Soft Comput (2009) 13:333–343
DOI 10.1007/s00500-008-0333-9

FOCUS

The optimized copyright protection system with genetic watermarking

Hsiang-Cheh Huang · Chi-Ming Chu · Jeng-Shyang Pan

Published online: 27 May 2008

Abstract Applications for robust watermarking is one of the major branches in digital rights management (DRM) systems and related researches. Based on existing experiences to evaluate the applicability of robust watermarking, it is generally agreed that three parameters or requirements, including the quality of watermarked contents, the survivability of extracted watermark after deliberate or unintentional attacks, and the number of bits embedded, need to be considered. However, performances relating to these three parameters conflict with each other, and the trade off must be searched for. In this paper, we take all the three requirements into consideration, and add the flexibility to meet the specific design in implementation. With the aid of genetic algorithm, we design an applicable system that would obtain the good quality, acceptable survivability, and reasonable capacity after watermarking. Simulation results present the effectiveness in practical implementation and possible application of the proposed algorithm.

1 Introduction

Multimedia contents are easily spread over the Internet. Due to the ease of delivery and modification of digital files, the copyrights might be infringed upon. To deal with this problem, digital rights management (DRM) systems can prevent users from using such contents illegally (Koenen et al. 2004). In DRM systems, *encryption* and *robust watermarking* are two major schemes for applications (Pan et al. 2004a, 2007; Shehab et al. 2008). By using encryption to protect data, the encrypted digital contents look like random and noisy patterns, which will cause the eavesdroppers to suspect the existence of hidden secrets. Furthermore, if one bit is received erroneously during transmission, part or whole of received data would not be decrypted, leading to the uselessness of such contents. Under the umbrella of watermarking researches, the main goal is to cope with the deliberately or unintentionally applied modifications, called attacks, and such a kind of watermarking schemes is regarded as robust watermarking. For robust watermarking, the watermarked contents and their original counterparts look similar, or even identical from subjective point of view. During the transmission, if some parts are received in error (Pan et al. 2004b,c; Chang et al. 2007), or are attacked by some means (Petitcolas 2004; Macq et al. 2004), the received contents can partially be recognized and the copyright can be preserved. Thus, we focus on robust watermarking and propose an applicable solution with optimization techniques in DRM implementation.

In this paper, we use the digital images to represent the multimedia contents. It is generally agreed that for one watermarking algorithm, the watermarked image quality (or *imperceptibility*), the survivability, represented by the correct rate of extracted watermark (or *robustness*), and the number of bits embedded (or *capacity*), are the three most important factors to assess how good the algorithm and implementation are. However, some trade off must be searched for because the three factors conflict with each other. Here we employ genetic algorithm (GA) (Gen and Cheng 1997; Huang et al. 2001) to find an optimized solution that can reach better imperceptibility, more robustness, and reasonable capacity. The scheme can be directly applicable to DRM systems.

H.-C. Huang (✉)
National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC
e-mail: huang.hc@gmail.com

C.-M. Chu · J.-S. Pan
National Kaohsiung University of Applied Sciences, Kaohsiung 807, Taiwan, ROC

This paper is organized as follows. In Sect. 2 we point out the need for optimization in a watermarking system. In Sect. 3 we describe proposed algorithm by modifying and extending previous works. Two detailed case studies are presented in Sect. 4, and simulation results are demonstrated in Sect. 5. Finally, we conclude this paper in Sect. 6.

2 Watermarking requirements

2.1 Elements for robust watermarking

As we stated in Sect. 1, the three major requirements for robust watermarking are imperceptibility, robustness, and capacity. Their interrelationships can be discussed as follows. And we can see why they conflict with each other.

- *Watermark imperceptibility* refers to whether the viewer can perceive the existence of embedded watermark or not from subjective point of view (Huang et al. 2007; Wang et al. 2007). Objectively speaking, it also means the quality of the watermarked image, measured by the error induced between the watermarked and original images due to watermark embedding, and represented by numerical values such as the Peak Signal-to-Noise Ratio (PSNR), with the definition in Eq. (1). Let the original image and the watermarked one be $\boldsymbol{X}$ and $\boldsymbol{X}'$, respectively, with image size of $M \times N$. The PSNR is

$$\mathrm{PSNR} = 10 \times \log_{10} \left(\frac{255^2}{\frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (X(i,j) - X'(i,j))^2} \right). \quad (1)$$

  The larger the PSNR value, the better the outcome. To make the watermarked image imperceptible, the watermark should be hidden into less significant parts, such as the least significant bits in the spatial domain or the high frequency components in the transform domain, in order to make the induced error as small as possible.
- *Watermark robustness* means the capability that the watermarked media can withstand deliberate or unintentional media processing, called attacks, including filtering, resizing, or rotation (Pan et al. 2004a, 2007). There are also benchmarks to perform attacks (Petitcolas 2004). From subjective viewpoint, the extracted watermark needs to be as similar as the embedded one. And objectively speaking, the correlation between the two needs to be measured. People often use the bit correct rate (BCR), with the definition in Eq. (2), to assess the robustness of the watermarking algorithm. Let the embedded and extracted watermarks be $\boldsymbol{W}$ and $\boldsymbol{W}'$, respectively, with size of $M_W \times N_W$. The BCR is

$$\mathrm{BCR} = 1 - \frac{1}{M_W \times N_W} \sum_{i=1}^{M_W} \sum_{j=1}^{N_W} \left[W(i,j) \oplus W'(i,j) \right], \quad (2)$$

  where $\oplus$ denotes the exclusive-or (XOR) operation. The larger the BCR value, the better the result. To make the algorithm robust, the watermark needs to be hidden into more important parts, such as the most significant bits or the low frequency components, in order to resist common attacks.
- *Watermark capacity* attributes to the number of bits embedded into the original media, that is, the size of watermark. We can see that from the results in Huang et al. (2002) and Shieh et al. (2003), embedding more bits into the contents would directly cause the degradation of the quality of watermarked image. On the contrary, embedding too few bits may lead to the result that the extracted watermark may hardly be comprehensible even though the watermarked image quality can be guaranteed. Thus, the watermark capacity needs to be carefully chosen to be meaningful.

2.2 Relationships among watermarking requirements

Based on the descriptions above, we can discuss the relationships as follows:

Imperceptibility refers to the error induced during watermark embedding. The goal for reaching imperceptibility is to increase the PSNR value. And relationships can be listed as follows.

- Less alteration induced into watermarked image is much desired. Based on this standpoint, embedding fewer bits, namely, decreasing the capacity, meets the goal.
- Furthermore, embedding bits into higher frequency coefficient would alter the image as less as possible. However, this would degrade the robustness.

Robustness refers to the capability to resist attacks. The goal for reaching better robustness is to enhance the BCR values. And relationships can be listed as follows:

- In addition to obtaining larger BCR value, extracted watermark needs to be recognizable. Based on this standpoint, the capacity should be more than some threshold.
- Embedding bits into lower frequency coefficient would increase the robustness. However, this would sacrifice the imperceptibility.

Capacity refers to the total number of bits embedded into the original image. The goal for reaching suitable capacity is that the number of embedded bits needs to be more than some threshold. And relationships can be listed as follows:

- Even though increasing the capacity is desired, the appropriate number of capacity should lie above some threshold in order to make the extracted watermark recognizable. On the other hand, embedding too many bits may sacrifice the imperceptibility. If the increase in capacity is obtained, using error control codes (ECC) (Morelos-Zaragoza 2006) for encoding the watermark is able to enhance the robustness. Based on this discussion, the capacity should be carefully chosen.
- Once the capacity is determined, embedding into higher frequency coefficients meets the goal for imperceptibility. On the contrary, embedding into lower frequency coefficients meets the goal for robustness. Hence, some trade off must be searched for, and this is the major contribution of the paper in Shieh et al. (2004).

Because there are conflicts among the three requirements described above, we employ the optimization technique for finding the better outcome. First, the fitness function should be designed. Next, parameters relating to the optimization technique should be carefully chosen.

From the above discussions, we lead to the results that better imperceptibility, more robustness, and the reasonable number of capacity are all required for designing the algorithm. For measuring imperceptibility, we use PSNR of watermarked image to serve as an objective measure. In evaluating robustness, after applying some deliberate attacks (Huang et al. 2007; Chu et al. 2008), we calculate the BCR. For measuring capacity, we count the average number of bits, C, embedded into one 8×8 block, since we are going to embed the watermark with discrete cosine transform (DCT). The fitness function for training at iteration i can be defined:

$$f_i = \mathrm{PSNR}_i + \lambda_1 \cdot \frac{1}{n} \sum_{k=1}^{n} \mathrm{BCR}_{k,i} + \lambda_2 \cdot C_i. \tag{3}$$

The first term, PSNR_i, denotes the imperceptibility. In the second term, because we expect to cope with n different attacks, we calculate the robustness after certain attacks, $\mathrm{BCR}_{k,i}$, respectively, and the average of these BCR values is served as the robustness measure. In the third term, C_i implies the capacity. Because PSNR values are usually more than 30 dB, the BCR values lie between 0 and 1, and the capacity can be set to 1–4 bits per block after considering practical situations, and the average capacity must lie between 1 and 4 bit/block, we find that values corresponding to the three components lie into various ranges because of their inherent characteristics. Thus, we introduce two weighting factors, λ_1 and λ_2, into the fitness function. The main reason is to balance the effects and contributions from these three factors. And the goal of our optimization algorithm is to find the maximum value in Eq. (3).

Figure 1 is the conceptual illustration of robust watermarking with GA optimization. Original image $\boldsymbol{X}$ is the input. At the beginning, we set the number of iteration for training. In the training process, for every population in GA, the number of bits for embedding into every 8×8 block of the original image, one after another, is decided first, and the watermark capacity is regarded as one of the three parts in the fitness function. Next, the watermark is embedded into the original images, namely, the populations in GA, and the PSNR values of watermarked images are obtained. Finally, we apply n different attacks, for instance, JPEG compression for every population, and we try to extract the embedded watermarks from every attacked image. The BCR values between the embedded and extracted watermarks under different attacks are obtained. After calculating the fitness function, we proceed with this process in the next iteration until meeting the terminating condition. Finally, the optimized, watermarked image, $\boldsymbol{Y}$, and associated secret key, key_1, are delivered to the reception side.

3 Proposed algorithm

We employ genetic algorithm (GA) for optimizing the three requirements above. GA is constituted of three major steps: *selection*, *crossover*, and *mutation* (Gen and Cheng 1997; Huang et al. 2001). Based on the fitness function in Eq. (3) and the flow graph in Fig. 1, we propose an integration of our watermarking scheme with GA procedures.

3.1 Preprocessing in GA

GA is a process to emulate the natural selection. We need to have populations for training in GA to perform the three steps. The number of populations is generally chosen to be an even number to ease the operation of the crossover step.

Every population is composed of chromosomes. The chromosome is a binary string, and the binary representation denotes the places for watermark embedding in one block. It has variable length, which depends on the watermark capacity decided. By concatenating the binary string of every block in the image, we obtain one population in GA. Because we perform 8×8 DCT for watermark embedding (Shieh et al. 2004), we have 64 DCT coefficients, ranging between 0 and 63, and these coefficients are represented with a 6-bit string, in one block. If the size of original image is $W \times H$, and let the maximal capacity be $C_{\max}$ bit/block, the length of the

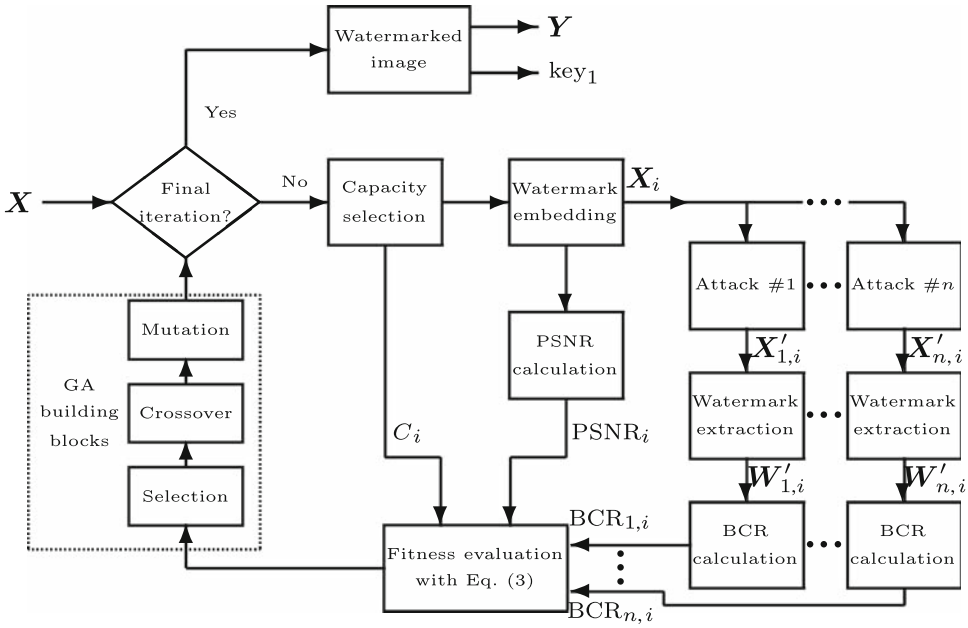

Fig. 1 Building blocks of watermarking with GA-based optimization schemes

population is $\frac{W}{8} \times \frac{H}{8} \times 6 \times C_{\max}$ bits. For certain blocks that are embedded fewer bits than $C_{\max}$, considering the practicability in implementation, the remaining parts of the chromosome are replaced by consecutive bits of 0's. The size of original image is 512×512 and that of the binary watermark is 128×128 in this paper. Hence, $C_{\max}$ is set to be $\frac{128 \times 128}{\frac{512}{8} \times \frac{512}{8}} = 4$ bit/block.

3.2 Deciding the capacity in one iteration

After considering practical implementations in GA in Sect. 3.1, $C_{\max} = 4$. And the capacity for every 8×8 block is variable, ranging between 1 to 4 bits per block. After calculation of 8×8 DCT, 64 DCT coefficients can be produced, and the range lies between 0 and 63, where 0 denotes the DC coefficient, i denotes the ith AC coefficients, $1 \leq i \leq 63$ for watermark embedding.

For clearly explaining our implementation, we describe an instance as follows. If the capacity $C = 2$ for a certain block, and suppose that the 5th and 17th coefficients are selected, then the chromosome in this block is represented by a 24-bit string `000101 010001 000000 000000`, where the first 12 bits denote the position, and the final 12 bits represent the remaining part, which are intentionally inserted to ease the implementation. At the first training iteration, all the AC coefficients are randomly selected for watermark embedding.

3.3 Embedding watermark with DCT

We modify conventional schemes (Shieh et al. 2004) for DCT-based watermarking to embed the binary watermark.

Step 1. *Performing the DCT of original image*: 8×8 DCT is performed on the entire 512×512 image. For one block, it leads to one DC coefficient, presented by coefficient 0, and 63 AC coefficients, presented by coefficients 1–63.

Step 2. *Determining the capacities and positions for embedding*: The goal of our algorithm is to search for the proper positions for embedding based on decided capacity, leading to a trade off among the three requirements and suitable positions.

Step 3. *Obtaining the threshold for embedding*: The average values of DC and other 63 AC coefficients among the $\frac{512}{8} \times \frac{512}{8}$, or 4096 blocks are served as the thresholds for watermark embedding.

Step 4. *Embedding the watermark bits*: The thresholds in Step 3 are represented by a vector $\boldsymbol{a} = [a_0, a_1, \cdots, a_{63}]$, where $\boldsymbol{a}$ denotes the average value. The DC coefficient is prohibited for embedding. And we use the vector $\boldsymbol{r} = \left[\frac{a_0}{a_1}, \frac{a_0}{a_2}, \cdots, \frac{a_0}{a_{63}}\right]$ to serve as the reference for modifying the AC coefficients, where $\boldsymbol{r}$ denotes the ratio between DC and AC coefficients. Embedding of watermark meets one of the two situations below:

- If bit 0 is embedded, the AC coefficient in selected position is modified. If it is larger than the reference value in $\boldsymbol{r}$, it is decreased to be smaller than the corresponding element in $\boldsymbol{r}$ by a parameter δ. If not, the value is kept unchanged.
- If bit 1 is embedded, the coefficient is modified to be in contrary with the previous condition.

Step 5. *Performing the inverse DCT of modified coefficients*: Inverse DCT is calculated to obtain the watermarked image. The corresponding positions for watermark embedding are also recorded, and the PSNR of the watermarked image can be obtained.

3.4 Choosing proper attacks

For verifying robust watermarking, applying attacks to watermarked image is necessary. However, attacks need to be properly selected such that the attacked images still retain its meaningfulness and commercial value. For instance, image cropping attack is unsuitable since too much information would be discarded, and subjective image quality is degraded. In this paper, we choose three kinds of attacks (Petitcolas 2004), namely, JPEG compression with different quality factors (QF), low-pass filtering (LPF), and median filtering (MF), to perform the attacks. Attacked images look similar to their original counterpart after applying these properly selected attacks. The BCR values after experiencing these attacks are calculated, and the average of these values are included into the fitness function.

3.5 Extraction of watermark

Let the watermarked image $\boldsymbol{Y}$ in Fig. 1, after applying attack, be denoted by $\boldsymbol{Z}$. We calculate the DCT of the attacked image $\boldsymbol{Z}$, and generate the new reference value $\boldsymbol{r}'$ by following Step 4 in Sect. 3.3. The extracted watermark bit is determined by one of the two situations below:

- If the selected coefficient divided by the average of DC value in $\boldsymbol{Z}$ is smaller than its corresponding coefficient in $\boldsymbol{r}'$, we decide the extracted watermark bit to be 0;
- If the selected coefficient divided by the average of DC value in $\boldsymbol{Z}$ is larger than its corresponding coefficient in $\boldsymbol{r}'$, we decide the extracted watermark bit to be 1.

3.6 Evaluating fitness

We gather the average capacity C in Sect. 3.2, calculate the PSNR in Step 5 of Sect. 3.3, and obtain the average of BCR in Sects. 3.4 and 3.5, and then combine them altogether to calculate the fitness value with Eq. (3). Every population corresponds to one fitness value in the training iteration.

3.7 GA procedures

The generated binary strings are ready for GA procedures. In this paper, the number of populations is 10. The selection rate is 0.5, meaning that only the 5 populations with higher fitness values are kept for the next iteration, and the remaining 5 are produced by the crossover operation. The mutation rate is 0.1, meaning that 10% of all the bits are randomly selected and intentionally flipped. The main theme for GA is to search for the proper coefficient positions for watermark embedding, leading to the associated secret key, key_1. The secret key can be delivered with the scheme in Piva et al. (2002). Weighting factor λ_1 is set to be between 0 and 200, and its counterpart λ_2 is set to range between 0 and 50 in the GA training process. The parameter for altering the selected DCT coefficients in Step 4 in Sect. 3.3, δ, is fixed to 5. The major reason for choosing these values is to balance the effects from the three requirements, because we would like to have equal contribution from the three requirements to some extent. Based on this setting, fitness value from the three weighted requirements can lie among the following ranges:

- PSNR part: from 30 to 55, observed from simulation results;
- BCR part multiplied by weighting factors: from 0 to 200, because BCR $\in [0, 1]$ and $\lambda_1 \in [0, 200]$;
- capacity part multiplied by weighting factors: from 0 to 200, because $C \in [1, 4]$ and $\lambda_2 \in [0, 50]$.

Results with these different combinations of weighting factors are verified in Sect. 5.

3.8 The stopping condition

Once the number of training iterations in GA is reached, the optimization process is stopped. The population with the largest fitness value in the final iteration is the optimized watermarked image. Corresponding secret key with this image is also delivered to the receiver (Piva et al. 2002).

4 Case studies in optimized embedding

4.1 Fixed embedding capacity

We choose the test image `bridge` with the picture size of 512×512, illustrated in Fig. 2a. The binary watermark with the size of 128×128 is prepared, shown in Fig. 2b. In Fig. 2, the width and height between the two images are carefully chosen to be 4:1. And we will compare with the results shown in Shieh et al. (2004). Because in Shieh et al. (2004), authors used normalized correlation (NC) to represent the watermark robustness, and we use BCR here, hence we show extracted watermarks for subjective evaluation in Fig. 3. Both Shieh et al. (2004) and our paper denote imperceptibility by using PSNR, we make comparisons in Table 1. Based on the settings by including JPEG quantization tables with watermarking in Shieh et al. (2004) for reference, we obtain reasonable results with the algorithm proposed in this paper.

We choose the JPEG attack with quality factor QF $= 80$ to validate the proposed algorithm. The weighting factors, λ_1 and λ_2 in Eq. (3), are set to $\lambda_1 = 50$ and $\lambda_2 = 0$, respectively. The main reason for setting $\lambda_2 = 0$ is that we can manually adjust the capacity to see the performances between imperceptibility and robustness. First, we compare the extracted

(a) Original bridge image (b) Watermark

Fig. 2 **a** The original bridge image with size 512×512. **b** The binary watermark with size 128 × 128

(a) 128 × 128 bits BCR = 0.9232

(b) 128 × 96 bits BCR = 0.9224

(c) 128 × 64 bits BCR = 0.9430

(d) 128 × 32 bits BCR = 0.9739

Fig. 3 **a–d** Extracted watermarks with capacities 4, 3, 2, and 1 bit/block, respectively

watermarks in Fig. 3. Figure 3a shows the one with capacity of 4 bit/block. Figure 3b–d illustrates those with capacity of 3, 2, 1 bit/block, respectively, leading to the watermark size of 128 × 96, 128 × 64, and 128 × 32. Figure 3a can be clearly perceived, and the BCR value is high. We can also see that in Fig. 3b, the capacity is 3 bit/block, and only the upper three quarters can be recognized. The bottom quarter is intentionally set to bit 0 for comparison. Figure 3c and d also have similar phenomena. For embedding 3 or 4 bit/block, similar BCR values can be obtained. When decreasing the capacity to 2 or 1 bit/block, the BCR values grow. However, even though the BCR values are high enough, decreasing the capacity leads to be less meaningful in the extracted watermarks.

Table 1 Comparisons of capacity (in bit/block) and imperceptibility, represented by PSNR (in dB), between our algorithm and existing one

Scheme	Capacity (bit/block)	Imperceptibility (dB)
Existing (Shieh et al. 2004)	4	34.79
Proposed	4	33.95
	3	35.24
	2	37.57
	1	40.50

Table 1 makes comparison between the scheme in this paper and that in Shieh et al. (2004). We can see that comparable results can be obtained. When we lower the embedded capacity, the PSNR values get higher. This is because less DCT coefficients get modified, and it proves our discussions in Sect. 2.

In Fig. 4, we present the number of embedded positions that is associated with the results in Fig. 2. Due to the JPEG compression attack that tends to discard the higher frequency coefficients, lower to middle frequency coefficients, namely, AC_2 and AC_{18}, are mostly embedded. Moreover, with the values indicated on the vertical axis, we can see that the total number of embedded bits decreases from Fig. 4a to d.

From the data in Figs. 3, 4 and Table 1 above, we can find out that the three requirements have their own characteristics inherently, and they influence on another. By taking the watermark capacity into account, we have more flexibility in the design of algorithm.

4.2 Variable embedding capacity

Considering the fitness function in Eq. (3), we choose $\lambda_1 = 50$ and $\lambda_2 = 15$ for the detailed case study among the three requirements. The main reason for choosing such values is to balance the contributions from all the three requirements. Regarding to the attacking schemes, the JPEG compression with QF = 80 is chosen for verifying our algorithm in this case study. Moreover, attacking schemes with the 3 × 3 low-pass filtering (LPF), and the 3 × 3 median filtering (MF), are also examined, and results are depicted in Sect. 5. In GA, we choose 20 populations with selection rate of 0.5 and mutation rate of 0.1 for optimization.

After training for 100 iterations under the preliminary for better imperceptibility and better robustness under the JPEG attack, we obtain the optimized output with PSNR = 45.91 dB in Fig. 5, and we can hardly differentiate the differences between the original image and the watermarked one subjectively. Regarding to the watermark robustness in addition to the JPEG attack, we also employ

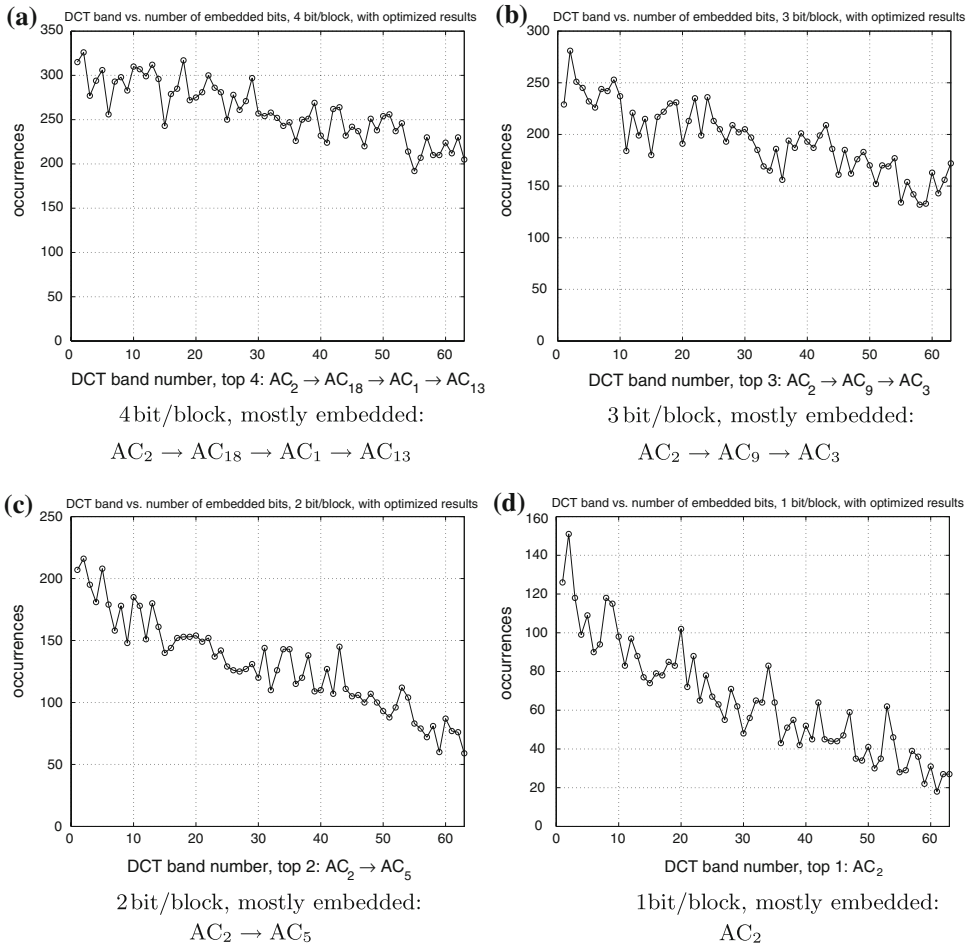

Fig. 4 The histogram between embedding DCT coefficients and the number of bits embedded. **a**–**d** Embedding coefficients with capacity 4, 3, 2, and 1 bit/block, respectively

two other attacking schemes altogether on the watermarked image to see whether our attack can survive after other attacks or not. For making comparisons conveniently, we put the embedded watermark again in Fig. 6a. And we can see that BCR = 0.9603 for JPEG attack in Fig. 6b, BCR = 0.7128 for LPF attack in Fig. 6c, and BCR = 0.7469 for MF attack in Fig. 6d.

It is easily comprehended that in this case, we can obtain better imperceptibility and it can successfully resist the JPEG attack. However, it cannot survive under other attacks, such LPF and MF due to the fact that the BCR after JPEG attack is included into the fitness function in GA in Eq. (3), but others are not. With this observation, when coping with several different attacks, all the extracted BCR values need to be integrated into the fitness function. For clearly representing the effects under various capacities, in the extracted watermark, bit 0 and bit 1 are denoted by black and white pixels, respectively, while those in the remaining parts are intentionally denoted by grey pixels. This phenomena can be seen from Fig. 6b to d. On the one hand, the watermark extracted from JPEG-attacked image, shown in Fig. 6b, can be clearly

Fig. 5 Watermarked output, PSNR = 45.91 dB

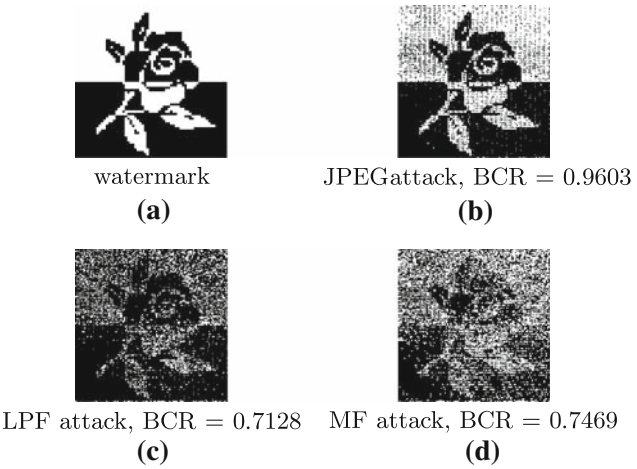

Fig. 6 Comparisons of embedded watermark and extracted ones after different attacks. From subjective viewpoint, (**c**) and (**d**) do not survive well under LPF and MF attacks. **a** Embedded watermark containing $128 \times 128 = 16384$ bits. **b** Extracted from JPEG attack. **c** Extracted from LPF attack. **d** Extracted from MF attack

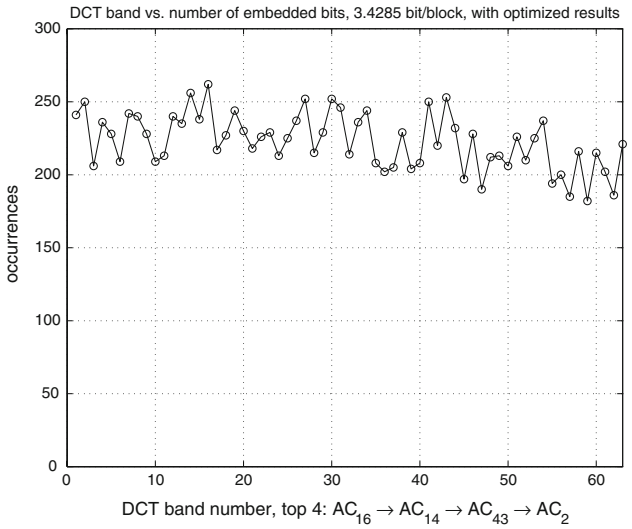

Fig. 7 The histogram between embedding DCT coefficients and the number of bits embedded. AC_{16} and AC_{14} are the top-two coefficients for embedding, and a total of 14043 bits, or 3.4285 bit/block, are embedded

recognizable, and the BCR value is very high. On the other hand, the watermark extracted from LPF- and MF-attacked image, illustrated in Fig. 6c and d, respectively, can hardly be recognized, and also the BCR values are not high enough. This result is reasonable because we focus on the JPEG attack, and Fig. 6b verifies this phenomenon.

Next, we check the histogram for embedding coefficients in Fig. 7, and we find that a total of 14043 bits are embedded. For measuring imperceptibility, objective value is acceptable and most parts for watermarking are invisible. For evaluating robustness, the BCR value is high enough, while the extracted watermark is easily recognized under JPEG attack. For embedding positions, the 16th and 14th coefficients (or AC_{16} and AC_{14}, respectively) are embedded mostly, which follows the concept of embedding into 'middle frequency bands' proposed in literature (Huang et al. 2007; Shieh et al. 2004).

5 Simulation results

5.1 Selection of weighting factors

Besides the case study depicted in Sect. 4, we provide more results with our experiments as follows. Table 2 demonstrates the performances among imperceptibility, robustness, and capacity, and the two DCT coefficients that are mostly embedded, under a variety of weighting factors. These results are obtained after 100 training iterations in GA, with selection rate of 0.5 and mutation rate of 0.1. We perform lots of experiments based on the preliminary conditions that $\lambda_1 \in [0, 200]$ and $\lambda_2 \in [0, 50]$, and we present results with 15 of all the experiments in Table 2. These experiments can be classified into three categories:

1. fixing the robustness factor λ_1 to 50, and varying the capacity factor λ_2 from 10 to 30 with a stepsize of 5.
2. fixing the robustness factor λ_1 to 100, and varying the capacity factor λ_2 from 10 to 30 with a stepsize of 5.
3. fixing the robustness factor λ_1 to 150, and varying the capacity factor λ_2 from 10 to 30 with a stepsize of 5.

From the numerical values in Table 2, and the subjective evaluation from Fig. 9, we observe that by increasing the weighting factor of capacity, we can see that the average capacity gets increased, while the BCR values gets somewhat reduced. PSNR values fluctuate a bit, but comparing to the original image, they remain objectively unnoticed in the watermarked parts. It is because of the embedding position selected after GA optimization. According to the data presented in Fig. 8, the best embedding bands also lie in low to middle frequency bands.

Furthermore, we can easily see that the BCR values after LPF attack are much lower than their counterparts after MF and JPEG attacks. To alleviate this problem, the weighting factor associated with robustness, λ_1, should be increased to enhance the contribution from the robustness in the fitness function. Comparing the three sets of data with $(\lambda_1, \lambda_2) = (50, 15)$, $(100, 15)$, and $(150, 15)$ for instance, we observe that by simply increasing the value of λ_1, both the resulting PSNR and capacity get decreased. Figure 9 also demonstrate this observation from subjective point of view.

Table 2 Comparisons of imperceptibility (in dB), robustness, and capacity (in bit/block) with different weighting factors

PSNR (dB)	BCR (JPG)	BCR (LPF)	BCR (MF)	Capacity (bit/block)	The two best bands	Factors	
						λ_1	λ_2
45.46	0.9161	0.7897	0.8484	3.3364	$AC_4 \rightarrow AC_2$	50	10
44.34	0.9013	0.7199	0.8103	3.7200	$AC_4 \rightarrow AC_2$	50	15
43.23	0.8838	0.6698	0.7695	3.9302	$AC_1 \rightarrow AC_{12}$	50	20
43.01	0.8820	0.6611	0.7599	3.9819	$AC_7 \rightarrow AC_4$	50	25
42.95	0.8824	0.6598	0.7588	3.9941	$AC_1 \rightarrow AC_2$	50	30
41.58	0.9349	0.8799	0.9011	2.8264	$AC_4 \rightarrow AC_1$	100	10
40.98	0.9402	0.8268	0.8959	3.1804	$AC_1 \rightarrow AC_4$	100	15
40.94	0.9308	0.7875	0.8681	3.5144	$AC_4 \rightarrow AC_1$	100	20
40.33	0.9118	0.7374	0.8375	3.7473	$AC_4 \rightarrow AC_1$	100	25
39.92	0.9058	0.7054	0.8167	3.8752	$AC_1 \rightarrow AC_4$	100	30
40.78	0.9421	0.8762	0.9181	2.7021	$AC_4 \rightarrow AC_2$	150	10
39.94	0.9530	0.8871	0.9215	2.8979	$AC_4 \rightarrow AC_2$	150	15
39.90	0.9534	0.8683	0.9131	3.1506	$AC_4 \rightarrow AC_1$	150	20
39.55	0.9420	0.8304	0.8994	3.3611	$AC_1 \rightarrow AC_4$	150	25
38.99	0.9332	0.7838	0.8702	3.5781	$AC_4 \rightarrow AC_1$	150	30

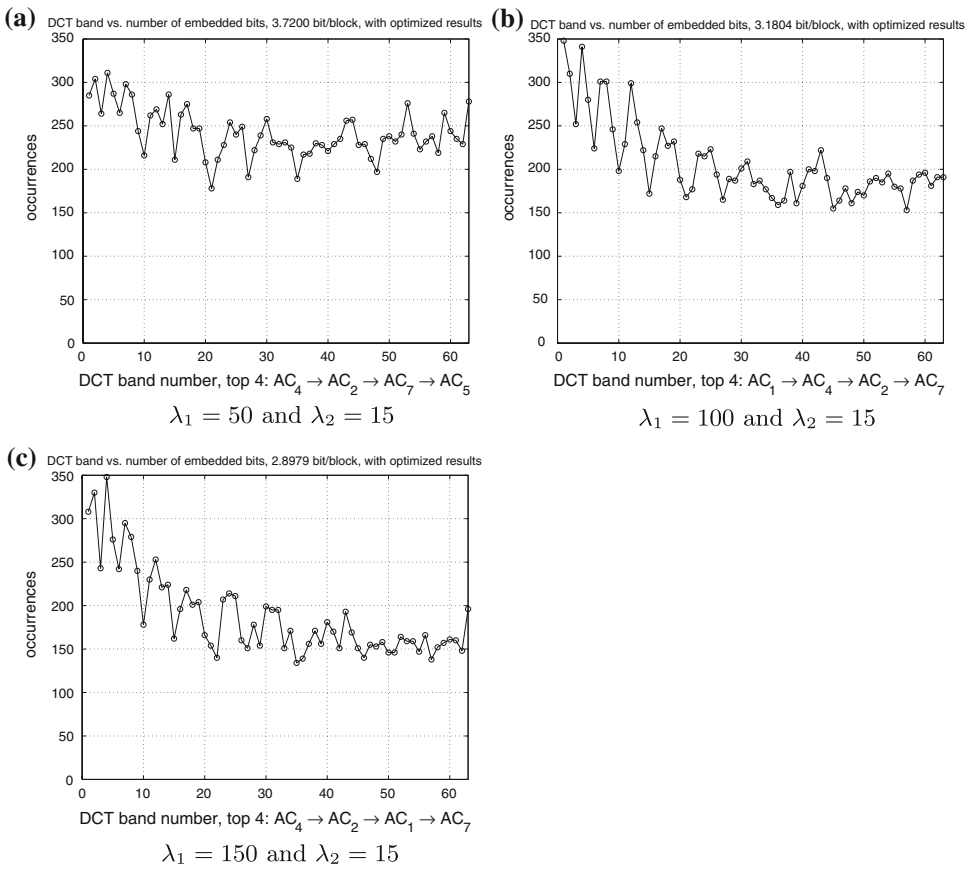

Fig. 8 Comparisons of the histograms of embedding coefficients with different weighting factors in Eq. (3)

Summing up, the weighting factors need to be carefully chosen based on twofold. The first is that contributions from the different requirements are supposed to have nearly equal contribution. The second is that both the watermarked image and extracted watermark need to be recognized from subjective point of view.

JPEG attack, BCR = 0.9013 | LPFattack, BCR = 0.7199 | MF attack, BCR = 0.8103

(a) Results with $\lambda_1 = 50$ and $\lambda_2 = 15$

JPEG attack, BCR = 0.9402 | LPFattack, BCR = 0.8268 | MF attack, BCR = 0.8959

(b) Results with $\lambda_1 = 100$ and $\lambda_2 = 15$

JPEG attack, BCR = 0.9530 | LPFattack, BCR = 0.8268 | MF attack, BCR = 0.8959

(c) Results with $\lambda_1 = 150$ and $\lambda_2 = 15$

Fig. 9 Comparisons of the extracted watermarks and BCR values with different weighting factors in Eq. (3)

5.2 Combination of various attacks

Based on the building blocks in Fig. 1, algorithm designer can choose different attacks for making optimization. We show the combination of various attacks with GA in Table 3 as follows.

In Table 3, we list all the seven combinations from the three independent attacks with the weighting factors of $\lambda_1 = 50$ and $\lambda_2 = 10$. The BCR values, shown in italics, are not optimized based on the type of attacks. For instance, for the results with JPEG-type attack in the first row, numerical values for $BCR_{(LPF)}$ and $BCR_{(MF)}$ are shown in italics, because only $BCR_{(JPG)}$ is optimized. Because we embed the watermark into DCT coefficients, it seems that our algorithm tends to resist JPEG attack inherently. Therefore, in the first three rows, we can see that BCR values after JPEG attacks are high enough, and we need to take the LPF or MF attack into optimization to obtain the improved BCR values. In the fourth to sixth rows, we can see that only the BCR values with selected attacks perform better. In the last row, because we calculate the average BCR value in the fitness function in Eq. (3), if we choose all the three attacks altogether, $BCR_{(JPG)}$ tends to perform better inherently. Therefore, the two remaining BCR values may get decreased, and we can see that the numerical results present this phenomenon.

Summing up, our algorithm tends to resist JPEG attack based on its characteristics. And we may suggest to ignore the JPEG attack during the optimization process. By choosing the combination of LPF and MF attacks into the fitness function, acceptable results can be reached.

6 Conclusions

In this paper, we discussed about the optimization of robust watermarking with genetic algorithms. By finding trade-offs among robustness, capacity, and imperceptibility, we design a practical fitness function for optimization. We observe that the three requirements conflict with one another, thus, by applying GA, we can obtain the optimized outcome. Simulation results depict the improvements of our algorithm, hence the implementation of copyright protection system, and it is directly extendable to cope with a variety of attacks in the benchmarks. In addition, the weighting factors in the fitness function play an important role in the design of algorithm.

Table 3 Comparisons of imperceptibility (in dB), robustness, and capacity (in bit/block) with a different combination of attacks

Attack	PSNR (dB)	BCR (JPG)	BCR (LPF)	BCR (MF)	Capacity (bit/block)	The two best bands
J	45.91	0.9603	*0.7128*	*0.7469*	3.4285	$AC_{16} \rightarrow AC_{14}$
L	46.44	*0.9194*	0.8380	*0.7963*	3.2273	$AC_1 \rightarrow AC_4$
M	43.45	*0.9058*	*0.7947*	0.8486	3.3423	$AC_1 \rightarrow AC_4$
J & L	46.45	0.9262	0.8395	*0.7955*	3.1863	$AC_4 \rightarrow AC_1$
J & M	43.56	0.9107	*0.7500*	0.8718	3.3145	$AC_1 \rightarrow AC_4$
L & M	46.21	*0.9058*	0.7947	0.8486	3.3081	$AC_4 \rightarrow AC_2$
J & L & M	45.46	0.9196	0.7897	0.8484	3.3364	$AC_4 \rightarrow AC_2$

Weighting factors are set to be $(\lambda_1, \lambda_2) = (50, 10)$. To simplify the representation, we use the abbreviations of J, L, and M to represent JPEG, LPF, and MF, respectively. The italicized values are the BCR values corresponding to attacks that are not trained with GA. Therefore, these values show inferior results since they are not optimized

Properly selected weighting factors can lead to better results in overall performance. Other schemes, such as employing ECC into the watermark, can be considered to be integrated into our implementation in the future.

References

Koenen RH, Lacy J, Mackay M, Mitchell S (2004) The long march to interoperable digital rights management. Proc IEEE 92:883–897

Pan JS, Huang H-C, Jain LC (eds) (2004a) Intelligent watermarking techniques. World Scientific Publishing Company, Singapore, pp 3–38

Pan JS, Huang H-C, Jain LC, Fang WC (eds) (2007) Intelligent multimedia data hiding. Springer, Berlin

Shehab M, Bertino E, Ghafoor A (2008) Watermarking relational databases using optimization-based techniques. IEEE Trans Knowl Data Eng 20:116–129

Pan JS, Hsin YC, Huang H-C, Huang KC (2004b) Robust image watermarking based on multiple description vector quantisation. Electron Lett 40:1409–1410

Pan JS, Sung MT, Huang H-C, Liao BY (2004c) Robust VQ-based digital watermarking for the memoryless binary symmetric channel. IEICE Trans Fundam Electron Commun Comput Sci E-87: 1839–1841

Chang F-C, Huang H-C, Hang H-M (2007) Layered access control schemes on watermarked scalable media. J VLSI Signal Process Syst Signal Image Video Technol 49:443–455

Petitcolas FAP (2004) Stirmark benchmark 4.0. Available from: <http://www.petitcolas.net/fabien/watermarking/stirmark/>

Macq B, Dittmann J, Delp EJ (2004) Benchmarking of image watermarking algorithms for digital rights management. Proc IEEE 92:971–984

Gen M, Cheng R (1997) Genetic algorithms and engineering design. Wiley, New York, NY

Huang H-C, Pan JS, Lu ZM, Sun SH, Hang H-M (2001) Vector quantization based on genetic simulated annealing. Signal Processing 87:1513–1523

Huang H-C, Pan JS, Huang YH, Wang FH, Huang K-C (2007) Progressive watermarking techniques using genetic algorithms. Circuits Syst Signal Process 26:671–687

Wang S, Zheng D, Zhao J, Tam WJ, Speranza F (2007) An image quality evaluation method based on digital watermarking. IEEE Trans Circuits Syst Video Technol 17:98–105

Huang H-C, Wang FH, Pan JS (2002) A VQ-based robust multiwatermarking algorithm. IEICE Trans Fundam Electron Commun Comput Sci E-85:1719–1726

Shieh CS, Huang H-C, Wang FH, Pan JS (2003) An embedding algorithm for multiple watermarks. J Inf Sci Eng 19:381–395

Morelos-Zaragoza RH (2006) The art of error correcting coding, 2nd edn. Wiley, New York, NY

Shieh CS, Huang H-C, Wang FH, Pan JS (2004) Genetic watermarking based on transform domain techniques. Patt Recog 37:555–565

Chu SC, Huang H-C, Shi Y, Wu SY, Shieh CS (2008) Genetic watermarking for zerotree-based applications. Circuits Syst Signal Process 27:171–182

Piva A, Bartolini F, Barni M (2002) Managing copyright in open networks. IEEE Internet Comput 6:18–26