

Proposed Algorithm for Digital Image Watermarking Survival against JPEG Compression

Dr. Nidaa F. Hassan 

Computer Science Department, University of Technology/ Baghdad
Email:nidaaalalousi_5@yahoo.com

Ruaa Kadhim Jaber

Computer Science Department, University of Technology/Baghdad

Received on: 5/5/2013 & Accepted on: 15/8/2013

ABSTRACT

In this paper, a new algorithm is proposed to embed a watermark in digital image such that, it can survive against lossy JPEG (Joint Photographic Experts Group) compression. Analyzing operations are performed on the cover image before and after compression to determine strong locations survival against JPEG compression; these locations are used as host for the watermark. A map is used to embed these strong locations indices; used by the receiver to extract these indices. Fidelity Criteria evaluates errors between the original and cover images, good tests are achieved without perceptual degradation of the transparency of the cover image.

Key Words: Watermarking, Lossy Compression, JPEG.

اقتراح خوارزميه لاختفاء العلامة المائية في الصورة الرقمية تصمد أمام ضغط JPEG

الخلاصة

في هذا البحث , تم اقتراح خوارزميه جديدة لإخفاء العلامة المائية في الصور الرقمية يمكنها الصمود ضد ظروف ضعف او فقدان ظروف الضغط (Joint Photographic Experts Group). تم اجراء عمليات تحليل على صورة الغطاء قبل وبعد الضغط لتحديد الاماكن القوية القابلة للصمود امام ضغط JPEG, ان هذه الاماكن تم استخدامها كمضيف للعلامة المائية. اما الخريطة التي تم فيها أخفاء عناوين الاماكن القوية بشكل كتل , يتم استخدامها بواسطة المستلم لاستخراج العناوين . مقاييس الجودة قيمت الاخطاء بين الصورة الاصلية والصورة الغطاء , فقد تم تحقيق نتائج جيدة بدون تسبب انحلال ملحوظ للصورة الغطاء .

INTRODUCTION

Digital watermarking refers to specific information hiding techniques whose purpose is to embed secret information inside multimedia content, like images, video, or audio data. The watermark is typically added to specific field in the original content to identify the file's copyright information (author, rights, etc.) [1].

Although Compression is one of the most important operations that facilitates and speeds up the transmission of large files through the internet, lossy

compression results in losing some details of the file and is considered as one of the attacks that may circumvent the intended purpose of the watermarking technique for a given application and unintentionally destroy or distort the watermark [2].

Watermarking techniques that can embed a resistant watermark against lossy compression is needed, since most files must be compressed before storage or transmission in order to minimize the storage and bandwidth requirements.

In this paper, a new algorithm is proposed to embed a watermark in BMP image, which is resistant to JPEG (Joint Photographic Experts Group) compression.

WATERMARKING

Digital watermarking is a technique which embeds additional information called digital signature or watermark into the digital content in order to secure it, so that a watermark is a hidden signal added to images that can be detected or extracted later to make some affirmation about the host image [3].

The hidden information called watermark may be serial number , random number sequence, copyright messages, ownership identifiers, control signals, transaction dates, creators of the work, text, bi-level or grey-level image, or other digital formats. After inserting or embedding the watermark by specific algorithms, the original media will be slightly modified, and the modified media is called the watermarked media. There might be no or little perceptible differences between the original media content and the watermarked one. After embedding the watermark, the watermarked media are sent to the receiver via the Internet or other transmission channel. Whenever the copyright of the digital media is in question, this embedded information is decoded to identify the copyright owner [4]. Figure (1) shows the generic schema of digital watermarking.

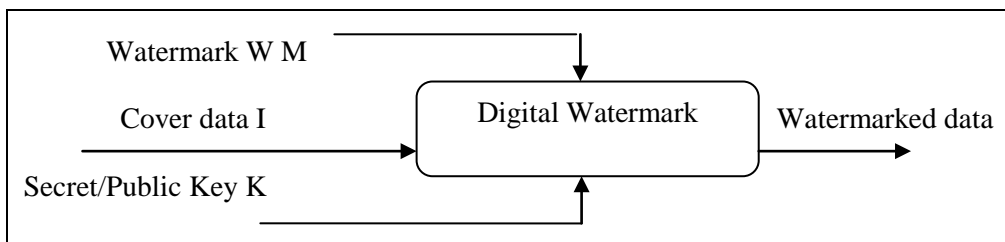


Figure (1) Generic Digital Watermarking Scheme [5].

Digital watermarking methods can be roughly categorized into two types: *Non-blind* and *Blind*. Non-blind methods require the original image at the detection end, whereas blind methods do not. [1].

A digital watermark should have two main properties, which are robustness and imperceptibility. Robustness means that the watermarked data can withstand different image processing attacks, while imperceptibility means that the watermark should not introduce perceptible artifacts [6].

IMAGE COMPRESSION

Image compression is an application of data compression that encodes the original image with few bits. The main goal of image compression is to reduce the storage quantity as much as possible, and the decoded image displayed in the monitor can be similar to the original image [7]. Figure (2) shows compressor / decompressor system.

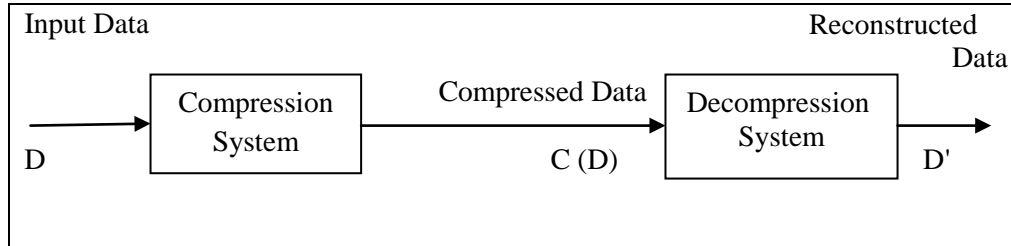


Figure (2) Data Compression and Decompression System [8].

Image compression systems can be classified as *lossless* or *lossy*, depending on whether the image reconstructed from the compressed one is exactly the same as the original image or not. Lossless image compression systems do not alter the original image quality, but they can only achieve low compression ratios. On the other hand, Lossy image compression systems try to take advantage of the quality perceived by the observer, reducing the number of bits needed to represent those parts of the image where the human visual system (HVS) is unable to detect all the information. In order to achieve such a goal, lossy image compression systems usually have architecture of four stages: *preprocessing, transformation, quantization and coding* [9].

The international standard JPEG (Joint Photographic Experts Group) is an example of lossy compression. It is widely used in digital imaging, digital cameras, embedding images in web pages, and many more applications. Whilst aimed at image compression, JPEG has found some popularity as a simple and effective method of compressing moving images (in the form of Motion JPEG). Image data is processed one 8 x 8 block at a time. Color components or planes (e.g. R, G, B or Y, C_r, C_b) may be processed separately (one complete component at a time) or in interleaved order (e.g. a block from each of three color components in succession) [10]. Each block is coded using the steps shown in Figure (3).

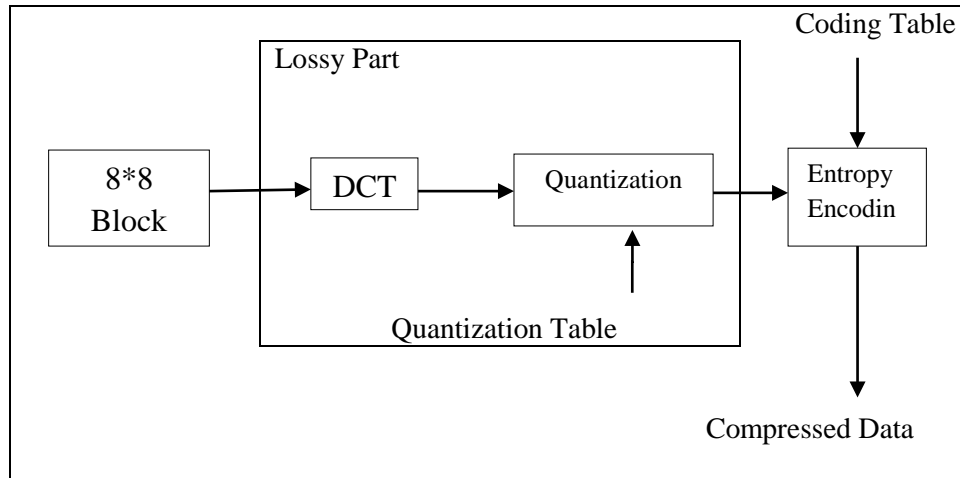


Figure (3) JPEG Compression [10].

PROPOSED METHOD

There is a contradictory between compressions and watermarking, although image compression aims to identify the perceptually insignificant parts of the data and remove them, watermarking techniques try to insert information into them. In this paper, a new watermarking algorithm is proposed to hide secret data in the cover image that are able to survive against JPEG compression, the proposed algorithm consists of the following phases:

Watermark Conversion

In this phase, the watermark (WM) to be embedded, is first split into characters, and the ASCII code for each character is saved in an array called ASCII_array (). For example the WM is (Hello*2013&), the ASCII_array () for this WM is shown in Table (1), algorithm (1) illustrates this conversion.

Table (1) Watermark Conversion.

Character	H	e	l	l	o	*	2	0	1	3	&
ASCII code	72	101	108	108	111	42	50	48	49	51	38

```

Algorithm (1): Watermark Conversion
Input: The watermark (WM).
Output: ASCII_array ( ).
Step 1: Initialize I to 1 .
Step 2: While ( I <= length ( WM ) ) do //WM is split into
characters.
    Character=mid (WM, I, 1)
    ASCII_array (I) = ASCII code for Character: I = I + 1
While end
  
```

Cover Image Analyzing

In this phase, the image to be used as a carrier for the watermark is analyzed. The analyzing operation is performed by subjecting the original image to JPEG and

then decompressing it, so three images are got, the original image (image_1), the compressed image (image_2) and the decompressed image (image_3), Figure (4) shows these images.

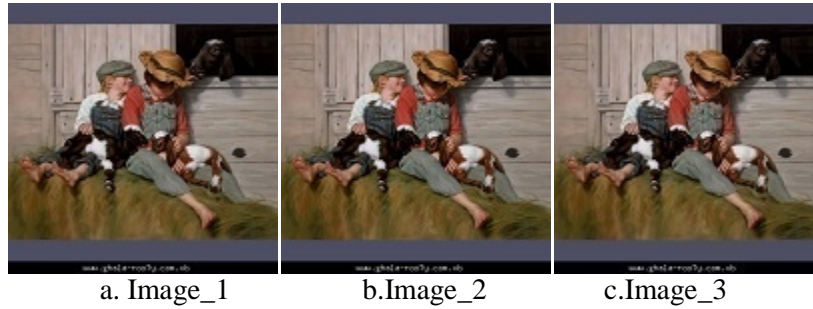


Figure (4) Image_1 is the original image, Image_2 is the compressed image and Image_3 is the decompressed image.

Image_1 is compared byte by byte with image_3, the byte which preserved its value before and after JPEG is considered as strong byte. The indices of strong bytes which will hold WM characters are saved in an array called Index_array (). Table (2) shows the strong and weak bytes in Image_1.

As an example, to embed letter (H) from Table (1), the index of the strong byte having the value (72) is saved in the Index array (), this process is repeated for all characters of the WM, table (3) shows an example of this operation which is illustrated in algorithm (2).

Table (2) Strong and Weak Bytes in Image_1.

Index	Byte from image_1	Byte from image_3	Byte type
60	0	0	Strong
61	1	4	Weak
62	0	0	Strong
63	0	0	Strong
64	2	3	Weak
65	20	20	Strong
66	0	0	Strong
67	2	4	Weak
68	61	61	Strong
...

Table (3) Indices of Strong Bytes which will hold the WM (Hello *2013&).

WM character	Strong bytes indices
H	7049
e	7548
l	7572
l	7638
o	8445
*	9532
2	9734
0	10221
1	11768
3	12258
&	12550

Algorithm (2): Cover Image Analyzing
Input: Original image (image_1), decompressed image (image_3), ASCII_array ().
Output: Index_array ().
Step 1 : Initialize I to 1 , Flag to 1 and index to 0.
Step 2: While (Flag = 1) do // Find a strong byte for each WM character.
 Index = Index + 1: Get image_1,, byte1: Get image_3,, byte3
 If value (byte1) = value (byte3) and value (byte1) = ASCII_array (I) then
 Index_array (I) =Index //Filling Index_array ().
 I = I + 1
 End if
 If (End of ASCII_array ()) then Flag = 0
 While end

Indices Manipulation

After determining the indices in the previous phase, these indices must be embedded in the cover image, so the receiver is able to determine which bytes were used as host for WM characters, i.e. indices are the map that must be followed to extract the embedded watermark. The indices are converted into binary digits before embedding, the problem arises here, each index may need different number of bits to represent it, resulting difficulty at extraction operation. To solve this problem, the indices must be manipulated before they are converted into binary digits.

The manipulation operation includes splitting each index into three digits , such that each digit requires seven bits , hence each index needs 21 bits , thus uniform representation of indices are got. The manipulation is performed by the following equations:

$$N1 = Round\left(\frac{Index}{1000}\right) \dots (1)$$

$$Index = Index - (N1 * 1000) \quad \dots (2)$$

$$N2 = Round \left(\frac{Index}{100} \right) \quad \dots (3)$$

$$Index = Index - (N1 * 100) \quad \dots (4)$$

$$N3 = Index \quad \dots (5)$$

For example, manipulated index (7079) by equations 1,2,3,4, and 5, resulted N1= 7, N2=0 and N3= 79. Table (4) shows manipulating the indices in Table (3).

Table (4) Manipulated Indices.

Index	N1	N2	N3
7049	7	0	49
7413	7	4	13
7419	7	4	19

The last element is the flag refers to the end of the map. Algorithm (3) illustrates indices manipulation.

Algorithm (3). Indices Manipulation
Input : Index_array () .
Output : Manp_index () .
Step 1 :Initialize I to 0.
Step 2: While (Not End Of Index_array ()) do
 I = I + 1: Index = Index_array (I) //split each index into three numbers.
 N1 = Round (N1 / 1000) : Index = Index - (N1 * 1000)
 N2 = Round (Index / 100) : Index = Index - (N2 * 100) : N3 = Index
 Manp_index (I , 1) = N1 : Manp_index (I , 2) = N2 : Manp_index (I , 3) = N3:
 While End

The numbers stored in *Manp_index ()* must be converted into binary digits before they can be embedded in the cover image.

Embedding Process

In this phase, WM and the map are embedded in the cover BMP image, BMP is treated as three parts as shown in Figure (5).

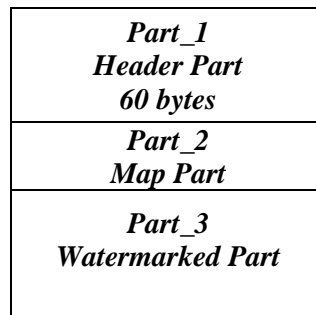


Figure (5) Parts of the Cover Image.

Part_1: This part contains the header of the cover image, its values are not allowed to be changed, so it is not used for embedding the map or the watermark, hence it is skipped (totally 60 bytes in size).

Part_2: *Bits_array* () are embedded in this part. Embedding is accomplished as follows:

For each bit, a block of size 5 bytes is used.

If bit =1 then the block bytes are replaced with the value 180.

If bit = 0 then the block bytes are replaced with the value 77.

The values 180 and 77 are selected after testing a number of threshold values, where it is found that these two values remain unchangeable against compression.

Part_3: This part is the watermarked part; it contains the strong bytes that hold WM characters. Embedding process is illustrated in algorithm (5).

Algorithm (5): Bits Embedding.

Input: The cover image (Cov_image), Bits_array ().
Output: Watermarked image (Wat_image).

Step 1: Initialize I to and J to 1.
// Header part skipping.
Get Cov_image , , Byte
Write Wat_image , , Byte :
While End

Step 2: While (I <= 60)

Step 3 : While (Not End Of Bits_array ()) do // Embedding process.
Bit = Bits_array (I).
For J = 1 to 5
If Bit = 1 then Byte = 180 Else Byte = 77: Write Wat_image , , Byte : Next J
I = I + 1: While end

Step 4: While (Not End of Cov_image) // Transferring remaining Cov_image bytes .
Get Cov_image , , Byte : Write Wat_image , , Byte
While End

Extraction Operation

In the embedding process, the watermark and its map are embedded in the cover image, the cover image is compressed using JPEG and sent to the receiver. At the receiver side the extraction process is performed as follows:

1. The cover image is received as compressed image, so it must be decompressed.
2. Header of cover image is skipped, which is equal to 60 bytes.
3. Map extraction, each value within the block of size 5 bytes checked as follows :

If the values within the range [120 to 180] then the extracted bit is 1.

While, If the values within the range [50 to 90] then the extracted bit is 0.

Table (5) shows an example of the extraction operation.

Table (5) Extraction Operation .

Block number	Index	Original block values	After embedding the map	After decompression	Embedded value 180 or 77	Embedded bit 1 or 0
1	60	43	180	112	Out of range	1
	61	85	180	160	180	
	62	66	180	165	180	
	63	45	180	170	180	
	64	84	180	135	180	
2	65	66	77	88	77	0
	66	49	77	85	77	
	67	75	77	99	Out of range	
	68	57	77	79	77	
	69	40	77	74	77	

Extracted bits are stored in an array called *its_array* (). Algorithm (6) illustrates map extraction.

Algorithm (6): Map Extraction.
Input: Watermarked image (*Wat_image*) .
Output: *Bits_array* ()
Step 1 : Initialize Flag to 1 , Value_1, Value_0 to 0 , and K ,J, I to 0 .
Step2: While (Flag = 1) do // Map Bits extraction starting beyond header part.
 For J = 1 to 5 Get *Wat_image* , , Byte
 If (Value (Byte) >=120 and Value (Byte) <= 180) then Value_1 = Value_1 + 1
 If (Value (Byte) >= 50 and Value (Byte) <= 90) then Value_0 = Value_0 + 1
 Next J I = I + 1
 If Value_1 > Value_0: *Bits_array* (I) = 1: K = 0
 Else
 Bits array (I) = 0: K = K + 1
 End if: If (K = 21) then Flag = 0
While End

Indices Calculation

After extracting the map bits in the previous phase, these bits are used to calculate the indices of the bytes that hold WM. Indices are calculated as follows:

1. (21) Bits represent one index.
2. The (21) bits are divided into three sections, each of 7 bits.
3. Each 7 bits are converted into number, i.e. three numbers are resulted N1, N2 and N3
4. The index is calculated using the following equation :

$$Index = (N1 * 1000) + (N2 * 100) + N3 \quad \dots (6)$$

As an example, if the extracted 21 bits are (000011100000000110001), then N1=(0000111), N2=(0000000) and N3 = (0110001). The index is calculated by equation 8 as follows:
 Index= (7*1000) + (0*100) +49 = 7049. An index calculation is illustrated in algorithm (7).

Algorithm (7) : Indices Calculation .

Input : Bits_array () .

Output : Index_array () .

Step 1 : Initialize I to 1 , K to 0 , Num to 0 , T to 1 , J to 1 and D to 0 .

Step 2: Converting the bits stored in the Bits_array () into integer numbers stored in the Num_array () .

Step 3: For I = 1 to K

// Indices calculating

N1 = Num_array (I)

I = I + 1

N2 = Num_array (I)

I = I + 1

N 3= Num_array (I)

Index = (N1 * 1000) + (N2 * 100) + N3

D = D + 1

Index_array (D) = Index

Next

Watermark Extraction

The indices that are calculated in the previous phase are used as guide for the WM, where the byte at each index represents the ASCII code for one of the WM characters, as shown in Table (6). Algorithm (8) illustrates WM extraction.

Table (6) Watermark Extraction.

Index	7049	7413	7419	...	12568	12814
Byte Value	72	101	108	...	51	38
Character	H	e	l	...	3	&

Algorithm (8): Watermark Extraction.

Input : The watermarked image (Wat_image) , Index_array () .

Output : The embedded Watermark (WM) .

Step 1 : Initialize I to 0 .

Step 2 : Get Wat_image , , All_Byte () //all Cov_image bytes are put in All_Byte () .

```

Step 3 : While ( Not End Of Index_array ( ) ) do // Extraction Operation.
                                                I = I + 1
                                                Index = Index_array ( I )
ASCII_code = All_Byte ( Index ) // Get the bytes at the
                                specified index.
Character = Char ( ASCII_code ) // Convert the byte value into
                                character.
WM = WM + Character //Merge characters to get the
                                watermark.
                                While End
    
```

RESULTS AND DISCUSSION

The removal of “irrelevant visual” information may lead to a loss of real or quantitative image information. Two types of criteria can be used to quantify a loss of information:

1. **Objective Tests:** information loss is expressed as a mathematical function of the input and output of the compression process.
2. **Subjective Tests:** This kind of measures requires a definition of a qualitative scale to assess image quality. This scale can then be used by some human expertise to subjectively determine the fidelity. Evaluation with subjective measures requires careful selection of the test subject and carefully designed evaluation experiments [11].

Objective tests are adopted for testing the performance of the suggested method, and they are MAE (Mean Absolute Error), MSE (Mean Square Error), PSNR (Peak Signal to Noise Ratio), SNR (Signal to Noise Ratio) and Correlation Measures. These measures define the overall error between (N) samples of the input BMP image file, and the corresponding samples of the BMP watermarked image file. This kind of measures can be a good testing tool for quality; Table (7) list object tests for original images and watermarked images.

Table (7) Test images before and after watermarking.

Test Image	Watermark Images	MAE	MSE	PSNR	SNR	NMSE	Correlation Measure
Image_1	Image_1	0.35	28.8	33.5	22.6	644028	0.99
mage_2	Image_2	0.34	33.4	32.8	22.7	2200	0.99
Image_3	Image_3	0.53	57.3	30.5	24.7	262175	0.99

From the results listed in the above table, it is noted that these metrics give good results for keeping the watermarked image transparent , where higher PSNR and lower MSE values imply closer resemblance between the watermarked and the original images. Figure (6) shows an example of test images before and after hiding a watermark.







<i>Original Image</i>	<i>Watermarked Image</i>
	
<i>Test Image_1</i>	
	
<i>Test Image_2</i>	
	
Test Image_3	

Figure (6) Test images before and after hiding a watermark.

CONCLUSIONS

This paper presents a new watermarking algorithm; the basic concept of this algorithm is to select the optimal hosts that can stand against JPEG compression. To obtain a blind watermarking, a map of hiding path (strong locations) is stored in the host image.

REFERENCES

- [1]. Teruya and A. Kentaro , M. "A blind digital image watermarking method using interval wavelet decomposition" , International Journal of Signal Processing, Image Processing and Pattern Recognition Vol. 3, No. 2, June, 2010 .
- [2]. Rathee A. and Sharma S., "Security of Cyber Data With Reference to Image Watermarking Techniques", 2012.

- [3]. Mohamed Sathik and S. S. Sujatha , M. " An Improved Invisible Watermarking Technique for Image Authentication" , International Journal of Advanced Science and Technology ,Vol. 24 , November, 2010.
- [4]. Murty M., Veeraiah D. and Rao A. , " Digital Signature and Watermarking Methods For Image Authentication using Cryptography Analysis " , 2011.
- [5]. Katzenbeisser, S., Petitcolas F. , " Information Hiding Techniques for Steganography and Digital Watermarking " , 2000 .
- [6]. Hartung F. and Kutter M. , "Multimedia Watermarking Techniques " ,1999.
- [7]. Yi Wei W. , " An Introduction to Image Compression" ,2011.
- [8]. Acharya T. and Tsai P. , " JPEG2000 Standard for Image Compression Concepts, Algorithms and VLSI Architectures" , 2005.
- [9]. Joancomarti J. , Minguillon J. and Megyas D. , " A family of image watermarking schemes based on lossy compression",2000.
- [10]. Richardson I. " Video CODEC Design ,developing image and video compression systems" , 2002.
- [11]. Umbaugh S., "Computer Vision and Image Processing", Prentice Hall PTR, 1998.