Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT

PERGAMON

Pattern Recognition 37 (2004) 555–565

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

Genetic watermarking based on transform-domain techniques

Chin-Shiuh Shieh[a], Hsiang-Cheh Huang[b,*], Feng-Hsing Wang[c], Jeng-Shyang Pan[a,d]

[a] *Department of Electronic Engineering, Nat'l Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan, ROC*
[b] *Department of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan, ROC*
[c] *School of Electrical and Information Engineering, University of South Australia, Adelaide, Australia*
[d] *Department of Automatic Test and Control, Harbin Institute of Technology, Harbin, China*

Received 20 November 2002; accepted 25 July 2003

Abstract

An innovative watermarking scheme based on genetic algorithms (GA) in the transform domain is proposed. It is robust against watermarking attacks, which are commonly employed in the literature. In addition, the watermarked image quality is also considered. In this paper, we employ GA for optimizing both the fundamentally conflicting requirements.

Watermarking with GA is easy for implementation. We also examine the effectiveness of our scheme by checking the fitness function in GA, which includes both factors related to robustness and invisibility. Simulation results also show both the robustness under attacks, and the improvement in watermarked image quality with GA.

Keywords: Digital watermarking; Genetic algorithm (GA); Discrete cosine transform (DCT); Peak signal-to-noise ratio (PSNR); Normalized cross correlation (NC); Fitness function

1. Introduction

With the widespread use of Internet and the development in computer industry, the digital media, including images, audios, and video, are easily acquired in our daily life. Digital multimedia contents suffer from infringing upon the copyrights with the digital nature of unlimited duplication, easy modification and quick transfer over the Internet. As a result, data piracy has become a serious issue. Hence, some copyright protection schemes need to be employed to conquer this problem. In this paper, we concentrate our research topic on image watermarking for copyright protection.

Digital watermarking for images is one way to embed the secret information, or the *watermark*, into the original image itself to protect the ownership of the original sources [1–3]. On the one hand, the watermarking schemes can be categorized as "visible" and "invisible" watermarking. Comparisons between the two schemes are listed as follows:

1. The visible watermarks, for instance, are those company logos on one corner of the TV screen when we watch TV programs. Although the logos, or watermarks, are easily identified, they are usually not robust against image cropping. Therefore, visible watermarks can be easily removed from original images.
2. The invisible watermarks are more secure and robust than the visible watermarks. The embedding locations are secret, and only the authorized persons with the secret keys in the watermarking system can extract the secret watermark. The watermarked image should look similar to the original one, and should not cause suspicion by others.

On the other hand, digital watermarking can also be categorized as "robust" and "fragile" watermarking. Robust watermarks are designed to have the ability to detect the

* Corresponding author. Tel.: +886-918-952-075; fax: +886-3-573-1791.
E-mail address: huangh@cc.nctu.edu.tw (H.-C. Huang).

0031-3203/$30.00
doi:10.1016/j.patcog.2003.07.003

watermark after some image processing operations, called attacks. After certain attacks and the watermark extraction process, the extracted watermarks should be highly correlated with the embedded ones. That is, the extracted watermarks need be recognizable in the robust watermarking system. In contrast, for fragile watermarks, they are designed to become invalid after even the slightest modification of the watermarked image. Because the watermarks become undetectable, they do not resist intentional or unintentional attacks. Therefore, fragile watermarks are mainly used for authentication purposes.

There are a variety of schemes for embedding the watermark into the original image [4]. Typical schemes for digital watermarking were based on transform-domain techniques with discrete cosine transform (DCT) [5–7], discrete wavelet transform (DWT) [8], discrete Fourier transform (DFT) [9], spatial-domain methods [10,11], and vector quantization (VQ) domain schemes [12]. The above-mentioned schemes employ the embedding of the watermark into some of the selected coefficients in their corresponding domains, which might be fixed in a pre-determined set of coefficients. One major disadvantage for these typical schemes is that during transmission over the Internet or the mobile channels, the watermarked images might be processed, or attacked, in order to remove the existence of the watermark [13,14]. When the attackers dissolve the relationships between the original multimedia and the pre-determined set for watermark embedding, the watermarking capability for copyright protection no longer exists. Another disadvantage for typical schemes is how to decide and choose the pre-determined set. For watermark embedding in the DCT domain, if we embed the watermark in the higher frequency bands, even though the watermarked image quality is good, it is vulnerable to the low pass filtering (LPF) attack. Thus, embedding into the higher frequency bands coefficients is not robust, although the watermarked image quality is assured. In contrast, if we embed the watermark into the coefficients in the lower frequency bands, it should be robust against common image processing attacks such as the LPF attack. However, embedding in the lower frequency bands will cause the resulting watermarked image quality greatly degrades to compare with the original image. This comes from the fact that the energies of most natural images are concentrated in lower frequency bands, and the human eyes are more sensitive to the noise caused by modifying the lower frequency coefficients. Hence, aside from the two observations above, some researchers claim to embed the watermarks into the "middle-frequency bands" to serve as a trade-off for watermark embedding in the transform domain [5].

Therefore, from the observations and explanations above, we make use of genetic algorithm (GA) [15,16] to find the optimal frequency bands for watermark embedding into our DCT-based watermarking system, which can simultaneously improve security, robustness, and image quality of the watermarked image. Because the scheme operates in the transform domain, it contains three main parts, including image transformation, watermark embedding, and watermark extraction.

This paper is organized as follows. We describe the fundamental concepts of genetic algorithms in Section 2. Section 3 demonstrates the algorithm for embedding the watermark in the DCT domain with GA. Section 4 depicts the watermark extraction algorithm. Section 5 illustrates the simulation results, and we also show the superiority of our scheme over the results proposed by other researchers in this section. Section 6 briefly discusses with the proposed algorithm and the simulation results. And we conclude this paper in Section 7.

2. Fundamental concepts of genetic algorithms

Conventional search techniques are often incapable of optimizing non-linear functions with multiple variables. One scheme called the "genetic algorithm" (GA), based on the concept of natural genetics, is a directed random search technique. The exceptional contribution of this method was developed by Holland [15] over the course of 1960s and 1970s, and finally popularized by Goldberg [16].

In the genetic algorithms, the parameters are represented by an encoded binary string, called the "chromosome". And the elements in the binary strings, or the "genes", are adjusted to minimize or maximize the fitness value. The fitness function generates its fitness value, which is composed of multiple variables to be optimized by GA. For every iteration in GA, a pre-determined number of individuals will correspondingly produce fitness values associated with the chromosomes. Fig. 1 demonstrates the flow chart for a typical binary GA. It begins by defining the optimization parameters, the fitness function, and the fitness value, and it ends by testing for convergence. According to the applications for optimization, designers need to carefully define the necessary elements for training with GA. Then, we are able to evaluate the fitness function in addition to the terminating criteria with the natural selection, crossover, and mutation operations in a reasonable way [17].

Assuming that we employ GA to search for the largest fitness value with a given fitness function. In GA, as shown in Fig. 1, the core components are depicted as follows.

1. *Select mate*: A large portion of the low fitness individuals is discarded through this natural selection step. Of the N individuals in one iteration, only the top N_{good} individuals survive for mating, and the bottom $N_{\text{bad}} = N - N_{\text{good}}$ ones are discarded to make room for the new offspring in the next iteration. Therefore, the selection rate is N_{good}/N.
2. *Crossover*: Crossover is the first way that a GA explores a fitness surface. Two individuals are chosen from N_{good} individuals to produce two new offspring. A crossover point is selected between the first and last chromosomes of the parents' individuals. Then, the fractions of each

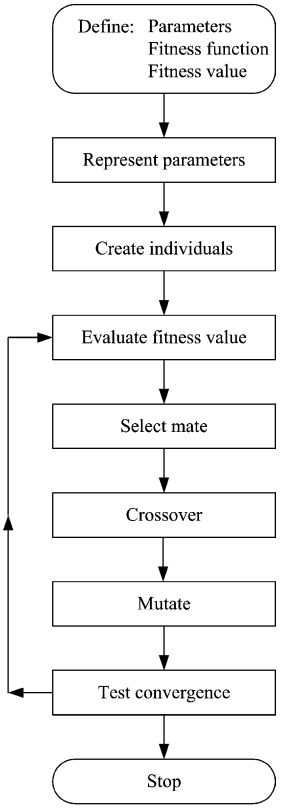

Fig. 1. The flow chart of genetic algorithms.

individual after the crossover point are exchanged, and two new offspring are produced.

3. *Mutate*: Mutation is the second way that a GA explores a fitness surface. It introduces traits not in the original individuals, and keeps GA from converging too fast. The pre-determined mutation rate should be low. Most mutations deteriorate the fitness of an individual, however, the occasional improvement of the fitness adds diversity and strengthens the individual.

After obtaining the fundamental concepts in GA, we are able to design an optimized DCT-based watermarking system with the aid of GA.

3. The embedding algorithm

Let the input image be $\boldsymbol{X}$ with size $M \times N$. Our goal is to embed a robust watermark into the DCT frequency bands of $\boldsymbol{X}$, and have a watermarked reconstruction $\boldsymbol{X}'$ after optimization.

Before the embedding procedure, we need to transform the spatial domain pixels into DCT domain frequency bands. We perform the 8×8 block DCT on $\boldsymbol{X}$ first and get the coefficients in the frequency bands, $\boldsymbol{Y}$,

$$\boldsymbol{Y} = \mathrm{DCT}(\boldsymbol{X}) \tag{1}$$

and

$$\boldsymbol{Y} = \bigcup_{m=1}^{M/8} \bigcup_{n=1}^{N/8} \boldsymbol{Y}_{(m,n)}. \tag{2}$$

For one non-overlapping block (m,n) in $\boldsymbol{X}$, the resulting 64 DCT bands $\boldsymbol{Y}_{(m,n)}$ can be represented by

$$\boldsymbol{Y}_{(m,n)} = \bigcup_{k=0}^{63} \{Y_{(m,n)}(k)\}, \quad 1 \leqslant m \leqslant \tfrac{M}{8}, \quad 1 \leqslant n \leqslant \tfrac{N}{8}. \tag{3}$$

$Y_{(m,n)}(k)$ are zigzag ordered DCT coefficients, which can be shown in Fig. 2. Afterwards, we are able to embed the watermark in the DCT domain.

Assuming that the binary-valued watermark to be embedded is $\boldsymbol{W}$, having size $M_W \times N_W$. A pseudo-random number traversing method [1] is applied to permute the watermark to disperse its spatial relationship. With a pre-determined key, key_0, in the pseudo-random number generating system, we have the permuted watermark $\boldsymbol{W}_P$,

$$\boldsymbol{W}_P = \mathrm{permute}(\boldsymbol{W}, \mathrm{key}_0). \tag{4}$$

And we use $\boldsymbol{W}_P$ for embedding the watermark bits into the selected DCT frequency bands.

To embed the binary watermark into the original source, we need to adopt some relationships, or the *polarities*, $\boldsymbol{P}$, between $\boldsymbol{Y}$ and $\boldsymbol{W}_P$. The meanings of polarities will further be explained in Eqs. (6) and (7). The frequency set, $\boldsymbol{F}$, which will take both the imperceptibility and robustness requirements into account, will be chosen to embed $\boldsymbol{W}_P$ after GA-training. For each 8×8 non-overlapping block in the image, only the coefficients in $(64 \times M_W \cdot N_W / M \cdot N)$ frequency bands will be included in the frequency set, $\boldsymbol{F}$, which are then modified for watermark embedding. Before GA training, the frequency set can be expressed by

$$\boldsymbol{F} = \bigcup_{m=1}^{M/8} \bigcup_{n=1}^{N/8} \{F_{(m,n)}(i) = Y_{(m,n)}(k)\}, \tag{5}$$

where $k = 1, 2, \ldots, 63$, and $i = 0, 1, \ldots, (64 \cdot M_W \cdot N_W / M \cdot N - 1)$ are the randomly selected $(64 \cdot M_W \cdot N_W / M \cdot N)$ frequency bands in $\boldsymbol{F}$ out of the 63 AC coefficients in the DCT domain (except for the DC coefficient) in each block. In our algorithm, the DC coefficient of every block is fixed for the reference in watermark embedding. We randomly choose the frequency bands in each 8×8 block for the initialization of the zeroth iteration in GA. One example for randomly choosing the frequency bands is illustrated in Fig. 2, where the four colored blocks, or the 6th, 9th, 12th, and 29th

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

Fig. 2. The mathematical representations for the zigzag ordered DCT coefficients $Y_{(m,n)}(\cdot)$. The randomly selected bands for watermark embedding are shown in colored positions.

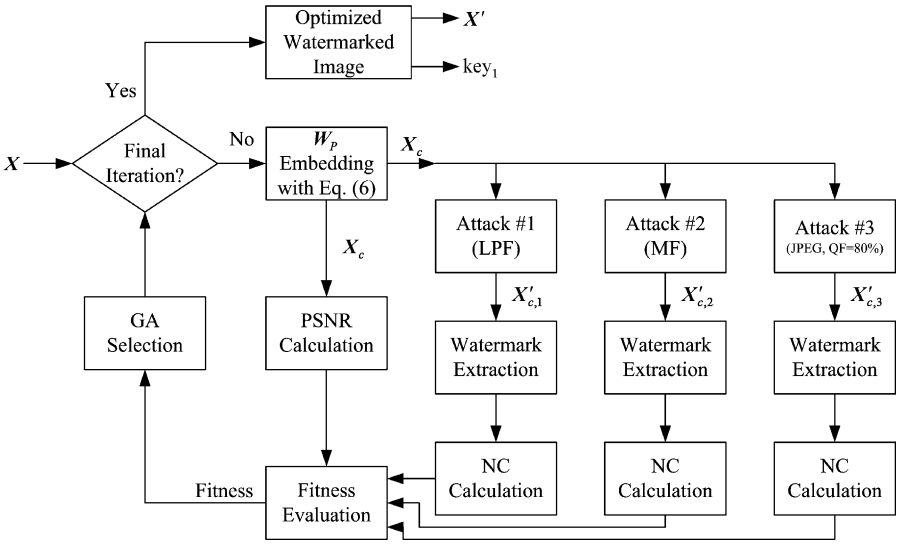

Fig. 3. The block diagram for watermark embedding in the GA-based watermarking system.

AC coefficients, $\{Y_{(m,n)}(6), Y_{(m,n)}(9), Y_{(m,n)}(12), Y_{(m,n)}(29)\}$ in block (m,n), denote the four randomly selected frequency bands to embed the watermark bits. However, randomly selecting the bands might cause the degradation in watermarked image quality and its robustness. Thus, by applying GA, the DCT frequency bands are chosen in $\boldsymbol{F}$ for $\boldsymbol{W}_P$ to watch both the watermarked image quality and the robustness under certain attacks in every training iteration. Consequently, GA determines the embedding locations in the DCT domain in Eq. (5). The block diagram for illustrating watermark embedding with GA is shown in Fig. 3.

Once the bands in the frequency set $\boldsymbol{F}$ are selected in the training process, we designate the mapping between i and k in Eq. (5) as the secret key, key_1. After completing GA training, key_1 is transmitted over an open network by integrating cryptography with our watermarking technology [18]. Next, we generate a reference table, $\boldsymbol{R} = \{R(i)\}$, $i \in \boldsymbol{F}$,

with the DCT coefficients in every frequency band $\boldsymbol{Y}$ by using the ratios between the DC and AC coefficients, which is denoted by

$$R(i)=\sum_{m=1}^{M/8}\sum_{n=1}^{N/8}\left(\frac{Y_{(m,n)}(0)}{Y_{(m,n)}(i)}\right),\quad i\in[1,63]. \tag{6}$$

We use the DC value of every block, e.g., $Y_{(m,n)}(0)$ in $\boldsymbol{Y}_{(m,n)}$ of block (m,n), as a reference value, and produce the relationships among the DC value of one block, the current AC coefficients for embedding, and the reference table for further operation with $\boldsymbol{W}_P$. Then, we can calculate the polarities, $\boldsymbol{P}=\bigcup_{m=1}^{M/8}\bigcup_{n=1}^{N/8}\bigcup_{i\in\boldsymbol{F}}\{P_{(m,n)}(i)\}$, of the DCT coefficients in the frequency set by

$$P_{(m,n)}(i)=\begin{cases}1 & \text{if } (Y_{(m,n)}(i)\cdot R(i))\geqslant Y_{(m,n)}(0),\ \ i\in\boldsymbol{F},\\ 0 & \text{otherwise;}\end{cases} \tag{7}$$

$$Y'_{(m,n)}(i)=\begin{cases}Y_{(m,n)}(i) & \text{if } P_{(m,n)}(i)=W_{P(m,n)}(i)=0,\ \ i\in\boldsymbol{F},\\ Y_{(m,n)}(i)+1 & \text{if } P_{(m,n)}(i)=0 \text{ and } W_{P(m,n)}(i)=1,\ i\in\boldsymbol{F},\\ Y_{(m,n)}(i) & \text{if } P_{(m,n)}(i)=W_{P(m,n)}(i)=1,\ \ i\in\boldsymbol{F},\\ Y_{(m,n)}(i)-1 & \text{otherwise.}\end{cases} \tag{8}$$

$\boldsymbol{Y}$ remains unchanged if $i\notin\boldsymbol{F}$. Next, we obtain the watermarked DCT coefficients for every frequency band, $\boldsymbol{Y}'$,

$$\boldsymbol{Y}'=\bigcup_{m=1}^{M/8}\bigcup_{n=1}^{N/8}\bigcup_{k=0}^{63}\{Y'_{(m,n)}(k)\}. \tag{9}$$

After embedding with the polarities in every GA iteration, we are able to perform inverse DCT on $\boldsymbol{Y}'$, and get the watermarked image of the current iteration, $\boldsymbol{X}_c$,

$$\boldsymbol{X}_c=\text{inverse\_DCT}(\boldsymbol{Y}'). \tag{10}$$

According to the definitions in statistics, the mean squared error (MSE) between the original and watermarked images is defined by

$$\text{MSE}_c=\frac{1}{M\cdot N}\sum_{i=1}^{M}\sum_{j=1}^{N}(X(i,j)-X_c(i,j))^2, \tag{11}$$

where $X(i,j)$ and $X_c(i,j)$ denote the pixel value at position (i,j) of the original image $\boldsymbol{X}$ and the watermarked images of the current iteration c, $\boldsymbol{X}_c$, respectively. Consequently, the watermarked image quality is represented by the peak signal-to-noise ratio (PSNR) between $\boldsymbol{X}$ and $\boldsymbol{X}_c$, formulated the by

$$\text{PSNR}_c=10\log_{10}\cdot\left(\frac{255^2}{\text{MSE}_c}\right)\ \ (\text{dB}). \tag{12}$$

Next, we apply the attacking schemes on $\boldsymbol{X}_c$, and the attacked images associated with the attacking schemes are denoted by $\boldsymbol{X}'_{c,p}$, where p is the number of attacking schemes. There is a watermark attacking benchmark, called "Stirmark" [19], to evaluate the robustness of the watermarking algorithms. Not all watermarking applications require robustness to all possible signal processing operations. In addition, the watermarked image after attacks needs to be worthy of using or transmitting by others; therefore, some attack like image-cropping is not employed in our GA training procedure. In this paper, we employ three major attacking schemes from Stirmark, namely, low-pass filtering (LPF) attack [20], median filtering (MF) attack [20], and JPEG attack with quality factor of 80% [21], hence $p=3$. We extract the watermarks from $\boldsymbol{X}'_{c,p}$, and calculate the normalized cross-correlation (NC) values [1,5] between the embedded watermarks and the extracted ones. The NC between the embedded watermark, $W(i,j)$, and the extracted watermark, $W'(i,j)$, is defined by

$$\text{NC}=\frac{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}[W(i,j)\cdot W'(i,j)]}{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}[W(i,j)]^2}, \tag{13}$$

which is the cross-correlation normalized by the energy of the watermark to give unity as the peak correlation.

After obtaining the PSNR in the watermarked image and the three NC values after attacking, we are ready to start the GA training process. According to the definition of GA, we need to assign the *fitness function* in the cth iteration with

$$f_c=\text{PSNR}_c+\sum_{h=1}^{p}(\text{NC}_{c,h}\cdot\lambda_{c,h}), \tag{14}$$

where f_c, p, and $\lambda_{c,h}$ are the fitness value, the number of attacking schemes and the weighting factors for the NC values, respectively [15]. In Eq. (14), PSNR_c plays the role of imperceptibility measure, while $\text{NC}_{c,h}$ plays the role of robustness measure. Because the PSNR values are dozens of times larger than the associated NC values in the GA fitness function, we need to magnify the NC values with the weighting factors $\lambda_{c,h}$ in the fitness function to balance the influences caused by both the imperceptibility and robustness requirements [17].

Completing the whole procedures in one iteration, we feedback the selected individuals, or the watermarked images survived with the larger fitness values of the current iteration, $\boldsymbol{X}_c$, for further training with the crossover, mutation, and selection procedures in the next GA iteration. After completing the pre-determined number of iterations, we output both the watermarked image, $\boldsymbol{X}'$, and the secret

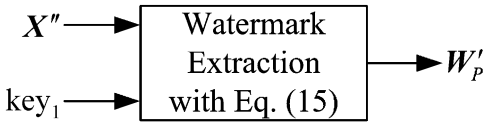

Fig. 4. The block diagram for watermark extraction in the GA-based watermarking system.

key from Eq. (5), key_1, and transmit them to the receiver over the Internet or the wireless channels, which are depicted in Fig. 3. For transmitting the secret key, key_1, we can integrate cryptography with our watermarking schemes to protect the copyright in an open network such as the Internet [18].

4. The extraction algorithm

In extracting the watermarks, the original image $\boldsymbol{X}$ is not required in our algorithm. However, the optimized watermarked image might be subject to some intentional or unintentional attack, and the resulting image after attack is represented by $\boldsymbol{X}''$. We calculate the DCT of the watermarked image after attacking $\boldsymbol{Y}''$, in the attacked $\boldsymbol{X}''$, with the secret key corresponding to the frequency set $\boldsymbol{F}$, key_1. We then reproduce the estimated reference table $\boldsymbol{R}'$ from the attacked $\boldsymbol{X}''$ by following the operations in Eq. (6), and we are able to extract the permuted watermark,

$$W'_{P,(m,n)}(i) = \begin{cases} 1 & \text{if } Y''_{(m,n)}(i) \cdot R'(i) \geqslant Y''_{(m,n)}(0) \ \ \forall i; \\ 0 & \text{otherwise;} \end{cases} \tag{15}$$

$$\boldsymbol{W}'_P = \bigcup_{m=0}^{M/M_W-1} \bigcup_{n=0}^{N/N_W-1} W'_{P,(m,n)}(i), \qquad i \in \boldsymbol{F}. \tag{16}$$

Finally, we use key_0 in Eq. (4) to acquire the extracted watermark $\boldsymbol{W}'$ from $\boldsymbol{W}'_P$,

$$\boldsymbol{W}' = \text{inverse\_permute}(\boldsymbol{W}'_P, \text{key}_0). \tag{17}$$

The block diagram for illustrating watermark extraction in the GA-watermarking system is depicted in Fig. 4.

5. Simulation results

In our simulation, we take the well-known test image, `Lena`, with size 512 × 512, as the original source, which is shown in Fig. 5. We have the embedded watermark, `rose`, with size 128 × 128, shown in Fig. 6. Hence, the number of bits to be embedded in one 8 × 8 non-overlapping block is $128^2/512^2 \cdot 64 = 4$. Next, taking the frequency set in Ref. [5] to be the initial set $\boldsymbol{F}$, i.e., $\boldsymbol{F} = \{Y_{(m,n)}(14), Y_{(m,n)}(15), Y_{(m,n)}(16), Y_{(m,n)}(27)\}$, for every block in our simulation. After watermark embedding

Fig. 5. The original test image `Lena` with size 512 × 512.

Fig. 6. The watermark `rose` with size 128 × 128.

in the DCT domain, we take the inverse DCT, and obtain the watermarked image for the zeroth iteration. We apply three attacks, namely, LPF, MF, and JPEG compression with quality factor 80% attacks mentioned above. Next, the resulting PSNR of the watermarked image, and the three NC values after attacking, work together to evaluate the fitness function.

In the GA training process, we choose ten individuals for every iteration, with the crossover rate of 0.25 and mutation rate of 0.05. The training iterations are set to 200. The five individuals with larger fitness values are reserved for the new individuals in the next iteration, thus the selection rate is 0.5. We simulate two cases with the different weighting factors. For simplicity, we use $p = 3$, $\lambda_{c,h} = 10$ and $\lambda_{c,h} = 30$, $\forall c, h$, in Eq. (14).

The watermarked images with our algorithm are depicted in Figs. 7 and 8, and their corresponding extracted watermarks after attacks are represented in Figs. 9 and 10. They are also tabulated in Tables 1 and 2 by comparing the PSNR and NC values with the increase of iteration numbers.

Fig. 7. The watermarked image at the 0th iteration in GA. PSNR = 30.19 dB.

Fig. 8. The watermarked image at the 200th iteration in GA. PSNR = 34.79 dB.

Figs. 7 and 8 represent the watermarked image at the 0th and 200th iteration in GA, with the PSNR of 30.19 and 34.79 dB, respectively. Figs. 9(a)–(c) show the extracted watermarks at the 0th iteration with the initial set $\boldsymbol{F}$, and Figs. 10(a)–(c) demonstrate their corresponding ones at the 200th iteration. We can observe the improvements in both the watermarked image quality and the NC values after certain attacks with the aid of GA. By observing the data in Tables 1 and 2, we find that both the PSNR and NC values increase with the

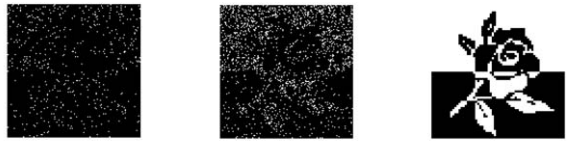

Fig. 9. The extracted watermarks and the NC values at the 0th iteration of the proposed algorithm under various attacking methods with $\lambda = 10$.

Fig. 10. The extracted watermarks and the NC values at the 200th iteration of the proposed algorithm under various attacking methods with $\lambda = 10$.

increase of iteration numbers. The selection of λ would influence both the watermarked image quality and the robustness measure at the final output $\boldsymbol{X}'$ in Fig. 3 if we change the weighting factor λ when we compare the results in Tables 1 and 2. As we mentioned in Section 2, we need to balance the influences caused by the elements in the GA fitness function in Eq. (14). Because the typical watermarked image quality is in the range of 30–40 dB, and the NC values may approach 1, if we set $\lambda_{c,h} = 10$, we have the better results in Table 1. In contrast, if we set $\lambda_{c,h} = 30$, the results in Table 2 are inferior to those in Table 1, because we superimpose the effects caused by NC.

In our algorithm, the bands for the watermark to be embedded, $\{F_{(m,n)}(i)\}$, differ from one block to another. The selected bands also differ from one test image to another. Therefore, from a statistical point of view, we record the number of occurrence in the 63 embedded frequency bands in all the blocks within our test image, and the histogram is shown in Fig. 11 for test image `Lena`. From the simulation data with $\lambda = 10$, we observe that $Y(6)$, $Y(9)$, $Y(11)$, and $Y(12)$ are the four bands to acquire the best fitness value. A simple application of the best bands that we acquire is to replace the random selection of embedded bands in Ref. [5] by the best bands after GA training. Both our algorithm and the algorithm in Ref. [5] embed a binary logo into the original image. In Table 3, the PSNR and NC values with the method in Ref. [5] are inferior to our results with the best band after GA training. If there are limitations in practical implementations of our algorithm, we can directly use

Table 1
The PSNR and NC values for Lena under different GA iterations with $\lambda = 10$

Iteration	PSNR (dB)	NC_1 (LPF)	NC_2 (MF)	NC_3 (JPEG)
0	30.19	0.5261	0.5759	1.0
50	34.61	0.7245	0.7768	1.0
100	34.73	0.7361	0.7901	1.0
150	34.77	0.7413	0.7933	1.0
200	34.79	0.7426	0.7947	1.0

Table 2
The PSNR and NC values for Lena under different GA iterations with $\lambda = 30$

Iteration	PSNR (dB)	NC_1 (LPF)	NC_2 (MF)	NC_3 (JPEG)
0	30.19	0.5261	0.5759	1.0
50	33.31	0.6861	0.7350	1.0
100	33.77	0.7011	0.7516	1.0
150	33.97	0.7077	0.7604	1.0
200	34.09	0.7101	0.7634	1.0

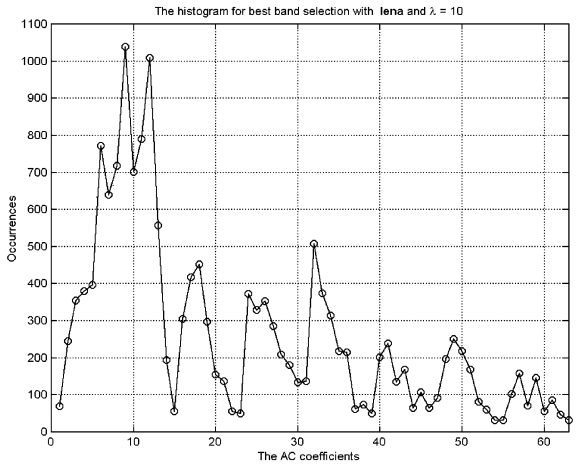

Fig. 11. The best band vs. number of occurrence plot with $\lambda = 10$ for the whole watermarked image. We observe that $Y(6)$, $Y(9)$, $Y(11)$, and $Y(12)$ are the ones with the best fitness values for embedding Lena.

the frequency bands with the largest number of occurrences for embedding the watermark bits into every block of the test image, i.e., we embed the watermark into the same frequency bands for every 8×8 block. Although the simulation results are not good enough as those after GA training, they are better than the results with random selection of frequency bands in Ref. [5].

We also conduct experiments on the two other well-known test images pepper and baboon. The histograms for the occurrences of frequency bands with $\lambda = 10$ are illustrated in Figs. 12 and 13, respectively. The best four bands for pepper are $Y(9)$, $Y(10)$, $Y(11)$, and $Y(12)$, and their counterparts for baboon are also $Y(9)$, $Y(10)$, $Y(11)$, and $Y(12)$. By comparing the three histograms from Figs. 11 to 13, we can see that the distributions in the three histograms are somewhat different because of the different spatial domain characteristics of the test images, although the best four frequency bands for both pepper and baboon are the same. Therefore, GA provides an effective means for designing a robust watermarking system to deal with the original images with different characteristics.

Besides the LPF, MF, and JPEG attacks employed in the GA training process, we also test the effectiveness of our algorithm to cope with the image-cropping attack. We perform attacks on the watermarked image by cropping 10%, 25%, and 40% of its surroundings. The resulting image with 40% cropping is shown in Fig. 14 for reference. We use key_1 in Eq. (5) for watermark extraction. The extracted watermarks under different cropping attacks are illustrated in Figs. 15(a)–(c), by cropping 10%, 25%, and 40% of areas in the watermarked image, respectively. The extracted watermarks survive well even under the cropping by 40% attack by using the proposed algorithm with GA. This proves the effectiveness of our algorithm.

Other attacking schemes, including some popular attacks in Stirmark [19], will be carefully chosen, and will be integrated into the GA-based watermarking algorithm in our future research. It is also the future work to test our algorithms for more attacks other than the image-cropping attacks presented in Fig. 15.

Table 3
The PSNR and NC values comparisons for Lena between the best bands trained with $\lambda = 10$ and the bands in Ref. [5]

	Selected ands	PSNR (dB)	NC_1 (LPF)	NC_2 (MF)	NC_3 (JPEG)
Results in Ref. [5]	$Y(14)$, $Y(15)$, $Y(16)$, $Y(27)$	30.19	0.5261	0.5759	1.0
Our results	$Y(6)$, $Y(9)$, $Y(11)$, $Y(12)$	32.52	0.6624	0.7134	1.0

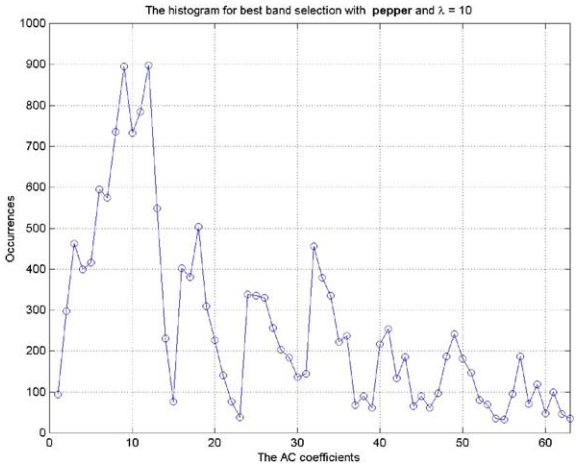

Fig. 12. The best band vs. number of occurrence plot with $\lambda = 10$ for the whole watermarked image. We observe that $Y(9)$, $Y(10)$, $Y(11)$, and $Y(12)$ are the ones with the best fitness values for embedding pepper.

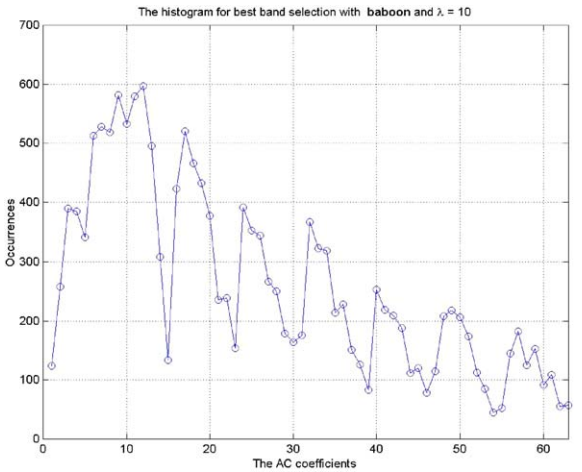

Fig. 13. The best band vs. number of occurrence plot with $\lambda = 10$ for the whole watermarked image. We observe that $Y(9)$, $Y(10)$, $Y(11)$, and $Y(12)$ are the ones with the best fitness values for embedding baboon.

6. Discussions

The goal of GA is to find an optimized solution under several conflicting requirements. In this paper, we select the two conflicting requirements for typical watermarking systems, namely, the watermarked image quality and the robustness of the watermarking algorithm. The simulation results in Section 5 also prove the effectiveness of our GA-based watermarking algorithm.

We observed the following points in our discussion for this paper:

1. In the fitness function, the parameters to be optimized are the watermarked image quality, represented by PSNR values, and the robustness of extracted watermarks, represented by NC values. Because PSNR values are dozens

Fig. 14. Watermarked image attacked by cropping 40% of its surroundings.

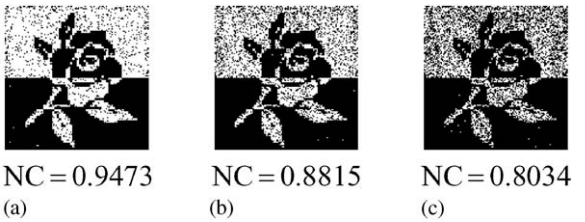

Fig. 15. The extracted watermarks under different attacking schemes. (a) Cropping by 10%. (b) Cropping by 25%. (c) Cropping by 40%.

of times larger than the NC values, we need to increase the influences cased by NC values. From the simulation results, we find that when we set the weighting factor to 10, we get better results. It is because there are three attacks associated with the corresponding NC values, consequently, for $\lambda = 10$, the effects caused by the two conflicting requirements might be balanced. Therefore, it is important to determine the conflicting requirements associated with their weighting factors in the fitness function before designing the GA-based watermarking system.

2. The parameters in GA should be carefully chosen to obtain the optimized output within a reasonable period of time. The number of individuals in our simulations is set to 10, and attacking schemes are limited to 3. With the settings above, the resulting computation time per iteration is 2 minutes with a Pentium-IV 1.8 GHz computer. Therefore, parameter selection is important to produce a good or acceptable output while keeping a reasonable time for computation.
3. It is clear that not all watermarking applications require robustness to all possible signal processing operations. In this paper, we aim at coping with the attacks to remove high frequency redundancies, which are commonly employed in literature. Also, some image-cropping attacks are also tested and the algorithm is robust to the image-cropping attack. We can change the other attacking schemes by substituting the attacking modules in the GA training process.
4. After GA training with the three test images above, the best four bands for watermark embedding are $\{Y(6), Y(9), Y(11), Y(12)\}$ and $\{Y(9), Y(10), Y(11), Y(12)\}$, respectively. Among the 63 AC frequency bands, the bands that we selected fit the assumption of the middle-frequency bands, as explained in Section 3.
5. The fitness value in GA increases with the increase in iteration numbers. And we can see the improvements in both the watermarked image quality, from 30.19 to 34.79 dB, and the NC values, from 0.5300 to 0.7426, if we take the LPF attack for instance. Therefore, GA provides an effective means for watermarking, with the carefully determined fitness function.
6. As we see from the watermark embedding procedure in Fig. 3, every block therein is a separate module in building the whole system. Hence, we can change the attacking schemes by substituting the blocks for attacking in Fig. 3. The proposed system is also applicable to watermarking in the spatial domain or in the wavelet domain, by changing the DCT embedding algorithm into the spatial or wavelet embedding algorithms. We may also substitute the attacking modules by other attacking schemes in literature. And this adds the flexibility for the application of the proposed structure. We can also employ the human visual system (HVS) model [2,3,14] for evaluating the watermarked image quality, and this part of work is being in progress for the future works of this paper.

7. Conclusion

A robust algorithm for DCT-based GA-watermarking has been presented in this paper. It is robust because we make use of GA to train the frequency set for embedding the watermark. In addition to the robustness of the proposed algorithm, we also improve the watermarked image quality with the aid of GA.

Simulation results reveal that if we just borrow the concepts of existing algorithms, both the watermarked image quality and the NC values of the extracted watermarks after certain attacks will be poor. Hence, GA offers a systematic way to consider the improvements of the fitness functions. With the simulation results under a variety of attacking techniques, we are able to claim its robustness and superiority over the existing algorithm with the proposed techniques. In comparison with the existed methods, watermark embedding with our scheme can get better-watermarked image qualities and the higher NC values in the extracted watermarks.

Acknowledgements

This work was supported by National Science Council (Taiwan, ROC) under Grant No. NSC91-E-2219-151-002.

References

[1] F.A.P. Petitcolas, R.J. Anderson, M.G. Kuhn, Information hiding—a survey, Proc. IEEE 87 (7) (1999) 1062–1078.
[2] F. Hartung, M. Kutter, Multimedia watermarking techniques, Proc. IEEE 87 (7) (1999) 1079–1107.
[3] D. Artz, Digital steganography: hiding data within data, IEEE Internet Comput. 5 (3) (2001) 75–80.
[4] C.I. Podilchuk, E.J. Delp, Digital watermarking: algorithms and applications, IEEE Signal Process. Mag. 18 (4) (2001) 33–46.
[5] C.T. Hsu, J.L. Wu, Hidden digital watermarks in images, IEEE Trans. Image Process. 8 (1) (1999) 58–68.
[6] J.R. Hernandez, M. Amado, F. Perez-Gonzalez, DCT-domain watermarking techniques for still images: detector performance analysis and a new structure, IEEE Trans. Image Process. 9 (1) (2000) 55–68.
[7] W.C. Chu, DCT-based image watermarking using subsampling, IEEE Trans. Multimedia 5 (1) (2003) 34–38.
[8] M. Barni, F. Bartolini, A. Piva, Improved wavelet-based watermarking through pixel-wise masking, Image Process. IEEE Trans. Image Process. 10 (5) (2001) 783–791.
[9] J.J.K. O'Ruanaidh, W.J. Dowling, F.M. Boland, Phase watermarking of digital image, Proceedings of the IEEE International Conference on Image Processing, Vol. 3, Lausanne, Switzerland, 1996, pp. 239–242.
[10] N. Nikolaidis, I. Pitas, Robust image watermarking in the spatial domain, Signal Process. 66 (1998) 385–403.
[11] C.I. Podilchuk, W.J. Zeng, Image-adaptive watermarking using visual models, IEEE J. Sel. Areas Commun. 16 (4) (1998) 525–539.

[12] H.-C. Huang, F.H. Wang, J.S. Pan, Efficient and robust watermarking algorithm with vector quantisation, Electron. Lett. 37 (13) (2001) 826–828.
[13] S. Craver, N. Memon, B.-L. Yeo, M.M. Yeung, Resolving rightful ownerships with invisible watermarking techniques: limitations, attacks, and implications, IEEE J. Sel. Areas Commun. 16 (4) (1998) 573–586.
[14] S. Voloshynovskiy, S. Pereira, T. Pun, J.J. Eggers, J.K. Su, Attacks on digital watermarks: classification, estimation based attacks, and benchmarks, IEEE Commun. Mag. 39 (8) (2001) 118–126.
[15] J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, MI, 1975.
[16] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1992.
[17] M. Gen, R. Cheng, Genetic Algorithms and Engineering Design, Wiley, New York, NY, 1997.
[18] A. Piva, F. Bartolini, M. Barni, Managing copyright in open networks, IEEE Internet Comput. 6 (3) (2002) 18–26.
[19] F.A.P. Petitcolas, Image watermarking—Stirmark, http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark, 2000.
[20] R.C. Gonzalez, R.E. Woods, Digital Image Processing, Addison-Wesley, Reading, MA, 1992.
[21] W.B. Pennebaker, J.L. Mitchell, JPEG: Still Image Data Compression Standard, Van Nostrand Reinhold, New York, 1993.

About the Author—CHIN-SHIUH SHIEH received the B.S. degree in Electronic Engineering from National Taiwan Institute of Technology, Taiwan, in 1989, and the M.S. degree in Electrical Engineering from National Taiwan University, Taiwan, in 1991. His current research interests are in self-learning fuzzy systems using evolutionary techniques, computer networking, and various issues in vector quantization.

About the Author—HSIANG-CHEH HUANG received the B.S., M.S. and Ph.D. degrees in Electronics Engineering from National Chiao Tung University, Taiwan, ROC, in 1995, 1997, and 2001, respectively. Currently, he is a post-doctor researcher in the Department of Electronics Engineering, National Chiao Tung University, Taiwan, ROC. His current research interests include pattern recognition and image processing.

About the Author—FENG-HSING WANG received the B.S. degree in Electronic Engineering from National Kaohsiung University of Applied Sciences, Taiwan, ROC. Currently, he is a Ph.D. candidate in the School of Electrical and Information Engineering, University of South Australia, Adelaide, Australia. His current research interests include computer architecture and image processing.

About the Author—JENG-SHYANG PAN received the B.S. degree in Electronic Engineering from the National Taiwan Institute of Technology, Taiwan, in 1986, the M.S. degree in Communication Engineering from National Chiao Tung University, Taiwan, ROC in 1988, and the Ph.D. degree in Electrical Engineering from the University of Edinburgh, UK, in 1996. Currently, he is a Professor in the Department of Electronic Engineering, National Kaohsiung University of Applied Sciences, Taiwan, ROC. He is also an advisor of postgraduate students both in the Department of Electrical and Information Engineering, University of South Australia and the Department of Automatic Test and Control, Harbin Institute of Technology. He has published more than 35 international journal papers and 70 conference papers. His current research interests include pattern recognition, information security and data mining.