil$ and each subimage has $P/k \times Q/l$ pixels. First, H is divided into $P/kn \times Q/ln$ blocks, each of which has $kn \times ln$ pixels. Then, each block is further divided into $k \times l$ subblocks, each of which has $n \times n$ pixels. Let $A_{i,j}$ be the jth subblock in block i, where $i = 0, 1, \ldots, ((P \times Q)/(kn \times ln) - 1)$ and $j = 0, 1, \ldots, (kl - 1)$, respectively. Finally, through downsampling, all jth subblocks ($A_{i,j}$, $i = 0, 1, \ldots, ((P \times Q)/(kn \times ln) - 1)$) are grouped into a subimage H_j. For the convenience of explanation, all H_js are arranged into an image.

Figure 4(a) shows an image divided into 3×3 blocks with $P = Q = 48$, $k = l = 4$, and $n = 4$, each of which is further divided into 4×4 subblocks. Figure 4(b) shows the subimage segmentation result of Figure 4(a). Figure 4(c) shows the segmentation result of Figure 2(b) with $T = 120$, $k = 4$, $l = 4$, and $k \times l = 16 > 120/((4 \times 4)/2) = 15$. The rectangles in Figure 4(c) denote the 16 subimages.

3.1.3. Embedding. In the embedding stage, an $n \times n$ dispersed dithering screen with $n = 2^k$ and $k \in N$ is used. The $n \times n$ dispersed dithering screen is first divided into 2×2 dispersed dithering screens. Secondly, in each 2×2 dispersed dithering screen, two elements with thresholds more than 128 are grouped as a pair, and the other two elements with thresholds less than 128 are grouped as a pair. Hence, we can obtain $(n \times n)/2$ pairs.

Thirdly, sort all values in the $n \times n$ dispersed dithering screen and give each value an order; then each pair is represented by its corresponding order. For example, in Figure 1(b) with $n = 8$, the four elements in the top-left 2×2 dispersed dithering screen marked by a red rectangle have thresholds 6, 238, 134, and 70. The two elements with thresholds 6 and 70 are grouped as a pair, and 134 and 238 are grouped as another pair. After sorting the values in the dispersed dithering screen in Figure 1(b), the corresponding order of each element is shown in Figure 1(d). The two pairs are represented by their corresponding orders as (1, 17) and (33, 59).

Table 2: The black pixel numbers in eight pairs of the top-left subimage in Figure 4(c).

Pairs	The number of black pixels	
	p_u	p_v
(p_0, p_4)	0	18
(p_1, p_5)	0	32
(p_2, p_6)	10	35
(p_3, p_7)	16	44
(p_8, p_{15})	53	64
(p_9, p_{14})	54	64
(p_{10}, p_{12})	58	64
(p_{11}, p_{13})	60	64

Fourthly, let ODS_j be the position with order j in the dispersed dithering screen DS, where $j = 0, \ldots, (n \times n) - 1$. Figure 1(c) shows the order and position of each pixel in the 4×4 dispersed dithering screen shown in Figure 1(a); in this figure, $\text{ODS}_0 = (2, 2)$, $\text{ODS}_{15} = (2, 3)$. Fifthly, each subimage H_i is divided into blocks with size $n \times n$, and all pixels at position ODS_j of all blocks are grouped into a sub-subimage $SH_{i,j}$.

Sixthly, the number, p_j, of black pixels at $SH_{i,j}$ is counted. Then, we take (p_u, p_v) as a pair, when $n = 4$, $(u, v) = (0, 4), (1, 5), (2, 6), (3, 7), (8, 15), (9, 14), (10, 12)$, or $(11, 13)$. Note that, for each of above-mentioned pair (p_u, p_v) with $n = 4$, the difference, $d_{(u,v)}$, of values of ODS_u and ODS_v in the dispersed dithering screen DS is not less than 32. However, for each pair (p_i, p_{i+1}) used by Guo et al., the difference, $d_{(i,i+1)}$, of values of ODS_i and ODS_{i+1} in the dispersed dithering screen DS is equal to 16. This will make each pair (p_u, p_v) used in the proposed method have larger $|p_u - p_v|$ than $|p_i - p_{i+1}|$ used in Guo et al.'s method. One example is given in Table 2. Table 2 shows the black pixel numbers of each pair (p_u, p_v) in the top-left subimage of Figure 4(c); the average difference of numbers of black pixels in the eight pairs is 16.75, which is greater than 5.25 in Guo et al.'s method (see Table 1). This means that the proposed method will provide higher robustness than Guo et al.'s method for the print-and-scan process.

Finally, one bit is embedded into each pair (p_u, p_v). If the embedding bit is 0 and $p_u < p_v$ or the embedding bit is 1 and $p_u > p_v$, nothing is done. Otherwise, in each block, the pixel at position ODS_u and the pixel at position ODS_v are swapped. Note that if $p_u = p_v$, no bit will be embedded into the pair. This kind of pairs is called equivalent black pixel pair (EBP).

3.1.4. Relocation. After embedding, all pixels are relocated to their original positions to form the embedded dithered image EH. Then EH can be printed on a paper to form a printed embedded dithered image EH_p. Note that the embedding capacity is $k \times l \times (n \times n)/2$ bits.

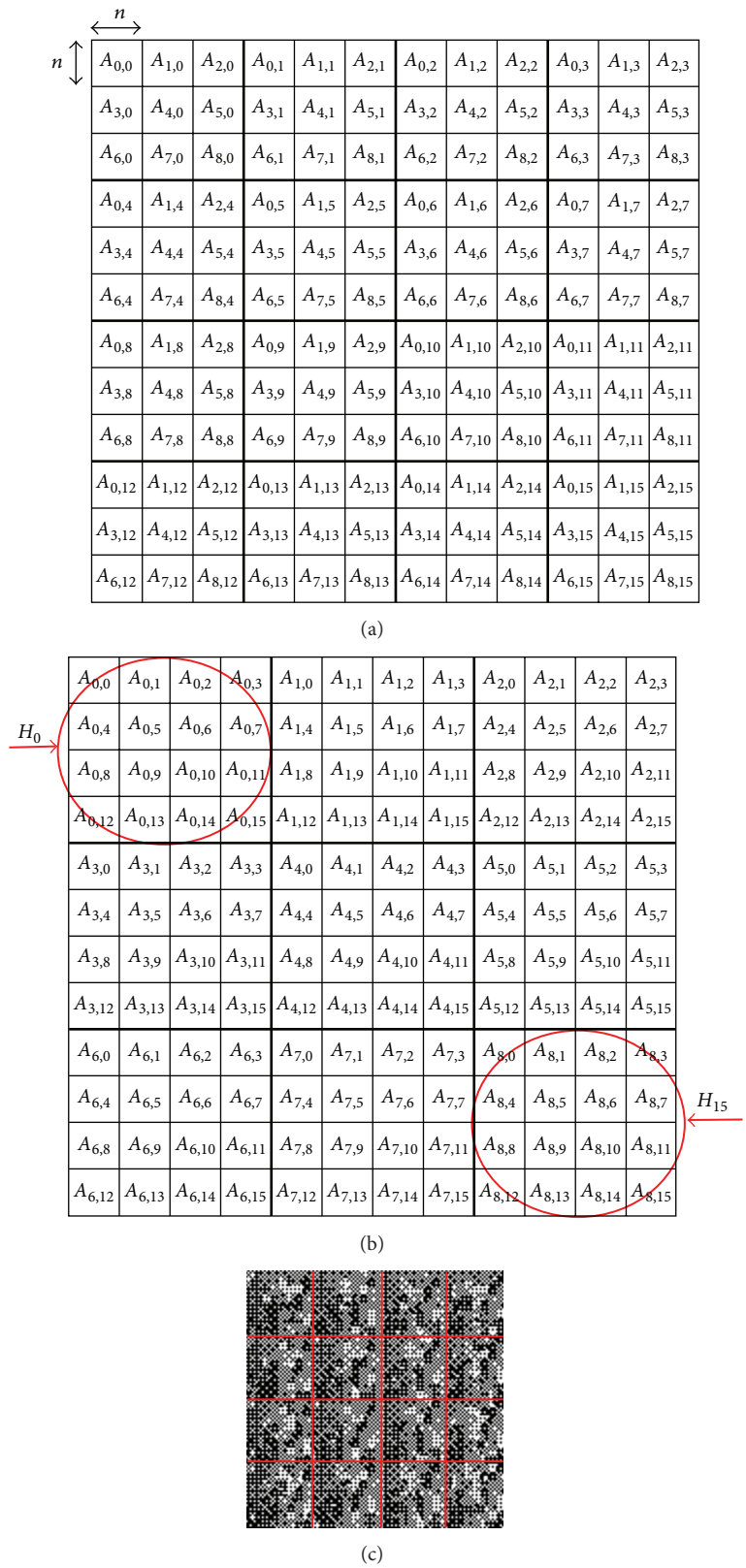
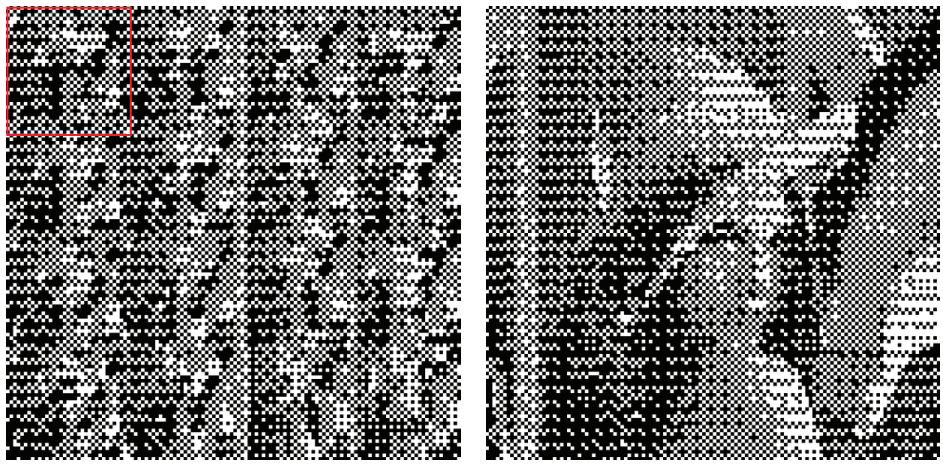


FIGURE 4: An example of subimage segmentation. (a) An image divided into 3 × 3 blocks, each of which has 4 × 4 subblocks. (b) The segmentation result of (a). (c) The segmentation result of Figure 2(b).



(a) (b)

Figure 5: An example of embedding algorithm. (a) The result of embedding 120 bits into Figure 4(c). (b) The embedded dithered image by relocating (a).

Figure 5(a) shows the result of embedding 120 watermark bits into Figure 4(c). The rectangle marked in Figure 5(a) denotes the first subimage with 8 watermark bits 10101010 embedded. Figure 5(b) shows the embedded dithered image resulting from relocating Figure 5(a).

3.2. Recovering Algorithm for Print-and-Scan Process. When we receive the paper with the printed embedded dithered image EH_p, a scanner is used to capture EH_p and produce a scanned image EH_s. The scanned image EH_s usually has geometrical distortion and dot gain effect due to the scanner and printer properties. Dot gain is a phenomenon in printing which causes the size of a dot to be increased or decreased slightly. Here, we adopt the recovering algorithm proposed by Guo et al. [12] to get the embedded dithered image EH'.

3.3. Watermark Extraction Algorithm. The watermark extraction contains two steps: subimage segmentation and extraction. To extract the embedded watermark, EH' will be segmented into several subimages through downsampling mentioned in the embedding algorithm. For each subimage H_i', we divide it into blocks with size $n \times n$ and then count the number, p_j, of black pixels at position ODS_j of all blocks. Then, (p_u, p_v) is taken as a pair, when $n = 4$, $(u, v) = (0, 4), (1, 5), (2, 6), (3, 7), (8, 15), (9, 14), (10, 12)$, or $(11, 13)$. Finally, for each pair (p_u, p_v), a watermark bit is extracted and considered to be 0 if $p_u < p_v$ and 1 if $p_u > p_v$. Otherwise, no watermark bit is embedded if $p_u = p_v$.

4. Experimental Results and Comparisons

Eight 512×512 test images used in [12, 15] are shown in Figure 6 and also used in our experiments. Figure 7 shows the visual quality comparison of different methods when $n = 4$. The regions marked by cycles in Figures 7(e)–7(h) are high-frequency ones of Figures 7(a)–7(d), respectively. Compared to Figure 7(e), we can see that Figure 7(h) is similar to Figure 7(e), but the boundary area in Figure 7(f) is smeared and unclear. This means that the proposed method provides higher visual quality in high-frequency areas than Guo et al.'s method [12]. Besides, we can see that Figure 7(c) has salt-and-pepper noises. Hence, the proposed method provides better visual quality than Hagit's method [13].

Figure 8 shows the visual quality comparison of different methods when $n = 8$. Compared to Figure 8(e), we can see that Figure 8(h) is similar to Figure 8(e), but the boundary area in Figure 8(f) is smeared and unclear. Compared to Figures 7(f) and 8(f), we can see that Figure 8(f) is more smeared and unclear than Figure 7(f). Besides, we can see that Figure 8(c) has more salt-and-pepper noises. Hence, the proposed method provides better visual quality than Hagit's method [13].

Next, two objective methods [7, 14] are used to measure the halftone image quality. One is the Pei-Guo-PSNR proposed by Pei and Guo [7]; it is adopted to measure the quality of a halftone image and is evaluated as follows:

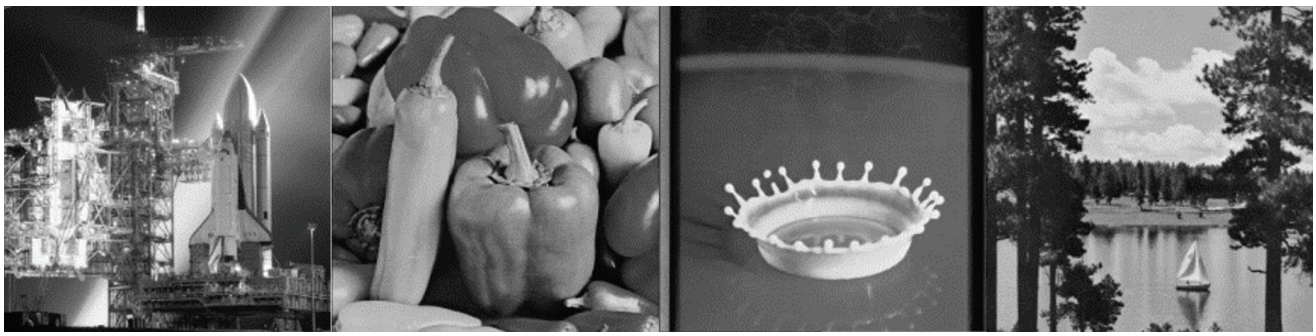
$$\text{Pei-Guo-PSNR} = \frac{P \times Q}{\sum_{i=1}^{P} \sum_{j=1}^{Q} \left[x_{i,j} - \sum_{(m,n) \in R} \sum w_{m,n} b_{i+m,j+n} \right]^2}, \quad (2)$$

where $w_{m,n}$ is an $r \times r$ least-mean square filter and can be obtained by a training process [7], $R = \{(m, n) \mid -(r-1)/2 \leq m, n \leq (r-1)/2\}$, x is the original grayscale image, and b is the corresponding halftone image. Here, a 7×7 least-mean square (LMS) filter [7] (see Figure 9) is adopted to measure the quality of halftone images. Table 3 shows the quality comparisons of various methods using Pei-Guo-PSNR; a random bit stream is adopted as a watermark. From this table, we can see that the proposed methods and Guo et al.'s method provide similar qualities when $n = 4$. But the proposed method provides better qualities than Hagit's method. When $n = 8$, the proposed method provides better qualities than other methods. Because Pei-Guo-PSNR is basically the PSNR of

Table 3: Quality comparisons of various algorithms using Pei-Guo-PSNR [12].

Algorithm		Original		Hagit's method [13]		Guo et al.'s method [12]		Proposed method	
		Average Pei-Guo-PSNR							
Screen size		$n = 4$	$n = 8$	$n = 4$	$n = 8$	$n = 4$	$n = 8$	$n = 4$	$n = 8$
Embedded bits	0	31.56	31.19	N/A	N/A	N/A	N/A	N/A	N/A
	32	N/A	N/A	26.83	27.55	29.21	26.29	28.65	29.14
	128	N/A	N/A	26.80	27.52	29.17	26.55	28.58	28.82
	512	N/A	N/A	26.87	27.38	28.91	26.31	28.49	28.92

(a)



(b)

Figure 6: Thumbnail of eight test images. (a) Lena, Mandrill, Earth, and Tiffany. (b) Shuttle, Peppers, Milk, and Lake.

the original grayscale image and a low-pass version of the halftone image, it measures effectively the distortions to the low-frequency image content [14]. But Pei-Guo-PSNR is improper for measuring the high-frequency image content.

Fu and Au [14] proposed another measure to treat the above-mentioned disadvantage. Let x be the original grayscale image; let EH be the embedded halftone image. Fu and Au [14] define two special classes of elements in EH as follows:

Class 1. Black pixel in bright region ($EH(m,n) = 0,\ x(m,n) \geq 128$).

Class 2. White pixel in dark region ($EH(m,n) = 255,\ x(m,n) \leq 127$).

Based on these two classes, Fu and Au [14] define four scores as follows:

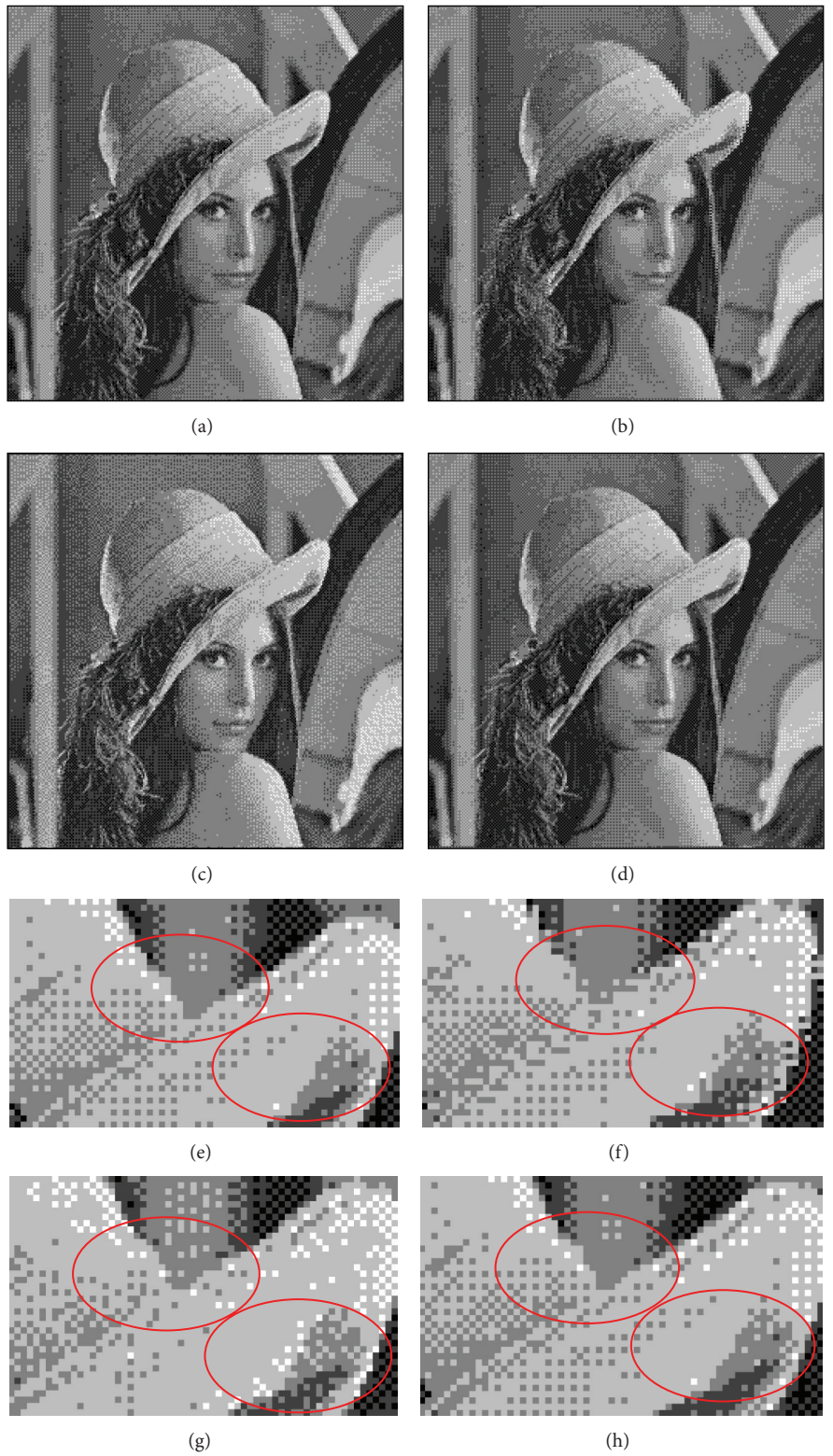
$$S_1 = \sum_{i=0}^{4} N_i,$$

$$S_2 = \sum_{i=0}^{4} (i+1)\, N_i,$$

$$S_3 = \frac{S_2}{S_1},$$

$$S_4 = \sum_{i=2}^{4} N_i, \tag{3}$$

where N_i is the total number of Class 1 and Class 2 elements in EH having i neighbors with the same pixel values in the 4-neighborhood. N_0 corresponds to the number of isolated Class 1 or Class 2 elements. S_3 and S_4 can be used to measure the visual disturbing of "salt-and-pepper" clusters formed by neighboring pixels [14]. Thus, we adopt scores S_3 and S_4 to measure the quality of a halftone image. Algorithms with smaller scores of S_3 and S_4 are better. Table 4 shows the quality comparisons of various methods based on scores S_3 and S_4.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 7: Visual quality comparison with 128 embedding bits when $n = 4$. (a) Original dithered image. (b) Embedded dithered image using Guo et al.'s method [12]. (c) Embedded dithered image using Hagit's method [13]. (d) Embedded dithered image using the proposed method. (e) Enlarged partial image of (a). (f) Enlarged partial image of (b). (g) Enlarged partial image of (c). (h) Enlarged partial image of (d).

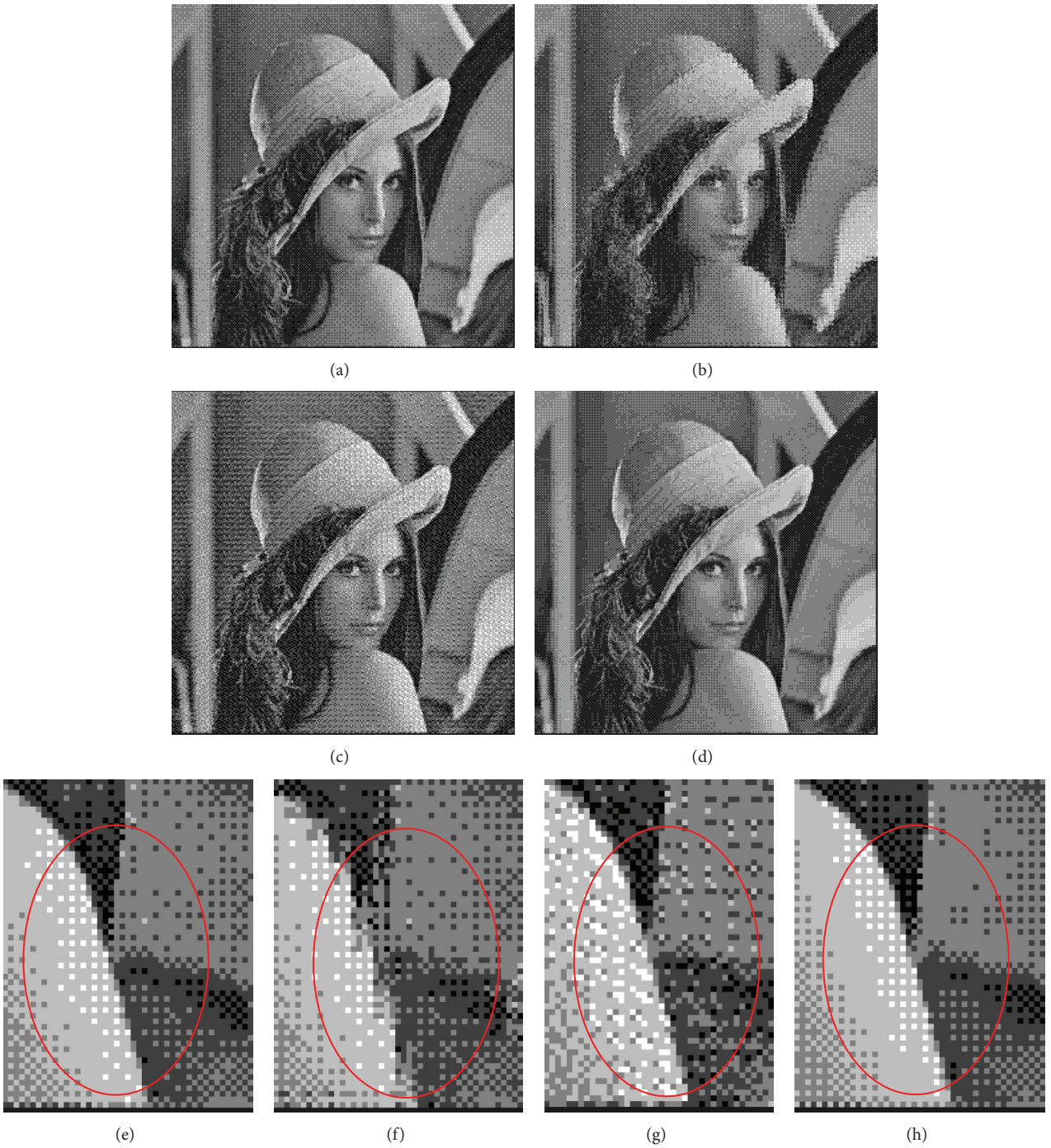


FIGURE 8: Visual quality comparison with 128 embedding bits when $n = 8$. (a) Original dithered image. (b) Embedded dithered image using Guo et al.'s method [12]. (c) Embedded dithered image using Hagit's method [13]. (d) Embedded dithered image using the proposed method. (e) Enlarged partial image of (a). (f) Enlarged partial image of (b). (g) Enlarged partial image of (c). (h) Enlarged partial image of (d).

From this table, we can see that the proposed method has smaller scores of S_3 and S_4; thus, it is better than other methods.

From the above experiments, we can see that the image quality of Guo et al.'s method becomes worse when the dispersed dithering screen size increases. The reason is that if the dispersed dithering screen size increases, the NIP problem in Guo et al.'s method becomes more serious. Many black pixels will be added to eliminate these NIPs in Guo et al.'s method. Hence, the number of black pixels in some subimages will be changed. In addition, the pixel swapping distance of Guo et al.'s method also increases if the dispersed dithering screen size increases. Hence, the boundary of Guo et al.'s method will become more smeared and unclear and its image

Table 4: Quality comparisons of various algorithms based on scores S_3 and S_4 [14].

Algorithm		Hagit's method [13]		Guo et al.'s method [12]		Proposed method	
		Average S_3					
Screen size		$n = 4$	$n = 8$	$n = 4$	$n = 8$	$n = 4$	$n = 8$
Embedded bits	32	2.49	2.23	1.61	1.69	1.61	1.18
	128	2.64	2.32	1.67	1.52	1.67	1.18
	512	2.58	2.32	1.68	1.61	1.68	1.18
		Average S_4					
Screen size		$n = 4$	$n = 8$	$n = 4$	$n = 8$	$n = 4$	$n = 8$
Embedded bits	32	17459	11164	1372	2583	829	899
	128	17528	11738	1503	1833	824	815
	512	17328	11382	1660	2240	799	854

Table 5: Quality comparisons of various algorithms with 64×64 watermark bits embedded in Figure 8(a).

Algorithm	Guo et al.'s method [12]	Proposed method	Feng et al.'s method [17]
Pei-Guo-PSNR	28.68	30.17	15.15
S_3	1.43	1.07	3.86
S_4	1847	201	54592

−0.001	0.006	0.013	0.015	0.013	0.006	−0.001
0.003	0.007	0.022	0.0.29	0.022	0.009	0.002
0.010	0.025	0.051	0.067	0.051	0.024	0.009
0.007	0.030	0.064	0.091	0.064	0.031	0.007
0.007	0.020	0.042	0.054	0.041	0.019	0.006
0.003	0.011	0.024	0.030	0.023	0.009	0.003
−0.004	0.002	0.008	0.006	0.007	0.002	−0.004

Figure 9: Coefficients of 7×7 LMS filter.

quality is degraded. However, the proposed method does not have this problem. Hence, the boundary is still clear when the dispersed dithering screen size increases.

Since Feng et al.'s method [17] can only embed $P/8 \times Q/8$ watermark bits, we compare the qualities of the embedded halftone images using Feng et al.'s method [17] and others only for $P/8 \times Q/8$ watermark bits embedded. Table 5 shows the comparison results for the 512×512 halftone image shown in Figure 8(a). From this table, we can see that the proposed method provides better qualities than other methods.

To justify the selection of (u, v) pair, we use three different kinds of selection for (u, v) pairs. The first selection is used by Guo et al.'s method with $v - u = 1$; all (u, v) pairs are $(0, 1)$, $(2, 3)$, $(4, 5)$, $(6, 7)$, $(8, 9)$, $(10, 11)$, $(12, 13)$, and $(14, 15)$. The second selection follows the rule $2 \leq v - u \leq 5$; all (u, v) pairs are $(0, 4)$, $(1, 5)$, $(2, 6)$, $(3, 7)$, $(8, 15)$, $(9, 14)$, $(10, 12)$, and $(11, 13)$. The third selection follows the rule $6 \leq v - u \leq 11$; all (u, v) pairs are $(0, 8)$, $(1, 9)$, $(2, 10)$, $(3, 11)$, $(4, 15)$, $(5, 14)$, $(6, 12)$, and $(7, 13)$. From the dispersed dithering screens shown in Figure 1, we can see that when $(v-u)$ is larger, $T_v - T_u$ is larger; this will make $|p_u - p_v|$ larger. When $|p_u - p_v|$ is larger, then it can provide a higher robustness for the print-and-scan process. But the quality will be degraded. Thus, the second selection will provide higher robustness and better quality. To prove this point, we have done some experiments based on these three kinds of selection by embedding 128 watermark bits into a 512×512 image.

Table 6 shows quality comparisons of three different kinds of pair selections using Pei-Guo-PSNR, scores S_3 and S_4. From this table, we can see that these pair selections provide similar qualities in Pei-Guo-PSNR. But, in S_3 and S_4, the second selection used in the proposed method has the best result, and the third selection has the worst result. Table 7 shows the black pixel numbers in eight pairs of the top-left subimage for various pair selections. The average difference on numbers of black pixels in the eight pairs for the second selection used in the proposed method is 223.5, the third selection is 238.5, and the first selection used in Guo et al.'s method is 62.75. This means that Guo et al.'s method provides less robustness than the other pair selections, and the third selection will provide similar robustness to the second pair selection for the print-and-scan process. However, the image quality using the third pair selection is worse than those of two pair selections. Therefore, under the consideration of image quality and robustness, the second selection used in the proposed method is better.

Furthermore, in the experiments of Guo et al. [12], based on a 4×4 dispersed dithering screen, the average percentages of NIPs with 8, 32, 128, and 512 bits embedded into each of

Table 6: Quality comparisons of three different kinds of pair selections using Pei-Guo-PSNR [12] and scores S_3 and S_4 [14].

Selection	The 1st selection used in Guo et al. [12] with $v - u = 1$			The 2nd selection used in the proposed method with $2 \leq v - u \leq 5$			The 3rd selection with $6 \leq v - u \leq 11$		
Images	Quality								
	Pei-Guo-PSNR	S_3	S_4	Pei-Guo-PSNR	S_3	S_4	Pei-Guo-PSNR	S_3	S_4
Lena	30.78	1.37	480	29.24	1.06	204	28.26	2.26	12967
Mandrill	27.30	2.24	3201	27.32	1.59	2758	26.54	2.36	16866
Earth	27.06	2.07	2772	28.17	1.28	1233	28.45	2.09	6841
Tiffany	31.06	1.12	189	29.06	1.03	159	28.82	1.70	1981
Shuttle	27.66	1.90	2134	28.08	1.20	871	28.12	2.18	10062
Peppers	30.34	1.46	755	29.00	1.08	381	28.25	2.18	10525
Milk	31.04	1.32	529	29.63	1.02	129	28.65	2.17	10080
Lake	28.15	1.88	1967	28.15	1.13	857	28.34	1.95	5753
Average	29.17	1.67	1503.4	28.58	1.17	824	28.18	2.11	9384.4

Table 7: The black pixel numbers of the eight pairs in the top-left subimage for different kinds of selections.

Selection								
The 1st selection used in Guo et al. [12] with $v - u = 1$			The 2nd selection used in the proposed method with $2 \le v - u \le 5$			The 3rd selection with $6 \le v - u \le 11$		
Pairs (u, v)	The number of black pixels		Pairs (u, v)	The number of black pixels		Pairs (u, v)	The number of black pixels	
	p_u	p_v		p_u	p_v		p_u	p_v
(0, 1)	322	352	(0, 4)	389	641	(0, 8)	389	734
(2, 3)	320	354	(1, 5)	391	725	(1, 9)	391	826
(4, 5)	386	540	(2, 6)	364	757	(2, 10)	364	884
(6, 7)	545	547	(3, 7)	421	712	(3, 11)	421	873
(8, 9)	611	758	(8, 15)	734	672	(4, 15)	641	672
(10, 11)	801	668	(9, 14)	826	675	(5, 14)	725	675
(12, 13)	672	674	(10, 12)	884	711	(6, 12)	757	711
(14, 15)	672	672	(11, 13)	873	741	(7, 13)	712	741

Table 8: Numbers of equivalent black pixel pairs in eight test images using the proposed method.

Number of subimages		1	4	16	64	128	256
Total number of pairs		8	32	128	512	1024	2048
Number of EBPs	Lena	0	0	0	0	0	0
	Mandrill	0	0	0	0	0	0
	Earth	0	0	0	0	0	0
	Tiffany	0	2	22	146	373	830
	Shuttle	0	0	0	0	0	0
	Peppers	0	0	0	0	0	0
	Milk	0	0	0	0	0	0
	Lake	0	0	0	0	0	0

eight test images shown in Figure 6 are 12.5%, 15.62%, 27.32%, and 45%, respectively. Guo et al. modify the number of black pixels in these NIPs before embedding watermark. This may lower visual quality. On the contrary, in the proposed method, we do not modify any pair before data embedding.

As to the embedding capacity, since the data embedding depends on the difference of black pixel numbers of each pair, if the difference is zero, the pair cannot be used for data embedding in the proposed method. This will reduce the embedding capacity. Fortunately, from our experimental results, we found that the situation rarely appears in most images. Table 8 shows the numbers of equivalent black pixel pairs (EBP) in eight dithered test images. From this table, we can see that most images have zero EBP, except "Tiffany" image. The reason is that most pixels in "Tiffany" image have gray values >90, and the gray values of pixels in a local area are nearly constant. For example, for pair (p_1, p_5), the corresponding values in DS are (24, 88) (see Figure 1); thus, each pixel in $SH_{i,1}$ and $SH_{i,5}$ for each i will be a white point. This will make $(p_1, p_5) = (0, 0)$ and make it an EBP.

In the next experiment, we demonstrate the robustness of the proposed method and Guo et al.'s method by cropping and tampering attacks. To measure the integrity of the extracted watermark, the correct decoding rate (CDR) is defined as follows:

$$\mathrm{CDR} = \frac{\mathrm{Lev\_Dist}\left(w, w'\right)}{T} \times 100\%, \tag{4}$$

where Lev_Dist(), w, and w' denote the Levenshtein distance [22], original watermark, and extracted watermark, respectively. The Levenshtein distance is a string metric for estimating the least number of edit operations that is necessary to modify one string to obtain another string. In Figures 10 and 11, a 32×32 watermark shown in Figure 10(a) is embedded into the halftone image in Figure 11(b) using the proposed method and Guo et al.'s method, respectively, and the results are shown in Figures 10(b) and 11(a), respectively. Figures 10(c) and 11(b) show the embedded dithered images cropped by 1/4 portion. Figures 10(d) and 11(c) show the embedded dithered images tampered with several words. Note that when the number of embedding bits >8, the dithered image is first divided into t subimages, and the watermark bits are also divided into t parts. Each part is embedded into one subimage. The resulting subimages using the proposed method and Guo et al.'s method are shown in Figures 10(e), 10(f), 11(d), and 11(e). From Figure 10(e), we can see that each subimage using the proposed method is also cropped by 1/4 portion; the reason is that each subimage is obtained by block downsampling (see Figure 4). The cropping will make 1/4 portion of all pixel pairs (u, v) lost. Since a watermark bit is embedded through the sign of $(p_u - p_v)$, 1/4 portion of pairs (u, v) lost will not affect the sign of $(p_u - p_v)$. Thus, the watermark can be extracted correctly (see Figure 10(g)). Figure 10(h) shows the watermark extracted from Figure 10(d) and it can also be extracted correctly. On the contrary, in Figure 11(d), we can see that some subimages using Guo et al.'s method are totally removed; this will make the watermark parts embedded in these subimages lost (see Figure 11(f)). Figures 11(f) and 11(g) show the watermarks extracted from Figures 11(b) and 11(c), respectively. Table 9 shows the average correct decoding rates of cropping 1/3, 1/4, and 1/2 portions, respectively. Note that in this experiment, each of the eight images is cropped at three different locations for

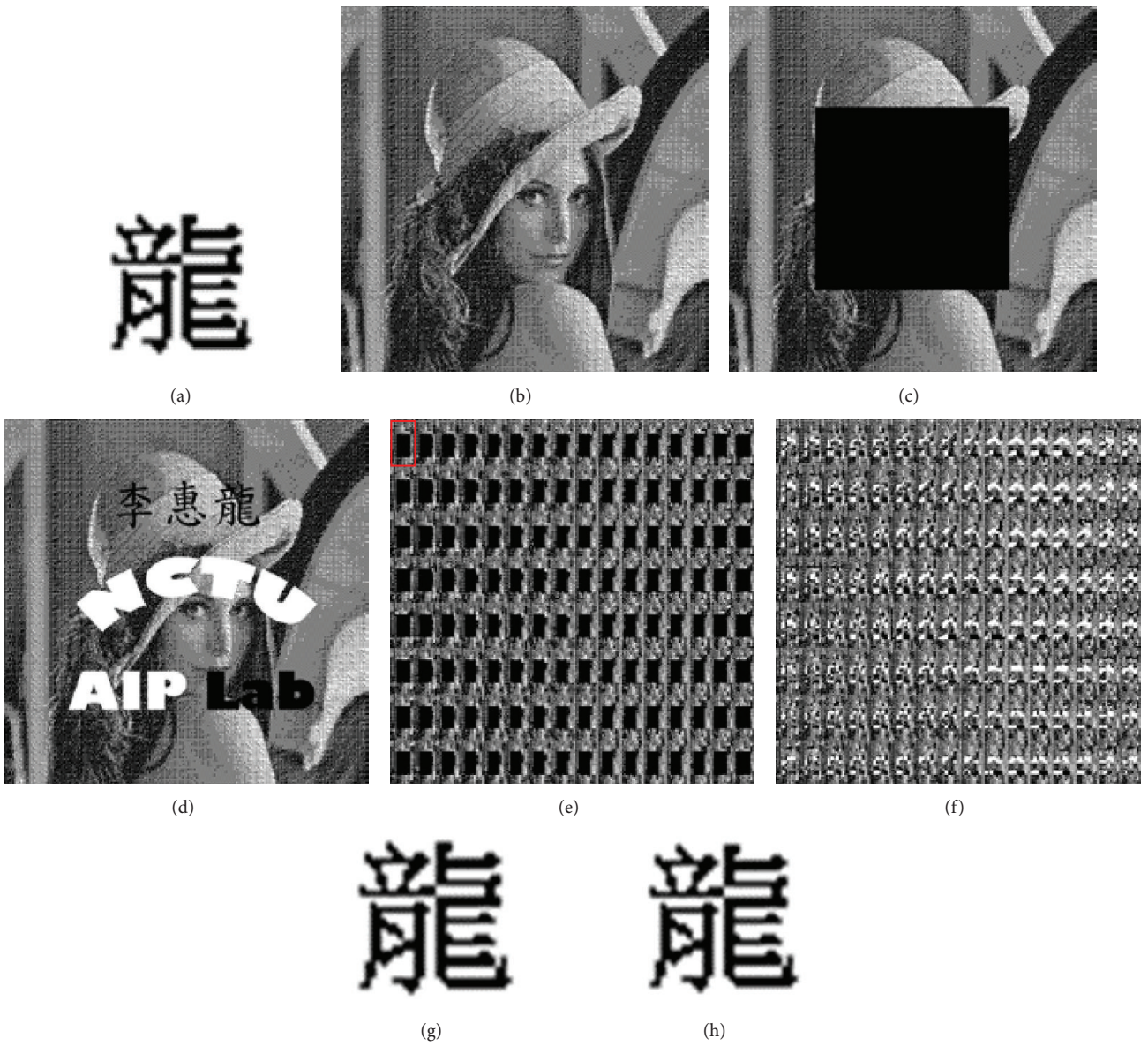


Figure 10: The robustness of the proposed method. (a) A 32×32 watermark. (b) Embedded dithered image using the proposed method. (c) Embedded dithered image cropped by 1/4 portion. (d) Embedded dithered image with tampering. (e) Downsampled subimages of (c). (f) Downsampled subimages of (d). (g) The watermark extracted from (c). (h) The watermark extracted from (d).

each cropping portion; thus there are 72 ($3 \times 3 \times 8$) cropped images. From this table, we can see that the proposed method is more robust than Guo et al.'s method and Feng et al.'s method [17].

To measure the robustness of the proposed method and Guo et al.'s method [12] under print-and-scan attack, for each of eight test images, we first embed 8, 32, 128, and 512 bits into its corresponding halftone image; then each embedded halftone image is printed at 150 dpi. After printing, each printed image is scanned at 150, 450, and 750 dpi, respectively, and then the embedded watermark from each scanned image is extracted. Table 10 shows the average correct decoding rate with eight test images as shown in Figure 6. From this table, we can see that the average correct decoding rates of the proposed method are higher than those of Guo et al.'s method. This means that the proposed method provides more robustness than Guo et al.'s method under print-and-scan attack.

5. Conclusions

In this paper, a robust watermarking method has been proposed for dithered halftone images. Before embedding, a dithered halftone image is divided into subimages through downsampling; this step provides robustness to cropping and tampering. In the embedding step, each pair (p_u, p_v) is taken to embed a watermark bit with $(u, v) = (0, 4), (1, 5), (2, 6), (3, 7), (8, 15), (9, 14), (10, 12)$, or $(11, 13)$; however, in

Table 9: Average correct decoding rates of different cropping size.

Cropping size	Average correct decoding rates				
	Guo et al.'s method [12]		Proposed method		Feng et al.'s method [17]
	Watermark size				
	32 × 32	64 × 64	32 × 32	64 × 64	64 × 64
1/2	50.00%	50.00%	93.98%	91.87%	67.20%
1/3	66.50%	66.55%	94.20%	94.77%	78.25%
1/4	75.00%	75.00%	95.24%	96.05%	83.64%

Table 10: Average correct decoding rates of all scanned embedded images with different scanning resolutions.

Algorithm		Guo et al.'s method [12]			Proposed method		
		Average correct decoding rate					
Scanning resolution		150	450	750	150	450	750
Embedded bits	32	65%	72%	70%	77%	85%	84%
	128	60%	70%	75%	78%	82%	86%
	512	57%	65%	72%	78%	83%	86%

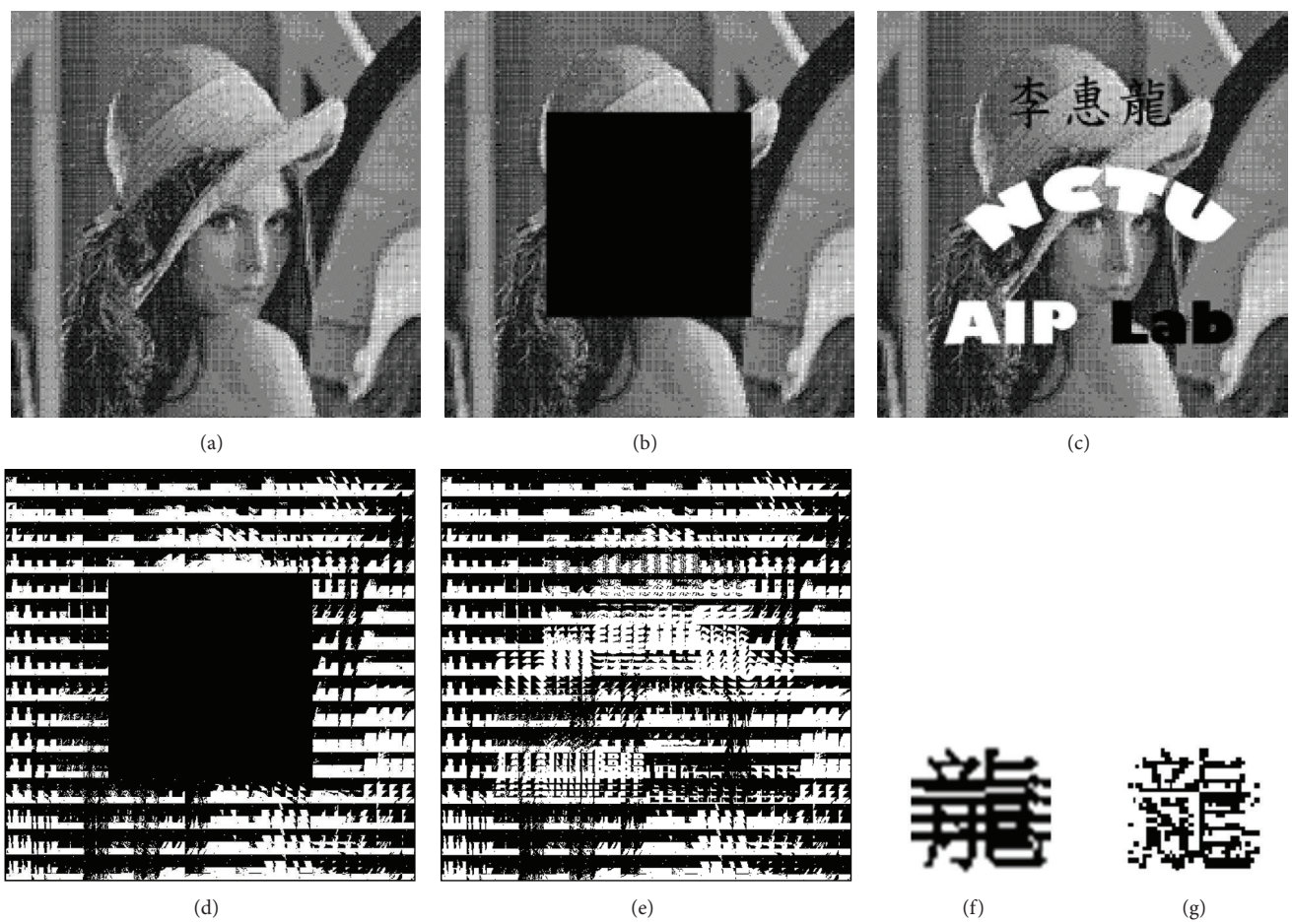


Figure 11: The robustness of Guo et al.'s method [12]. (a) Embedded dithered image using Guo et al.'s method. (b) Embedded dithered image cropped by 1/4 portion. (c) Embedded dithered image with tampering. (d) The result of applying bit-interleaving to each subimage of (b). (e) The result of applying bit-interleaving to each subimage of (c). (f) The watermark extracted from (b). (g) The watermark extracted from (c).

Guo et al.'s method, the pair (p_i, p_{i+1}) is taken to embed a watermark bit. Since the average of $|p_u - p_v|$ is larger than the average of $|p_i - p_{i+1}|$, this makes the proposed method provide higher correct decoding rate than Guo et al.'s method after print-and-scan process. Experimental results show that the proposed method actually provides higher robustness than Guo et al.'s method in cropping, tampering, and print-and-scan process. In addition, experimental results also show that the proposed method provides higher visual quality in the high-frequency areas than Guo et al.'s method.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by the Ministry of Science and Technology of Taiwan under Contract MOST 103-2221-E-009-121-MY2.

References

[1] R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial grey scale," in *Proceedings of the SID International Symposium*, Digest of Technical Papers, pp. 36–37, 1975.

[2] B. E. Bayers, "An optimum method for two level renditions of continuous tone pictures," in *Proceedings of the IEEE International Conference Communication*, pp. 2611–2615, June 1973.

[3] J. P. Allebach, R. Eschbach, and G. G. Marcu, "DBS: retrospective and future directions," in *Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts VI*, Proceedings of SPIE, pp. 358–376, San Jose, Calif, USA, January 2001.

[4] M. S. Fu and O. C. Au, "Data hiding watermarking for halftone images," *IEEE Transactions on Image Processing*, vol. 11, no. 4, pp. 477–484, 2002.

[5] M. S. Fu and O. C. Au, "A robust public watermark for halftone images," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. III/639–III/642, May 2002.

[6] M. S. Fu and O. C. Au, "Correlation-based watermarking for halftone images," in *Proceedings of the International Symposium on Circuits and Systems (ISCAS '04)*, vol. 2, pp. II-21–II-24, Vancouver, Canada, May 2004.

[7] S.-C. Pei and J.-M. Guo, "High-capacity data hiding in halftone images using minimal-error bit searching and least-mean square filter," *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1665–1679, 2006.

[8] R. Y. M. Li, O. C. Au, C. K. M. Yuk, S.-K. Yip, and S.-Y. Lam, "Halftone image data hiding with block-overlapping parity check," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '07)*, vol. 2, pp. II-193–II-196, Honolulu, Hawaii, USA, April 2007.

[9] J.-M. Guo and Y.-F. Liu, "Halftone-image security improving using overall minimal-error searching," *IEEE Transactions on Image Processing*, vol. 20, no. 10, pp. 2800–2812, 2011.

[10] Y. F. Guo, O. C. Au, and K. Tang, "Watermark embedding for multiscale error diffused halftone images by adopting visual cryptography," *International Journal of Digital Crime and Forensics*, vol. 7, no. 1, pp. 51–68, 2015.

[11] Y. F. Guo, O. C. Au, J. T. Zhou, K. Tang, and X. P. Fan, "Halftone image watermarking via optimization," *Signal Processing: Image Communication*, vol. 41, pp. 85–100, 2016.

[12] J.-M. Guo, S.-C. Pei, and H. Lee, "Paired subimage matching watermarking method on ordered dither images and its high-quality progressive coding," *IEEE Transactions on Multimedia*, vol. 10, no. 1, pp. 16–30, 2008.

[13] H. Z. Hel-Or, "Watermarking and copyright labelling of printed images," *Journal of Electronic Imaging*, vol. 10, no. 3, pp. 794–803, 2001.

[14] M. S. Fu and O. C. Au, "Data hiding in ordered dithered halftone images," *Circuits, Systems, and Signal Processing*, vol. 20, no. 2, pp. 209–232, 2001.

[15] S.-C. Pei, J.-M. Guo, and H. Lee, "Novel robust watermarking technique in dithering halftone images," *IEEE Signal Processing Letters*, vol. 12, no. 4, pp. 333–336, 2005.

[16] O. Bulan, G. Sharma, and V. Monga, "Orientation modulation for data Hiding in clustered-dot halftone prints," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2070–2084, 2010.

[17] L. Feng, D. Cong, H. Shu, and B. Liu, "Adaptive halftone watermarking algorithm based on particle swarm optimization," *Journal of Multimedia*, vol. 8, no. 3, pp. 183–190, 2013.

[18] C.-H. Son and H. S. Choo, "Watermark detection from clustered halftone dots via learned dictionary," *Signal Processing*, vol. 102, pp. 77–84, 2014.

[19] I. G. Chun and S. Ha, "A watermarking method for halftone images based on iterative halftoning method," in *E-Commerce and Web Technologies*, vol. 2738 of *Lecture Notes in Computer Science*, pp. 165–175, Springer, Berlin, Germany, 2003.

[20] J.-M. Guo, C.-C. Su, Y.-F. Liu, H. Lee, and J.-D. Lee, "Oriented modulation for watermarking in direct binary search halftone images," *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 4117–4127, 2012.

[21] J.-M. Guo, G.-H. Lai, K. Wong, and L.-C. Chang, "Progressive halftone watermarking using multilayer table lookup strategy," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2009–2024, 2015.

[22] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics—Doklady*, vol. 10, no. 8, pp. 707–710, 1966.