Chapter 5
Fast Watermark Verification by Progressive Image Matching

5.1 Overview of Proposed Method

In this chapter, the study on fast watermark verification focuses on Joint Photographic Experts Group (JPEG) images only. Thus, in Section 5.1.1, some properties of JPEG images are introduced. And a review of employed invisible watermarking technique for compressed images is described in Section 5.1.2. And the proposed idea of fast watermark verification is introduced in Section 5.1.3. Also, in Section 5.2, the proposed method for fast watermark verification by progressive image matching is described. In Section 5.3, some experimental results are illustrated. Finally, in Section 5.4, some discussions and a summary are made.

5.1.1 Properties of JPEG Images

Figures 5.1 and 5.2 show the encoding and decoding frameworks of the JPEG. In the compression process, the source image is first divided into non-overlapping 8×8 blocks. Each 8×8 block is independently taken as an input to the forward DCT (FDCT), and is transformed into the frequency domain. The outputs of the FDCT are 64 integer values, called DCT coefficients. The coefficient with zero frequency is called the DC coefficient, while the other 63 coefficients are called the AC coefficients. These 64 DCT coefficients are quantized using a quantization table in

order to discard information that is not visually significant. After the quantization, the DC coefficient of each 8×8 block is treated separately from the 63 AC coefficients. The DC coefficients are encoded by predictive coding, and the 63 AC coefficients are reordered in a zigzag order for better performance of entropy coding. There are two entropy coding methods, which are the Huffman coding method and the arithmetic coding method, specified by the JPEG standard.

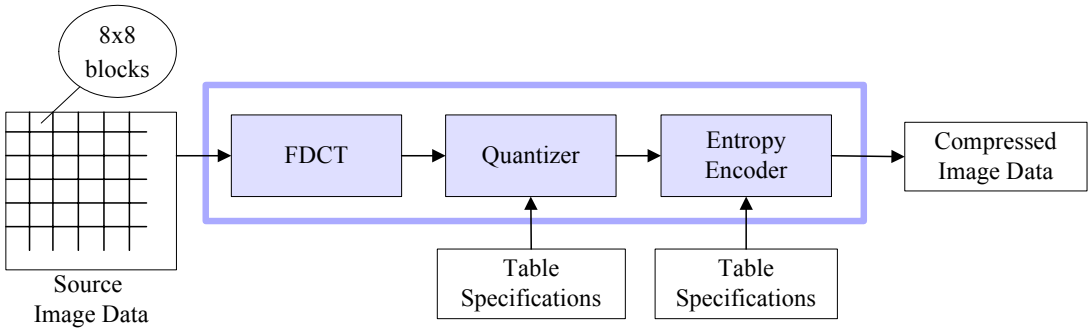


Figure 5.1 DCT-based encoder compression process.

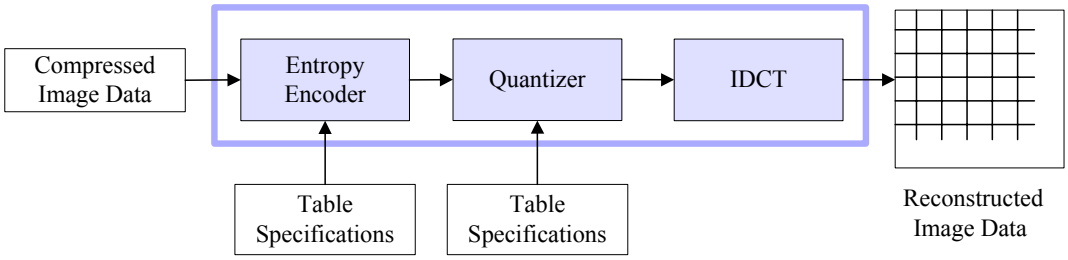


Figure 5.2 DCT-based decoder decompression process.

5.1.2 Review of Employed Invisible Watermarking Technique for JPEG Images

First of all, two DCT coefficients are selected to embed an irremovable invisible watermark in the employed method. The two DCT coefficients are chosen according to the quantization values associated with them in the standard quantization table, and the quantization values should be equal. Consequently, the magnitude relation between the two coefficients will be kept after the quantization step. Let (x, y) denote the location of a coefficient in an 8×8 block. According to Table 5.1, the coefficients located at (1, 4) and (2, 3) or (0, 5) and (3, 2) are good candidates for this purpose.

Table 5.1 Standard quantization table in JPEG compression standard (luminance component).

(x,y)	0	1	2	3	4	5	6	7
0	16	11	10	16	24	40	51	61
1	12	12	14	19	26	58	60	55
2	14	13	16	24	40	57	69	56
3	14	17	22	29	51	87	80	62
4	18	22	37	56	68	109	103	77
5	24	35	55	64	81	104	113	92
6	49	64	78	87	103	121	120	101
7	72	92	95	98	112	100	103	99

In the watermark embedding process of the employed method, we use an 8×8 image block to embed a pixel of the watermark. Therefore, we divide the *cover image* of size $M \times N$ into non-overlapping 8×8 image blocks and resize the binary watermark

image W to be $\left\lfloor \frac{M}{8} \right\rfloor \times \left\lfloor \frac{N}{8} \right\rfloor$ if the original size of the watermark is larger than $\left\lfloor \frac{M}{8} \right\rfloor \times \left\lfloor \frac{N}{8} \right\rfloor$. For each image block, the RGB color model are transformed into the YC_bC_r color model and the FDCT is performed on the Y channel. The magnitude relation of the two chosen DCT coefficients, called S_1 and S_2, is adjusted according to the corresponding watermark pixel. If the watermark pixel color is black, then we let S_1 to be larger than S_2. If S_2 is larger than S_1 originally, we swap two coefficients. If S_1 is larger than S_2 originally, two coefficients are kept unchanged. If the watermark pixel color is white, we let the two DCT coefficients be adjusted in the contrary way. After the watermark signal embedding process, these 64 DCT coefficients are transformed back into the spatial domain by the IDCT, and then the YC_bC_r color values are transformed back into the RGB color model. When all blocks are processed, a *stego-image* is obtained. A flowchart of the employed invisible watermark embedding process is shown in Figure 5.3.

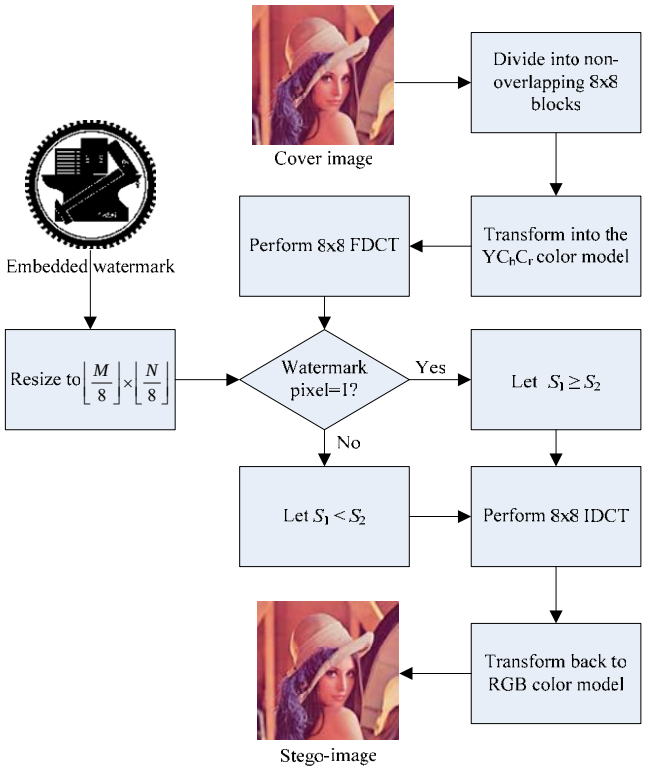


Figure 5.3 A flowchart of employed invisible watermark embedding process.

In the watermark extraction process, we extract the embedded invisible watermark signal by detecting the magnitude relation of the two chosen DCT coefficients. First, the stego-image is divided again into non-overlapping 8×8 image blocks. For each image block, the RGB color values are transformed into the YC_bC_r color values and the FDCT is performed on the resulting Y channel. After performing the FDCT, the magnitude relation of the two chosen DCT coefficients is detected. If S_1 is larger than S_2, the extracted watermark pixel color is decided to be black; otherwise, white. After all blocks are detected, the binary pixel values of the watermark are extracted and the binary embedded watermark reconstructed. A flowchart of the employed invisible watermark embedding process is shown in Figure 5.4.

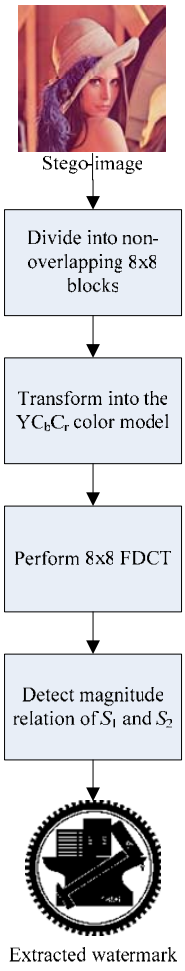


Figure 5.4 A flowchart of employed invisible watermark extraction process.

5.1.3 Proposed Idea of Fast Watermark Verification

The employed method described in the last section was implemented in the software InfoProtector to embed invisible watermarks into JPEG images. Relatively, the extraction process of the employed method is a bit of slow, so we propose the method for fast watermark verification by progressive image matching.

In certain conditions, we can judge a suspected image as a non-infringed one before all pixels of the embedded watermark are extracted. For example, we can reject the possibility of infringement of a suspected image immediately if a low similarity between partial pixels of the embedded watermark and the corresponding pixels of the declared image owner's watermark is found. Only when the similarity is high enough is the suspected image likely to be a copyright-infringed one. And then we will extract other pixels of a larger amount to calculate a new similarity for further judgment.

5.2 Proposed Method for Fast Watermark Verification

In Section 5.2.1, we described the process of fast watermark verification by progressive image matching, which is a *progressive* watermark extraction technique. And a detailed algorithm of the proposed method is described in Section 5.2.2.

5.2.1 Process of Fast Watermark Verification by Progressive Watermark Extraction

In the previous chapters, an *entire* embedded watermark is extracted to find out whether the suspected image is a copyright-infringed one or not. But in certain

conditions, it is not needed to extract the *entire* watermark to judge the suspected image as a non-infringed one; part of it is enough sometimes if progressive watermark image extraction and matching is conducted. That is, if the similarity between the partially extracted pixels and the corresponding pixels of the declared image owner's watermark is not high enough, we can infer immediately that the extracted watermark is not the watermark of the declared image owner's. So we do not have to continue the extraction work, which saves a lot of time. In the proposed method based on such an idea, only JPEG images are applicable.

First of all, some symbols that will be used in the proposed method are defined. Let S be a suspected image of size $M \times N$. And let it be divided into non-overlapping 8×8 image blocks and use $s_{i,j}$ to denote the resulting image block at position (i, j), where $0 \le i \le \left\lfloor \frac{M}{8} \right\rfloor$ and $0 \le j \le \left\lfloor \frac{N}{8} \right\rfloor$. If the size of the watermark W is larger than the embedding capacity of the cover image, the size of which is $\left\lfloor \frac{M}{8} \right\rfloor \times \left\lfloor \frac{N}{8} \right\rfloor$, we resize W to $\left\lfloor \frac{M}{8} \right\rfloor \times \left\lfloor \frac{N}{8} \right\rfloor$. Otherwise, we keep the size of W unchanged. Then, let the size of the watermark W be $E \times F$ and let w_{ij} be the pixel value of the watermark image at position (i, j), where $0 \le i \le E$ and $0 \le j \le F$. We cluster the non-overlapping 8×8 image blocks of a suspected image into four groups and then extract one group of image blocks in every single round.

Detailed descriptions of clustering an image and a watermark are given as follows. First, we divide image S into non-overlapping sets S_1, S_2, S_3, S_4 and S_5:

$S_1 = \{s_{i,j}\}$ where $i \bmod 8 = 0$, $j \bmod 8 = 0$, $0 \le i \le E$ and $0 \le j \le F$;

$S_2 = \{s_{i,j}\} - S_1$ where $i \bmod 4 = 0$, $j \bmod 4 = 0$, $0 \le i \le E$ and $0 \le j \le F$;

$S_3 = \{s_{i,j}\} - S_2 - S_1$ where $i \bmod 2 = 0$, $j \bmod 2 = 0$, $0 \le i \le E$ and $0 \le j \le F$;

$S_4 = \{s_{i,j}\} - S_3 - S_2 - S_1$ where $0 \le i \le E$ and $0 \le j \le F$;

$S_5 = S - S_4 - S_3 - S_2 - S_1$.

S_5 could be an empty set if $E=\left\lfloor \frac{M}{8} \right\rfloor$ and $F=\left\lfloor \frac{N}{8} \right\rfloor$, but this has no effect on our method. And $|S_1| \le |S_2| \le |S_3| \le |S_4|$. Also, we divide W into non-overlapping sets W_1, W_2, W_3 and W_4 in a similar way:

$W_1 = \{w_{i,j}\}$ where $i \bmod 8 = 0$, $j \bmod 8 = 0$, $0 \le i \le E$ and $0 \le j \le F$;

$W_2 = \{w_{i,j}\} - W_1$ where $i \bmod 4 = 0$, $j \bmod 4 = 0$, $0 \le i \le E$ and $0 \le j \le F$;

$W_3 = \{w_{i,j}\} - W_2 - W_1$ where $i \bmod 2 = 0$, $j \bmod 2 = 0$, $0 \le i \le E$ and $0 \le j \le F$;

$W_4 = W - W_3 - W_2 - W_1$.

And $|W_1| \le |W_2| \le |W_3| \le |W_4|$. An example of clustering a watermark in the use of W_i is shown in Figure 5.5.

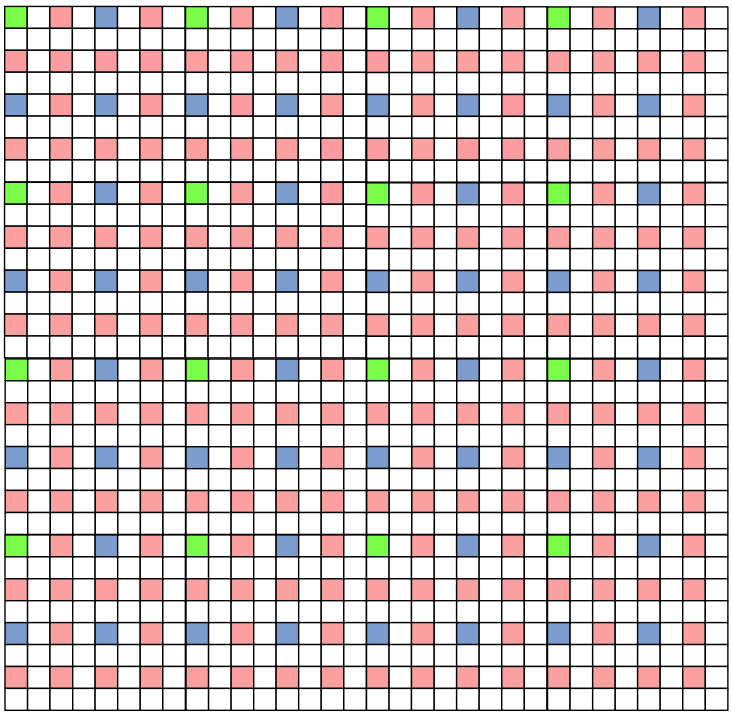
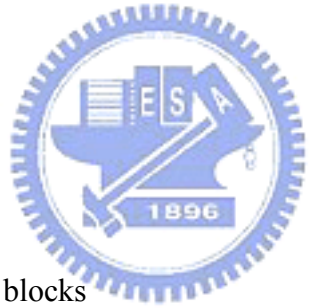


Figure 5.5 An example of clustering a watermark of size 32x32. W_1 is the set of green pixels, W_2 is the set of blue pixels, W_3 is the set of pink pixels, and W_4 is the set of white pixels.



At the beginning, we extract embedded watermark pixels from image blocks which belong to S_1. And we calculate the similarity between the extracted watermark and part of the declared image owner's watermark W_1, which equals the amount of pixels having the same values divided by the amount of the total extracted pixels. If the similarity is lower than a threshold value, which is set according to our experimental experience, we infer right away that the suspected image is a non-infringed image. And we stop the extraction. Only when the calculated similarity is higher than the threshold value do we consider the embedded watermark to be likely the same as the watermark of the declared image owner's. And we will extract watermark pixels from S_2 further. As just mentioned above, we calculate the similarity again and compare it with the threshold value. If the suspected image is not a copyright-infringed one, then we will get into the third extraction from S_3. If we finally get into the fourth extraction, S_4 will be extracted and after this extraction, the watermark is entirely extracted. If the similarity is lower than the threshold value, the suspected image is judged to be a non-infringed one. But if the similarity is still higher than the threshold value, then we judge the suspected image as a copyright-infringed one. A flowchart of the proposed method for fast watermark verification by progressive image matching is shown in Figure 5.6.

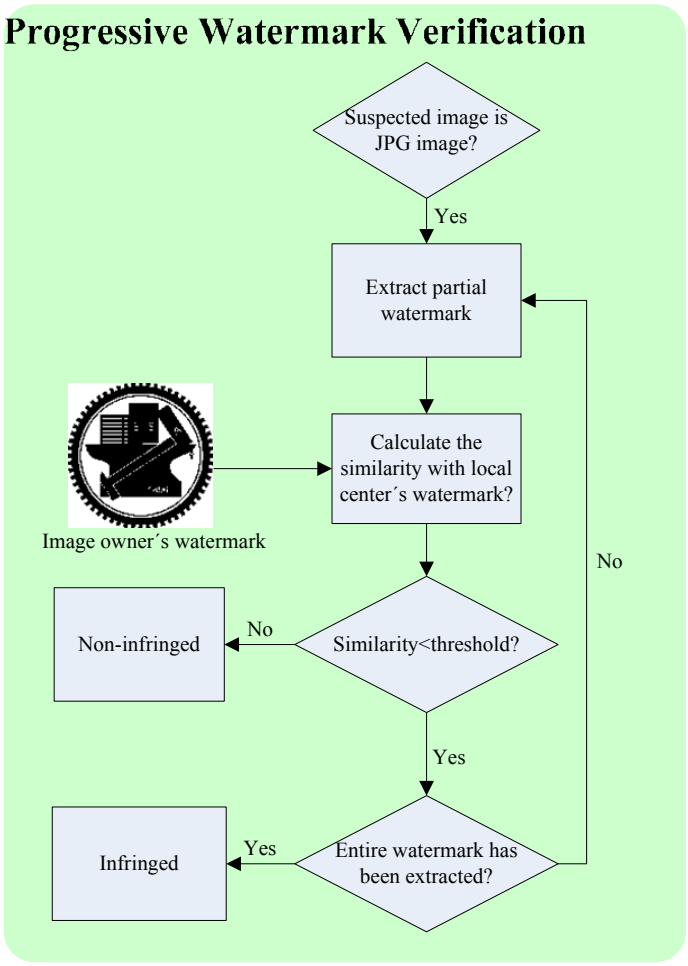


Figure 5.6 A flowchart of fast watermark verification by progressive image matching.

Some important properties of the proposed method should be emphasized. To able to exclude the suspected image which is unlike to be a copyright-infringed one with a low cost, we start to extract the smallest set S_1. Nevertheless, the amount should not be too small, or an unrepresentative similarity will be calculated. And we increase the amount of extracted image blocks round-by-round to make it possible to extract the *entire* watermark within four rounds. In addition, image blocks are clustered by their positions in order to make the extracted watermark pixels look meaningful. That is, we can see an indistinct extracted watermark turning into a clear extracted watermark.

The threshold value is an important comparison standard in the proposed method. It should not happen that we judge a suspected image as a non-infringed one erroneously because of the distortion resulting from geometric attacks. Hence, we set the threshold value by experiments on images subjected to different attacks. Figure 5.7(a) shows the extracted watermark for each round and (b) shows the extracted watermark for each round from an image subjected to the expansion attack. More experimental details and results will be given in Section 5.3.

(a)

(b)

Figure 5.7 (a) Extracted watermark for each round. (b) Extracted watermark from an image subjected to expansion attack for each round.

5.2.2 Detailed Algorithm

The process of fast watermark verification by progressive image matching can be expressed as an algorithm as follows. Let T be the threshold value mentioned previously.

Algorithm 5.1: *Fast watermark verification by progressive image matching*.

Input: A given suspected image S of size $M \times N$ and a declared image owner's watermark W.

Output: A verification result R.

Steps:

1. Divide the suspected image S into non-overlapping 8×8 image blocks. Let $s_{i,j}$ be the image block at position (i, j), where $0 \le i \le \left\lfloor \frac{M}{8} \right\rfloor$ and $0 \le j \le \left\lfloor \frac{N}{8} \right\rfloor$.
2. Resize the watermark W into one of size $\left\lfloor \frac{M}{8} \right\rfloor \times \left\lfloor \frac{N}{8} \right\rfloor$ if it is larger than $\left\lfloor \frac{M}{8} \right\rfloor \times \left\lfloor \frac{N}{8} \right\rfloor$. Otherwise, keep it unchanged. Denote the size of W by $E \times F$ and let $w_{i,j}$ be the pixel value of the watermark image at position (i, j), where $0 \le i \le E$ and $0 \le j \le F$.
3. Extract embedded watermark E_1 using the employed method described in Section 5.1.2 from 8×8 image blocks in the set $S_1 = \{s_{i,j}\}$ where $i \bmod 8 = 0$, $j \bmod 8 = 0$, $0 \le i \le E$ and $0 \le j \le F$. And perform the following operations.
   3.1 Compare E_1 with the corresponding pixels of W, which are in the set $W_1 = \{w_{i,j}\}$ where $i \bmod 8 = 0$, $j \bmod 8 = 0$, $0 \le i \le E$ and $0 \le j \le F$.
   3.2 If the similarity M_1 between E_1 and W_1 is smaller than the threshold value T, go to Step 7.1.
   3.3 If M_1 is larger than T, go to Step 4.
4. Extract embedded watermark E_2 from 8×8 image blocks in the set $S_2 = \{s_{i,j}\} - S_1$ where $i \bmod 4 = 0$, $j \bmod 4 = 0$, $0 \le i \le E$ and $0 \le j \le F$. And perform the following operations.
   4.1 Compare E_2 with the corresponding pixels of W, which are the in set

$W_2 = \{w_{i,j}\} - W_1$ where i mod 4 =0, j mod 4 = 0, $0 \le i \le E$ and $0 \le j \le F$.

4.2 If the similarity M_2 between E_2 and W_2 is smaller than T, go to Step 7.1.

4.3 If M_2 is larger than T, go to Step 5.

5 Extract embedded watermark E_3 from 8×8 image blocks in the set $S_3 = \{s_{i,j}\} - S_2 - S_1$ where i mod 2 =0, j mod 2 = 0, $0 \le i \le E$ and $0 \le j \le F$. And perform the following operations.

5.1 Compare E_3 with the corresponding pixels of W, which are in the set $W_3 = \{w_{i,j}\} - W_2 - W_1$ where i mod 2 =0, j mod 2 = 0, $0 \le i \le E$ and $0 \le j \le F$.

5.2 If the similarity M_3 between E_3 and W_3 is smaller than T, go to Step 7.1.

5.3 If M_3 is larger than T, go to Step 6.

6 Extract embedded watermark E_4 from 8×8 image blocks in the set $S_4 = \{s_{i,j}\} - S_3 - S_2 - S_1$ where $0 \le i \le E$ and $0 \le j \le F$. And perform the following operations.

6.1 Compare E_4 with the corresponding pixels of W, which are in the set $W_4 = W - W_3 - W_2 - W_1$.

6.2 If the similarity M_4 between E_4 and W_4 is smaller than T, go to Step 7.1.

6.3 If M_4 is larger than T, go to Step 7.2.

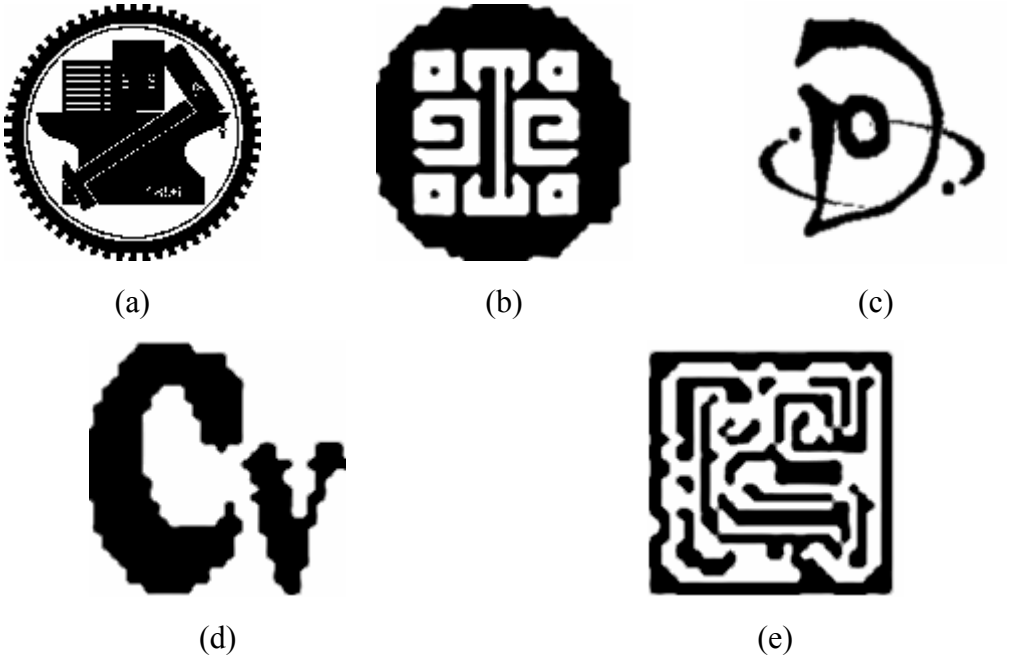
7 A verification result R is obtained to include exclusively one of the following two decisions:

7.1 We judge the suspected image as a non-infringed one.

7.2 We judge the suspected image as a copyright-infringed one.

5.3 Determination of Threshold Value and Experimental Results

Some experiments on setting the threshold value T mentioned previously are shown in this section. Five binary watermarks as shown in Figure 5.8 are used as invisible watermarks. The cover images are all of size 512×512. Figure 5.9(a) is the color image of "Jet," (b) is "Lena," (c) "Baboon," (d) "Painting," and (e) "Pepper."

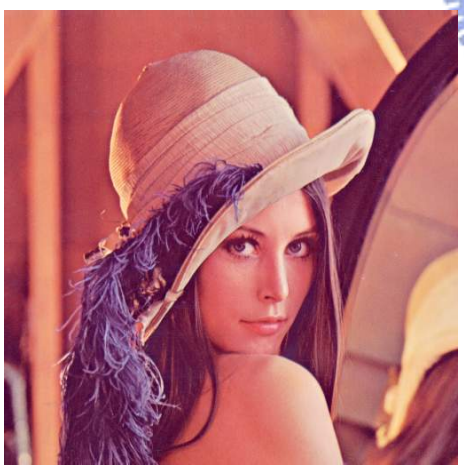


(a) (b) (c)

(d) (e)

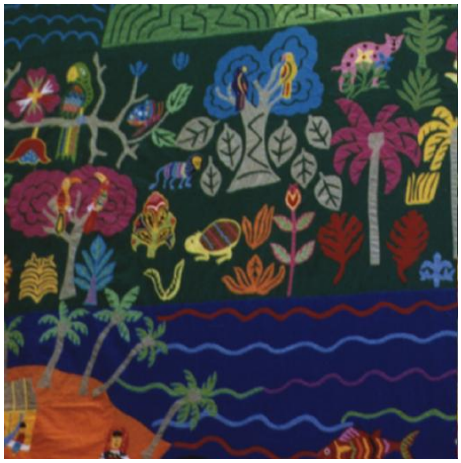
Figure 5.8 Five binary invisible watermark of size 128×128 and 32x32, respectively. (a) Watermark of National Chiao Tung University. (b) Watermark of National Palace Museum. (c) Watermark of digital archive program. (d) Watermark of computer vision laboratory in NCTU. (e) Watermark of National Museum of History.

(a)



(b)

(c)



(d)

(e)

Figure 5.9 Five cover images of size 512×512. (a) Color image of “Jet”, (b) Color image of “Lena,” (c) Color image of “Baboon,” (d) Color image of “Painting,” and (e) Color image of “Pepper.”

The purpose of the proposed method is to exclude non-infringed images as earlier as possible in the progressive checking process. However, it must be guaranteed that copyright-infringed images will not be considered as non-infringed ones. So we conducted experiments on copyright-infringed images and took different attacks into consideration to obtain an appropriate threshold value T for the similarity M_i mentioned in the above algorithm. We experimented on five cover images, and five watermarks of two different sizes were supposed to be embedded into each of them. So we had 150 stego-images. And each stego-image was supposed to be subject to three different attacks, respectively. Each image was allowed to have four rounds of progressive matching. Therefore, we obtained 600 experimental results about the similarity measures between the original embedded watermark and the extracted distorted watermarks after being subject to geometric attacks. The rotation attack and the flipping attack resulted in minor distortion for about one percent of difference on the stego-image, while the expansion attack resulted in major distortion for about nineteen percent of difference. According to the experimental results, 0.8 which is the smallest is finally set to be the threshold value T used in our method. The experimental results of similarity between the original embedded watermark and the extracted distorted watermark after being subject to the expansion attack are partially shown in Table 5.2.

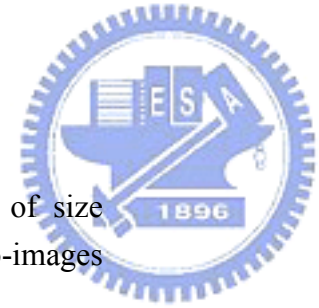


Table 5.2 Similarity in each of four rounds between original watermarks of size 128×128 and distorted watermark extracted from two different stego-images after being subject to the expansion attack.

	Round 1	Round 2	Round 3	Round 4
between distorted watermark ”NCTU ” extracted from “Jet” and original “NCTU”	58/64≈0.91	177/192≈0.92	663/705≈0.94	2732/2883≈0.95
between distorted watermark ”NCTU ” extracted from “Baboon” and original “NCTU”	58/64≈0.91	157/192≈0.82	613/705≈0.87	2470/2882≈0.86
between distorted watermark ”NPM” extracted from “Jet” and original “NPM”	59/64≈0.92	183/192≈0.95	666/705≈0.94	2740/2883≈0.95
between distorted watermark ”NPM” extracted from “Baboon” and original “NPM”	58/64≈0.91	164/192≈0.85	604/705≈0.86	2460/2883≈0.85

However, miscalculating a non-infringed image as a copyright-infringed one will probably happen when the embedded watermark and the declared image owner's watermark is very similar but we think that this is acceptable in our study, because we can finally differentiate the extracted watermark from the declared image owner's by human eyes. But we have experimented on 100 non-infringed images, and fortunately none was miscalculated as a copyright-infringed one.

5.4 Discussions and Summary

In this chapter, a method for fast watermark verification by progressive image matching has been proposed. We extract part of an embedded watermark using an employed watermarking technique and compare the extracted watermark with the corresponding pixels of the declared image owner's watermark. If the similarity between these two is lower than a pre-selected threshold value, we stop the extraction process and judge the suspected image as a non-infringed one. Conversely, if the similarity between these two is higher than the threshold value, we infer that the suspected image is likely to be a copyright-infringed one. And so we have to check the image further. That is, we extract more unprocessed pixels of the embedded watermark for recalculating the similarity to check whether we can judge the suspected image as a non-infringed one or not in this round. If the calculated similarity in the first, second, and third round is all higher than the threshold value, we will finally go into the fourth round. All unprocessed embedded watermark pixels will be extracted in this round. And we calculate the similarity as well as in the former rounds. If the similarity is still higher than the threshold, we can say that the suspected image is a copyright-infringed one. Otherwise, it is not.