A NOVEL ALGORITHM FOR EMBEDDING AND DETECTING DIGITAL WATERMARKS

Guo-rui Feng, Ling-ge Jiang, Chen He and Dong-jian Wang

Dept. of Electronic Eng., Shanghai Jiaotong Univ., Shanghai 200030, China
e-mail:2082@sjtu.edu.cn, lgjiang@info.sh.cn

ABSTRACT

In this paper, a novel algorithm is proposed to embed and detect digital watermarks. It is different from other algorithms, because the algorithm embeds digital bits in the relationship among pixels, whereas most of other algorithms embed digital bits in term of pixel's position. In the embedding phase one discrete chaotic map is used to create random sequences to encrypt watermarking bits, the other discrete chaotic map is used to permute the original image. Watermarking detection resorts to looking for a path with the minimal distance like Viterbi Algorithm, and it is performed without the original image. Experimental results show that the algorithm can effectively resist the influence of Gaussian Noise, JPEG compression, cropping operation, and recover original watermarks well.

1. INTRODUCTION

Recently Internet develops rapidly, and people can enjoy beautiful images from Internet here and there, so copyright protection becomes more and more important. To deal with unauthentic copyright problem, watermarking techniques are presented. At the same time, in order to increase attackers' difficulties, most of current watermarking schemes utilize encryption techniques.

Early works on watermarks are called Least Significant Bits Algorithm (LSB)[1]. It considers PN sequences as watermarking bits and modifies least significant bits according to watermarking bits in turn. Although this algorithm doesn't almost degrade the quality of the original image, watermarking bits are easily destroyed by attackers. Later, Patchwork Algorithm [2] is presented and it casts watermarking bits by modifying the statistical property of images. Above algorithms both belong to Spatial Domain Algorithm. In order to resist attacking, some scholars [3] propose that extra bits are added to original images to enhance robustness of algorithms. Many chaotic maps are presented to permute the original image [4-7].

This research is supported by Hi-Tech R&D Program of China (No,2002AA144110).

In this paper, a new watermarking algorithm is proposed. First, the original image is divided into 16×16 blocks, then generates embedding bits combined digital watermarking bits with chaotic sequences, at last embeds encrypted watermarking bits by modifying the relationship among pixels and makes them satisfy the minimal distance in the correct path which reflects the information of embedding watermarking bits. Watermarking detection resorts to looking for a path with the minimal distance like Viterbi Algorithm, and it is performed without the original image. According to the path, the algorithm can recover encrypted watermarking bits, then applies same chaotic sequences to generate encrypted bits, at last obtains digital watermarking bits. The MATLAB simulation experiments show the extracted watermarking images are still identified after JPEG compression rate reaches 36% or Gaussian Noise attack with $PSNR = 33.9$.

2. THE NEW WATERMARKING EMBEDING ALGORITHM

In order to convenience descriptions, some symbols are defined. Let $f(x,y)$ denote the gray level at position (x,y) and Z denote the original image of size $M\times M$, so position (x,y) should satisfy $1\le x, y \le M$, W denotes a binary watermarking image of size $N\times N$ bits, where $N<M$.

2.1 Generation of Chaotic sequences

There are many discrete chaotic maps applied to creating random sequences.

Logistic Map: $x_{n+1} = \mu x_n(1-x_n) \quad -1 \le x_n \le 1$

Chebyshev Map: $x_{n+1} = \cos(\mu \cos^{-1} x_n) \quad -1 \le x_n \le 1$

Set $\mu = 4$, above maps can both create chaotic sequences, and their Lyapunov Exponents are 0.693 and 1.3878 respectively, so this algorithm adopts Chebyshev Map with initial value k_1 (first secret key) and in the simulation we set $k_1 = 0.5$. Iterated Chebyshev Map on the initial point $x_0 = k_1$ generates a real value sequences. Foregoing 9 points are abandoned, and every two points, the algorithm remains one point beginning with tenth point. Expand remaining points and generate binary bit flow, i.e. every real number is expressed by the binary system, then take foregoing 8 bits to form $b_1(i)$, $1 \le i \le N \times N$.

2.2 Encrypting the watermarking image

The algorithm generates long sequences by linking every column, and forms $b_2(i)$, $1 \le i \le N \times N$, then creates the encrypted watermarking bits after using the following formula $b(i) = b_1(i) \oplus b_2(i), 1 \le i \le N \times N$, where $\oplus$ is XOR operation. If attackers don't know secret key, they can't recover the watermarking image.

2.3 Schemes of the image permuting

Usually there are two schemes to permute the image in watermarking algorithms, the first is to divide the original image into $n \times n$ blocks directly and locally permute the image in every block; the second is to divide the original image into $n \times n$ blocks after permuting. In this paper we adopt the first scheme. The permuting method is as follows:

Take B blocks with the same size, and set $Z_j (1 \le j \le B)$. The size of every block is 16×16, moreover, assume that the top left corner coordinate of $Z_j (1 \le j \le B)$ is (p_{lj}, q_{lj}), then define $(p_j, q_j) = (p_{lj} + 7, q_{lj} + 7)$. Deal with every block by using $A_N^k : Z_j \to Z_j$, which is defined as follows:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ k_2 & k_2 + 1 \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix} (\text{mod} 16)$$

where $(x_n, y_n) \in Z_j$, $1 \le Z_j \le B$, k_2 is second secrete key. In the simulation, we take $k_2 = 3$ and obtain that the period of A_N^k is 12. Experimental results show that in the permuting phase, when the map is executed 5 times, the performance of permuting the image is the best, hence in the detection phase, the map is executed 5 times too.

2.4 The embedding algorithm

According to values of $b(i)$, modify the corresponding pixel's gray levels. If i is odd, modify pixels along abscissa, otherwise modify pixels along ordinate. The methods are defined as follows:

We take the first block as example. Assume $d_{xy} = f(x, y)(\text{mod} 32)$ and $0 \le d_{xy} < 32$, define $floor(f(x, y), c)$ as follows:

$$floor(f(x, y), c) = f(x, y) + c - d_{xy}$$

a) Determine the gray level of the center point

$$f_{new}(p_1, q_1) = floor(f(p_1, q_1), 24)$$

b) When i is odd in the X_{1i}, modify pixels along abscissa. If $b(i) = 1$, calculate

$$f_{new}(p_1 + \frac{1+i}{2}, q_1) = floor(f(p_1 + \frac{1+i}{2}, q_1), 24)$$

$$f_{new}(p_1 - \frac{1+i}{2}, q_1) = floor(f(p_1 - \frac{1+i}{2}, q_1), 16)$$

if $b(i) = 0$, calculate

$$f_{new}(p_1 - \frac{1+i}{2}, q_1) = floor(f(p_1 - \frac{1+i}{2}, q_1), 24)$$

$$f_{new}(p_1 + \frac{1+i}{2}, q_1) = floor(f(p_1 + \frac{1+i}{2}, q_1), 16)$$

then $i = i + 1$.

c) When i is even in the X_{1i}, modify pixels along ordinate. If $b(i) = 1$, calculate

$$f_{new}(p_1, q_1 + \frac{i}{2}) = floor(f(p_1, q_1 + \frac{i}{2}), 24)$$

$$f_{new}(p_1, q_1 - \frac{i}{2}) = floor(f(p_1, q_1 - \frac{i}{2}), 0)$$

If $b(i) = 0$, calculate

$$f_{new}(p_1, q_1 - \frac{i}{2}) = floor(f(p_1, q_1 - \frac{i}{2}), 24)$$

$$f_{new}(p_1, q_1 + \frac{i}{2}) = floor(f(p_1, q_1 + \frac{i}{2}), 0)$$

then $i = i + 1$.

Repeat above steps until $i = 15$. Up to now the first block has been cast watermarking bits. Using the same method the algorithm casts the rest $b(i)$ into other blocks.

Finally permute all blocks back to readable image. Here, the map A_N^k is executed 7 times again in every block.

3. THE WATERMARKING DETECTION ALGORITHM

The detection algorithm is divided three steps.

Step 1: Divide the image into blocks like step(c) and 5 iterations of A_N^k are executed in every block.

Step 2: In every block, we look for the path with the minimal distance, therefore need define a distance norm. If $A, B \in Z$, define:

$$d(A,B) = [f(A) - f(B)](\mathrm{mod}32), 0 \le d(A,B) < 32 \cdot$$

In order to avoid error spread, the limited function is defined as follows:

$$sp(d(A,B)) = \frac{1}{1+\exp(-\frac{1}{4}d(A,B))} - \frac{1}{2}$$

The detecting trellis diagram is shown in Fig. 1.

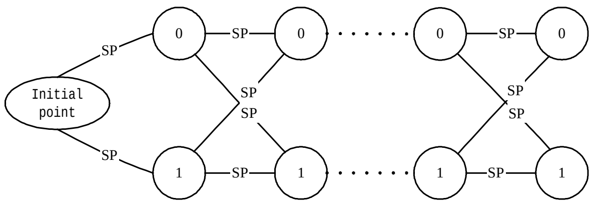

Fig.1. Trellis diagram for jth block

In the extracted watermarks phase, we use a technique like Viterbi Algorithm and obtain encrypted watermarking bits.

Step 3: According to the same Chebyshev Map and initial value k, the algorithm creates the same sequences $b_1(i)$, $1 \le i \le N \times N$. After using the following formula $b_2(i) = b_1(i) \oplus b(i), 1 \le i \le N \times N$, we can obtain watermarking bits.

4. EXPERIMENTAL SIMULATION

In this section, simulation results are presented. For clarity of the presentation we introduce following notations:

$$PSNR = 10\log \frac{255^2}{\frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}[f_{wn}(x,y) - f_w(x,y)]^2}$$

where $f_w(x,y)$ and $f_{wn}(x,y)$ are gray levels of the watermarked image and the watermarked image degraded by Gaussian distributed noise at position (x,y). In experiments we consider the original image "Lena" of size 512×512 and the watermarking image of size 64×64. The experimental results are shown as four groups in Figure 2, Figure 3, Figure 4 and Figure 5 respectively. Figure 2 denotes the original image with $158K$, watermarking image and watermarked "Lena" respectively. Figure 3 denotes the watermarked "Lena" compressed by JPEG to $88K, 58K$ and corresponding extracted watermarking images respectively. Figure 4 denotes the watermarked "Lena" degraded by Gaussian distributed noise with $PSNR = 39.9$, 36.9, 33.9 and corresponding extracted watermarking images respectively. Figure 5 denotes the cropped watermarked "Lena" and corresponding extracted watermarking images respectively.

5. CONCLUSION

In this paper, we present a new watermarking algorithm. By finding the optimal path, the algorithm can effectually recover the watermarking bits. It can effectively resist some attacks such as JPEG compression, Gaussian Noise or cropping. The simulation experiments show the extracted watermarking image is still identified after JPEG compression rate reaches 36%, Gaussian Noise attacks with $PSNR = 33.9$ or cropping operation.

6. REFERENCES

[1] R.G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne. "A digital watermark." in *Proc. IEEE Int. Conf. Image Processing*, Austin, Tex., pp. 86–90, 1994.

[2] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," *IBM Systems Journal*, vol. 35, pp. 313–336, 1996.

[3] M. F. Mansour and A. H. Tewfik, "Efficient decoding of watermarking schemes in the presence of false alarms." in *Proc. 2001 IEEE Workshop. Multimedia Signal*, pp. 523–528, 2001.

[4] A. Nikolaidis and I. Pitas, "Comparison of different chaotic maps with application to image." in *Proc. 2000 IEEE Int. Sym. Circuits and Systems (ISCAS'00)*, Vol. 5, Geneva, pp. 509–512, 2000.

[5] G. Voyatzis and I. Pitas, "Chaotic watermarks for embedding in the spatial digital image domain." in *Proc. 1998 IEEE Int. Conf. Image Processing, (ICIP'98)*, Vol. 2, pp. 432–436, 1998.

[6] G. Voyatzis and I. Pitas, "Applications of toral automorphisms in image watermarking." in *Proc. 1996 IEEE Int. Conf. Image Processing, (ICIP'96)*, Vol. 2, pp. 237–240, 1996.

[7] Jui-Cheng Yen, "Watermarks embedded in permuted domain." *Electronics Letters*, Volume: 37, pp. 80–81, 2001.

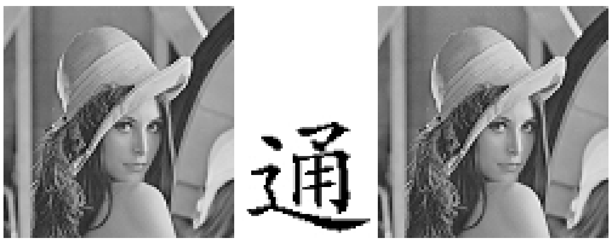

Fig. 2. (a) Original "Lena", (b) watermarking image, (c) watermarked "Lena".

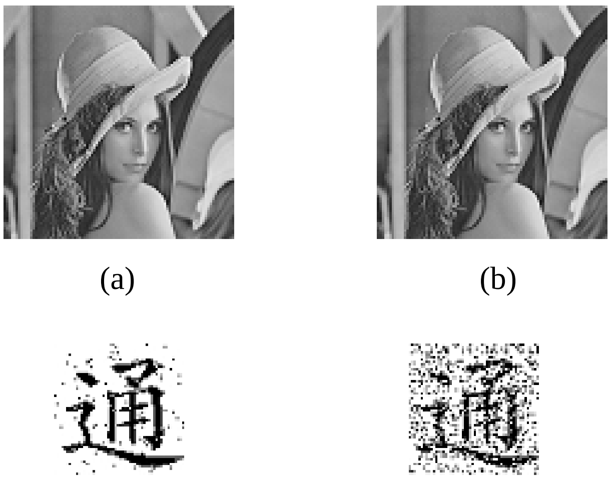

Fig. 3. (a), (b) are watermarked "Lena" compressed by JPEG to 88K, 58K, (c),(d) are corresponding extracted watermarking images.

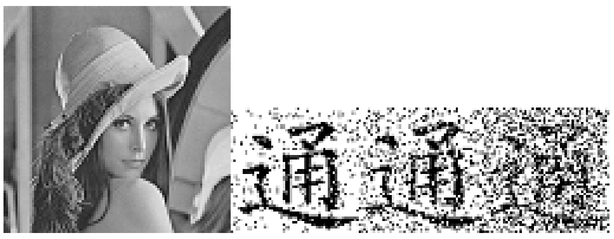

Fig. 4: (a), (b), (c) are watermarked "Lena" degraded by Gaussian Noise with PSNR=39.9, PSNR=36.9, PSNR=33.9, (d), (e), (f) are corresponding extracted watermarking images.

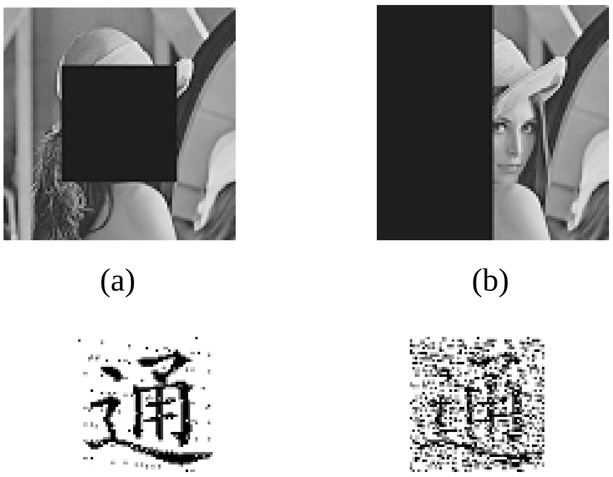

Fig 5: (a), (b) are cropped watermarked "Lena", (c), (d) are corresponding extracted watermarking images.