20

Digital Watermarking Using MATLAB

Pooya Monshizadeh Naini
University of Tehran
Iran

1. Introduction

Embedding a hidden stream of bits in a file is called Digital Watermarking. The file could be an image, audio, video or text. Nowadays, digital watermarking has many applications such as broadcast monitoring, owner identification, proof of ownership, transaction tracking, content authentication, copy control, device control, and file reconstruction (Cox et. al., 2008).
In literature, the host file is called the "asset", and the bit stream is called the "message". The main specifications of a watermarking system are: Robustness (Against intentional attacks or unintentional ones such as compression), Imperceptibility, and Capacity. Importance of each depends on the application. As a matter of fact there is a trade-off between these factors (Barni & Bartolini, 2004). Although watermarking in some literature includes visible imprints, here we only mean the invisible embedding of the data.
In this chapter, we will introduce how to use MATLAB to implement image watermarking algorithms. These algorithms include the most famous ones which are widely used in current literature or more complicated approaches are based upon. These are commonly divided into three categories (Barni & Bartolini, 2004)

1. Watermarking in Spatial Domain
2. Watermarking in Spectral Domain
3. Watermarking in Hybrid Domain

In section 2 we will go through some basic image processing commands in MATLAB. Section 3 provides information about different fundamental watermarking methods. Evaluating the algorithms is discussed in Section 4, and finally section 5 brings a conclusion.

2. Basic image processing commands in MATLAB

Digital Image, like many other files, is known as a matrix in MATLAB. Here, we go through several basic image processing commands.

2.1 Loading an image

For loading an image, it is better to put the image in the same folder with the m-file. This way, the image can be easily loaded through "imread" command:

```
A = imread('lena.tif');
```

Else if the image is in a different folder, it should be fully addressed:

```
A = imread('C:\Users\User1\Desktop\lena.tif');
```

The supported formats by MATLAB are: bmp, cur, fts(fits), gif, hdf, ico, j2c(j2k), jp2, jpf(jpx), jpg(jpeg), pbm, pcx, pgm, png, pnm, ppm, ras, tif(tiff), and xwd. 'A' is now a matrix of pixels brightness values. If the image is in black and white, the matrix is 2-dimmensional. However, if there is a color image, we will have a 3-dimmensional matrix, which has three planes of main colors: Red, Green, and Blue. The number of bits that are needed to preserve the value of every pixel is called "bit depth" of the image. The output class of "imread" command is "logical" for depth of one bit, "uint8" for bit depth between 2-8, and "uint16" for higher bit depths.

2.2 Displaying an image

The most common command for displaying an image(matrix) is "imshow":

```
imshow(A);
```

This command can also depict matrices with double values. If the values are not between 0-255, it is better to map them to this region. This can be simply done by adding an empty matrix to the command. This way, the lowest value of the matrix is considered '0', and the highest is considered 255:

```
imshow(A, [ ] );
```

2.3 Creating an image

"imwrite" is used for creating an image file out of a matrix. The image file is created in the same folder with the m-file if no address is given. This command has some useful parameters such as JPEG image compression ratio:

```
imwrite(A,'wm_lena.jpg','Mode','lossy','Quality',65,'Bitdepth',8);
```

3. Watermarking methods

As mentioned in Introduction there are 3 main categories for digital watermarking methods.

3.1 Watermarking in spatial domain

The message can be any coded or straight arrange of bits. Furthermore, the message can be another image. Consider the asset and the message as shown in Fig. 1.
Fig. 2 shows different bit-planes of the asset with a depth of 8 bits. Bit-plane is the plane that one specific bit of every pixel create.
The command "bitget" can be used here to create the bit-plane splitter function as depicted below:

```
function [B8,B7,B6,B5,B4,B3,B2,B1] = bitplane (pic)
   B1 = bitget(pic,1)*2^0;
   B2 = bitget(pic,2)*2^1 ;
   B3 = bitget(pic,3)*2^2 ;
   B4 = bitget(pic,4)*2^3 ;
   B5 = bitget(pic,5)*2^4 ;
   B6 = bitget(pic,6)*2^5 ;
   B7 = bitget(pic,7)*2^6 ;
   B8 = bitget(pic,8)*2^7 ;
end
```

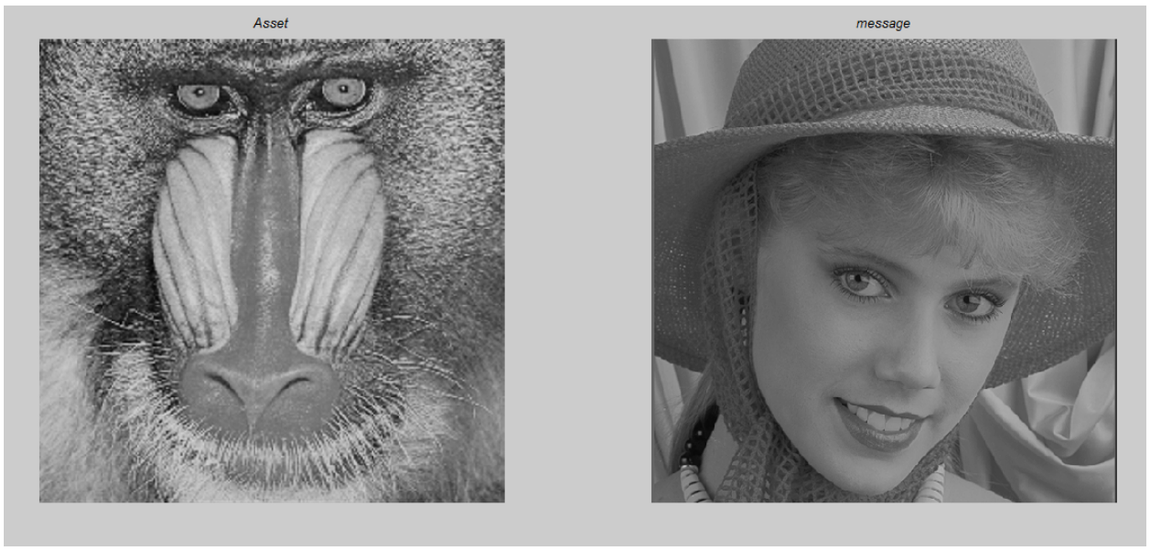


Fig. 1. Examples of asset and message

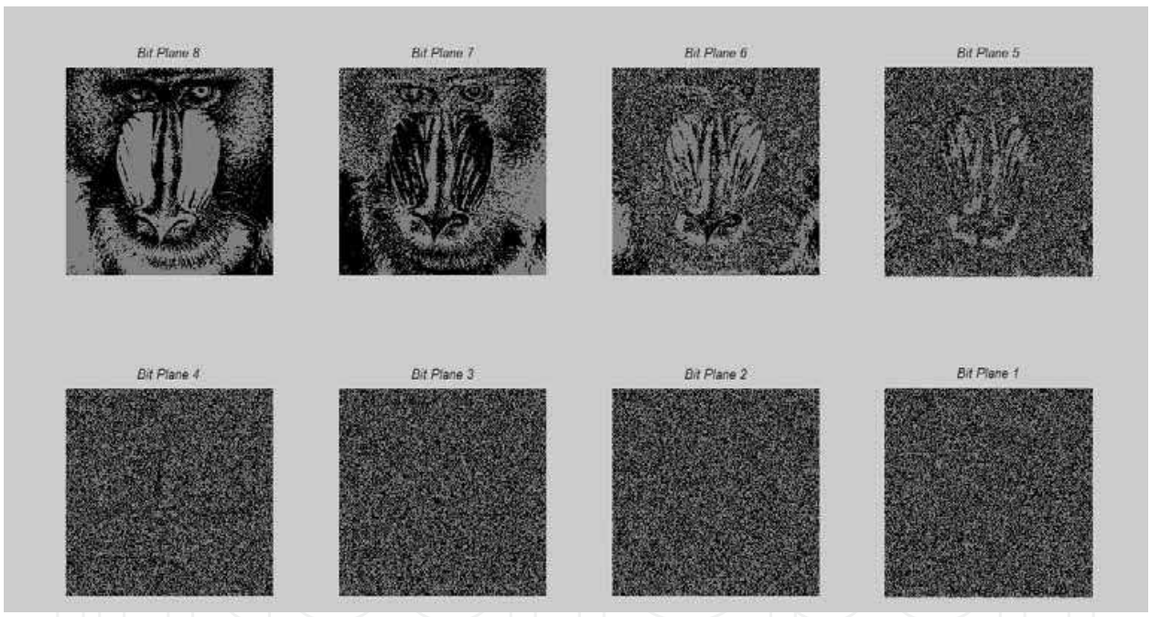


Fig. 2. Bit-planes of the asset image

The first bit-plane is the least significant one (LSB) and most of the time is hardly related to the main shapes of the picture. On the other hand, the last bit-plane is the most significant one (MSB) and contains the main lines and edges of the picture. We consider this image as the asset file. The message also, as shown in Fig. 3, contains 8 bit-planes. Note that the same story is true about lower bit-planes.

Now, as depicted in Fig. 4, we put the significant message bit-planes instead of insignificant bit-planes of the asset, and reconstruct the mandrill image.

As clear in Fig. 5, the resulting watermarked image has a good quality and the watermark message is imperceptible.

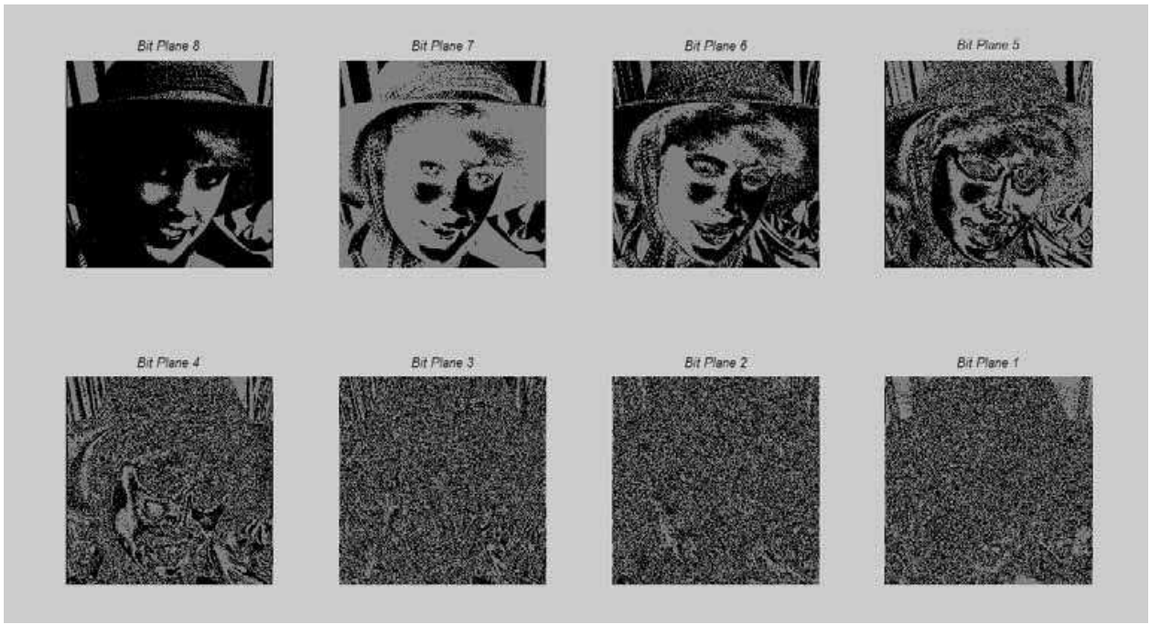


Fig. 3. Bit-planes of the message image

Fig. 4. Substituting the LSB(s) of the asset with MSB(s) of the message

The extraction process simply contains another bit-planes extraction and reconstruction of the message using insignificant bit-planes:

```
message = B3_w * 2^7 + B2_w * 2^6 + B1_w * 2^5;
```

Fig. 6 depicts the extracted message. Note that the main shape of the message is preserved by its highest bit-planes.

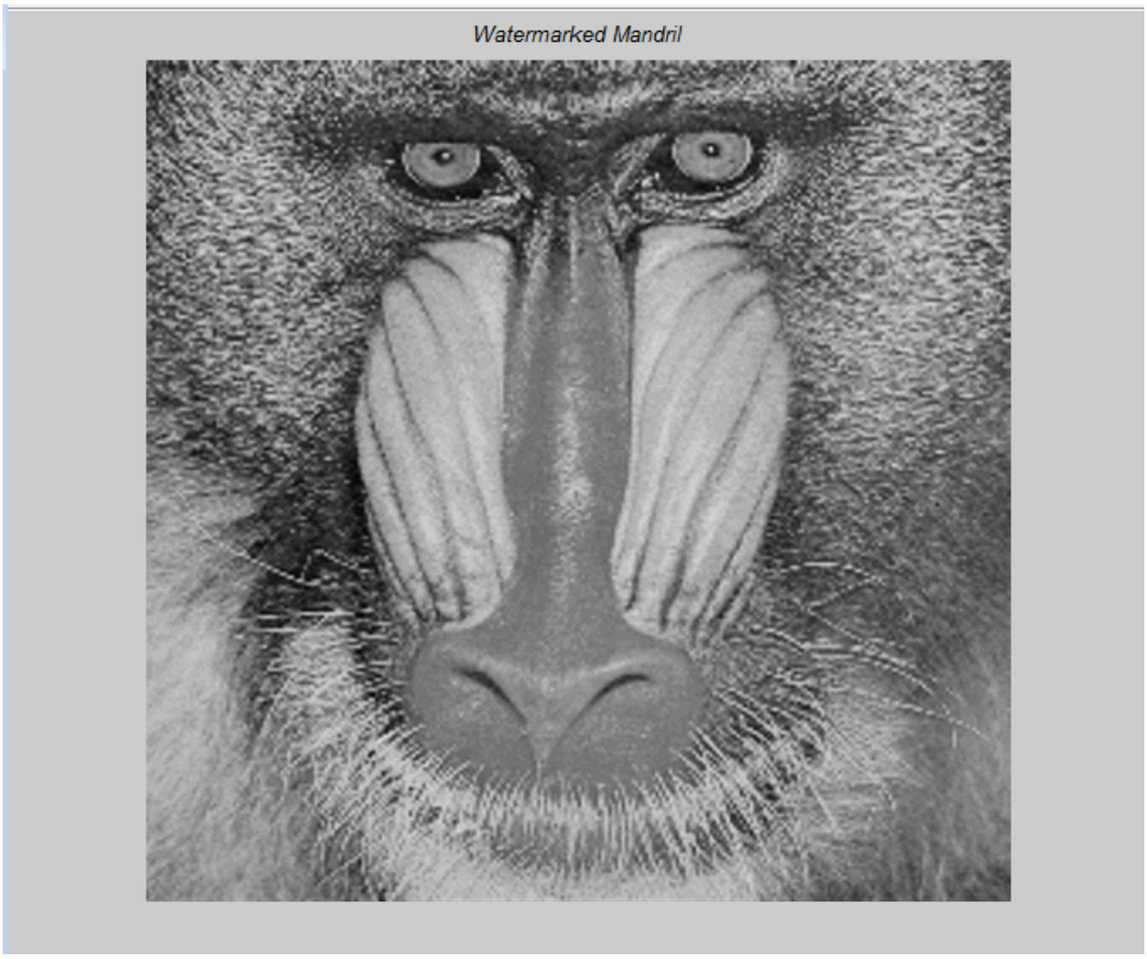


Fig. 5. Watermarked image

3.2 Watermarking in spectral domain

There are several transforms that brings an image into frequency domain. Among most common of those, we can mention are: Discrete Cosines Transform (DCT) and Fast Fourier Transform (FFT).

In frequency domain, coefficients are slightly modified. This will make some unnoticeable changes in the whole image and makes it more robust to attack compared to what we have in spatial methods. One of the most popular approaches in this category is the one proposed by Cox et al which is cited by 4166 articles so far according to Google Scholar[1]. In this method, discrete cosines transform (DCT) is applied on the asset image as shown in Fig. 7. Fortunately, there is a direct command for obtaining DCT coefficients of images:

```
B = dct2(A);
```

Note that the output is a matrix of the same size, but with values of "double" class. As illustrated in Fig. 7, the absolute values of the coefficients corresponding to the low

[1] http://scholar.google.com/scholar?cites=11123322117781572712&as_sdt=2005&sciodt=0, 5&hl=en

frequencies are higher and appear in the up-left corner of the square, while high frequency coefficients appear in down-right with lower absolute values.

Fig. 6. Extracted message

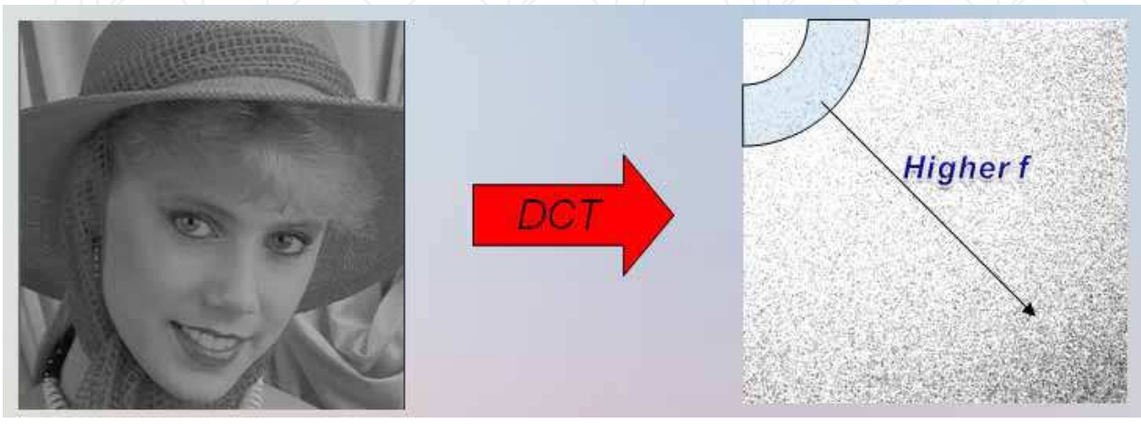


Fig. 7. Bringing an image into DCT domain

To have a better concept of values it is worth to mention that the largest value (51,614) is corresponding to the DC value of the image placed in position (0,0) of the square. "imshow" is used for displaying the DCT coefficients. The message is also coded into an spread spectrum sequence. This step makes the watermarking message robust against many attacks such as JPEG compression which aims to omit the unnoticeable details in high frequencies. Now how the message is added to the asset? Fig. 8 describes the process.

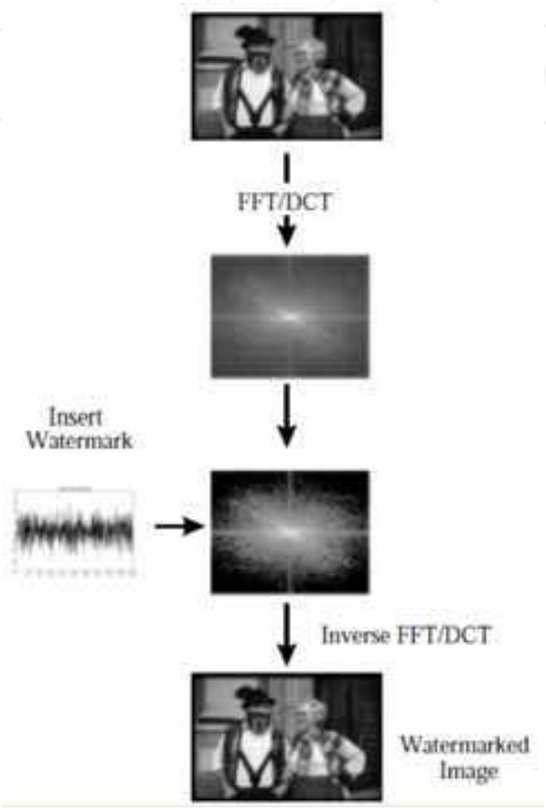


Fig. 8. Watermark embedding in spectral domain (Cox et. al. ,1997)

The question still remains that which coefficients are going to change and how. Cox et. al. use 1000 largest coefficients to embed a watermark sequence of length 1000. The only exception is the DC term, located in (0,0) of the DCT matrix, that should not be changed due to its perceptible change in the whole brightness of the picture. On the other hand, high frequencies are easily changed under common attacks such as compression. Nevertheless, the author suggests not to change some coefficients near to DC term due to their noticeable change. The suggested area is approximately depicted in Fig. 7 .Coefficients are modified according to the stream bits of the message using to the equation 1 (Cox et al., 1997):

$$C_{AW} = C_A \cdot (1 + \alpha \cdot W_i) \tag{1}$$

In which C_{AW} is the watermarked coefficient, C_A is the original one, α represents watermarking strength (e.g. 0.3), and W_i is the corresponding bit of the message data. The formula easily suggests that if a coefficient is larger, it should be modified to a greater extent. We can write the code for the method so far as follows (α=0.1):

```
[fname pthname]=uigetfile('*.jpg;*.png;*.tif;*bmp','Select the Asset Image'); %select image
I=imread([pthname fname]);
wmsz=1000; %watermark size
I=I(:,:,1);%get the first color in case of RGB image
[r,c]=size(I);
D=dct2(I);%get DCT of the Asset
D_vec=reshape(D,1,r*c);%putting all DCT values in a vector
[D_vec_srt,Idx]=sort(abs(D_vec),'descend');%re-ordering all the absolute values
W=randn(1,wmsz);%generate a Gaussian spread spectrum noise to use as watermark signal
Idx2=Idx(2:wmsz+1);%choosing 1000 biggest values other than the DC value
%finding associated row-column order for vector values
IND=zeros(wmsz,2);
for k=1:wmsz
   x=floor(Idx2(k)/r)+1;%associated culomn in the image
   y=mod(Idx2(k),r);%associated row in the image
   IND(k,1)=y;
   IND(k,2)=x;
end
D_w=D;
for k=1:wmsz
   %insert the WM signal into the DCT values
D_w(IND(k,1),IND(k,2))=D_w(IND(k,1),IND(k,2))+.1*D_w(IND(k,1),IND(k,2)).*W(k);
end
I2=idct2(D_w);%inverse DCT to produce the watermarked asset
```

The extraction process is simply subtracting the original DCT coefficients from the watermarked image ones. The code can be written like below:

```
W2=[];%will contain watermark signal extracted from the image
for k=1:wmsz
  W2=[W2(D_w(IND(k,1),IND(k,2))/D(IND(k,1),IND(k,2))-1)*10];%watermark extraction
end
```

Fig. 9 illustrates the process.
Cox et. al. provide equation (2) to check the similarity between the extracted watermark and the original sequence.

$$\mathrm{sim}\left(X,X^{*}\right)=\frac{X^{*}\cdot X}{\sqrt{X^{*}\cdot X^{*}}} \tag{2}$$

In which "X" is the original and "X*" is the extracted message. Creating a function for this equation would be useful:

```
function SIM=WM_detect(Wstar,Worig)
SIM=sum(Wstar.*Worig)/sqrt(sum(Wstar.*Wstar));
end
```

If the extracted message similarity is checked with 1000 random sequences including the original, a result such as what can be seen in Fig. 10 is obtained. Regarding this plot, a suitable value of threshold can be set to detect the original watermark.

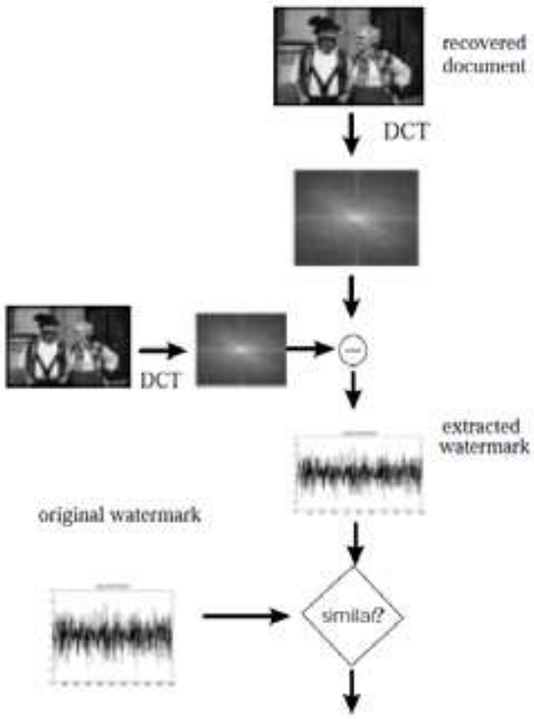


Fig. 9. Watermark extraction (Cox et. al., 1997)

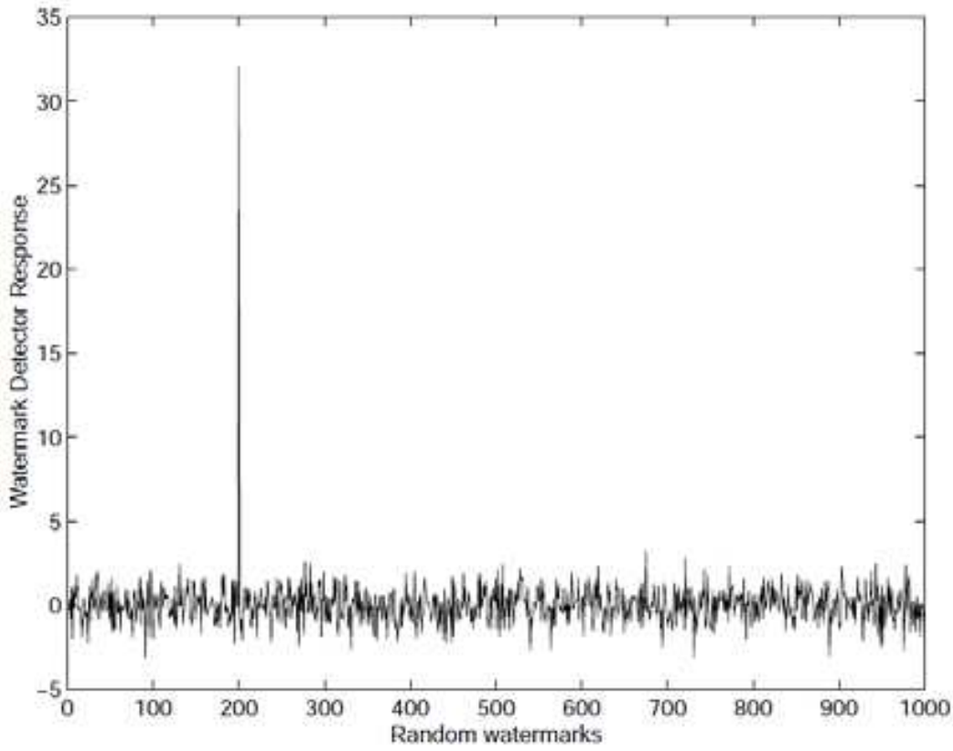


Fig. 10. Detector response to 1000 random sequences including the original (Cox et. al., 1997)

This algorithm, despite the previous in spatial domain, needs the original asset for extraction. Methods like these are called "non-blind detection" (LSB was a sample of "blind detection"). A solution for this can be setting fix mid-frequency coefficients for preserving the watermark message (Barni & Bartolini, 2004).

3.3 Watermarking in hybrid domain

Watermarking in hybrid domain means modifying the image regarding both spatial and spectral specifications. One popular algorithm in this domain is performing the previous method in small blocks of the image. This could happen in 8×8 blocks which ideally match JPEG compression to provide least distort to the message facing with JPEG compression attack (Barni & Bartolini, 2004). Fig. 11 illustrates this method. Pixels in blue represent intensity of middle frequencies in the image and are most suitable for carrying message data. The code has not been brought here because it is simply performing spread spectrum algorithm in separate smaller blocks.

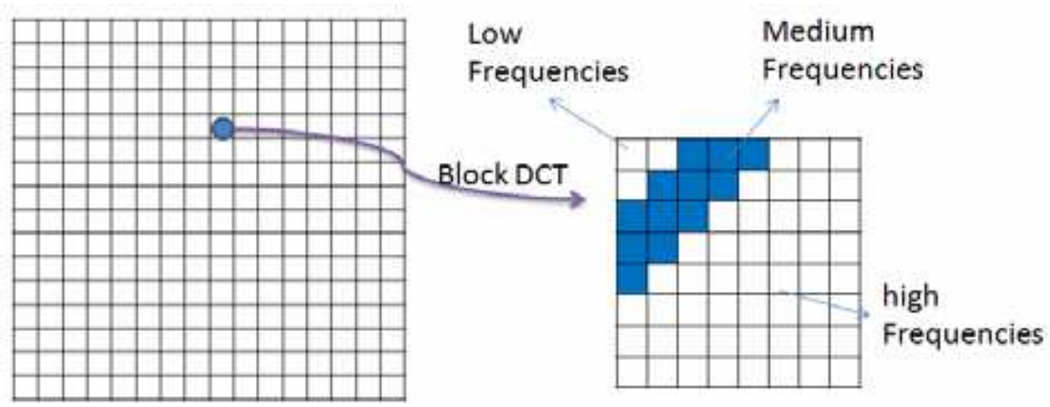


Fig. 11. Block-based hybrid method (recreated from Barni & Bartolini, 2004)

Another famous example of this is "Discrete Wavelet Transform" abbreviated as DWT. For bringing an image to the wavelet domain one can easily use the "dwt2" command:

```
[bA,bH,bV,bD] = dwt2(A, 'wname');
```

In which "wname" is the type of the filter you prefer to use as wavelet decomposition and reconstruction filters. This can be among the filter families of Daubechies ('db1'), Coiflets ('coif1'), Symlets ('sym2'), Discrete Meyer ('dmey'), Biorthogonal ('bior1.1'), and Reverse Biorthogonal (rbio1.1). The option is also provided that you can use your own defined filters:

```
Lo_D =  [1 2 1]/4;      % LP Decomposition Filter
Hi_D =  [1 -2 1]/4;     % HP Decomposition Filter
[bA,bH,bV,bD] = dwt2(A, Lo_D, Hi_D);
```

The same story is true about the inverse transform:

```
A = idwt2(bA,bH,bV,bD, 'wname')
```

Or

```
Lo_R =  [-1 2 6 2 -1]/8;      % LP Reconstruction Filter
Hi_R =  [1 2 -6 2 1]/8;     % HP Reconstruction Filter
B = idwt2(bA,bH,bV,bD, Lo_R, Hi_R);
```

The output decomposed matrices of "dwt2" are of class "double" and contain negative or above 255 values. Hence, there should be some mappings if you tend to display them. The result is provided in Fig. 12. In the code, the maximum value for sub-bands is set to 60 which is mapped to 255. This is because they have tiny values comparing the LL image.

```
[bA,bH,bV,bD] = dwt2(A,'bior1.1');
B=[bA,bH;bV,bD];
subplot(2,2,1);
imshow(abs(bA),[]);
subplot(2,2,2);
imshow(abs(bH),[0 60]);
subplot(2,2,3);
imshow(abs(bV),[0 60]);
subplot(2,2,4);
imshow(abs(bD),[0 60]);
```

Fig. 12. Single level wavelet decomposition

Watermarking usually takes place in sub-bands. Just like the spread spectrum method, largest coefficients can be modified according to a similar equation to (1). Another solution is to change LSBs of these values (Vatsa et. al., 2006).

4. Evaluation of watermarking methods

Several Functions are used to qualify the watermarking algorithm, examining tests on the resulted watermarked image.

4.1 Imperceptibility

The imperceptibility of the watermark is tested through comparing the watermarked image with the original one. Several tests are usually used in this regard.

4.1.1 MSE

Mean Squared Error (MSE) is one of the earliest tests that were performed to test if two pictures are similar. A function could be simply written according to equation (3).

$$M.S.E = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - X_i^*\right)^2 \tag{3}$$

```
function out = MSE (pic1, pic2)
e=0;
[m,n]=size(pic1);
for i=1:m
   for j=1:n
      e = e + double((pic1(i,j)-pic2(i,j))^2);
   end
end
out = e / (m*n);
end
```

4.1.2 PSNR

Pick Signal to Noise Ratio (PSNR) is a better test since it takes the signal strength into consideration (not only the error). Equation (4) describes how this value is obtained.

$$PSNR = 10.\log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{4}$$

```
function out=PSNR(pic1, pic2)
e=MSE(pic1, pic2);
m=max(max(pic1));
out=10*log(double(m)^2/e);
end
```

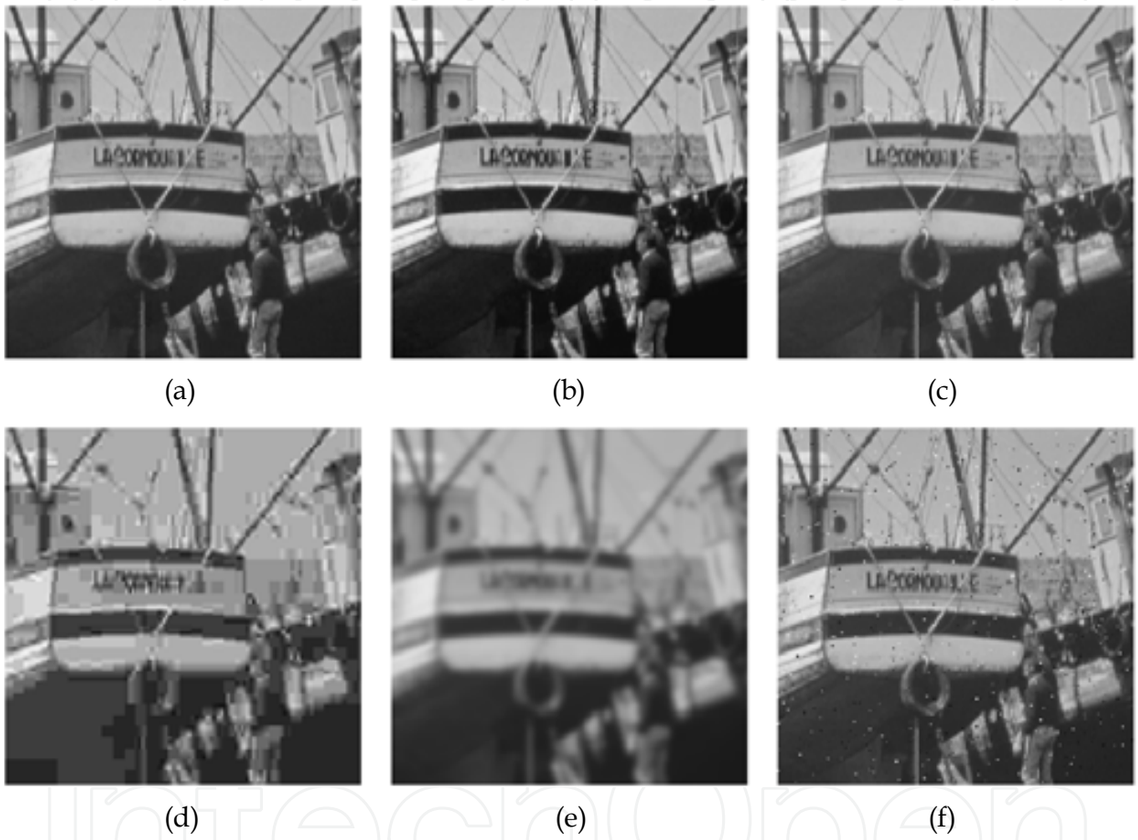
4.1.3 SSIM

The main problem about the previous two criteria is that they are not similar to what similarity means to human visual system (HVS). Structural Similarity (SSIM) is a function

defined as equation (5) by Wang et. al. in 2004 which overcame this problem to a great extent.

$$SSIM = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{5}$$

Where "μ", "σ", & "σ_{xy}" are mean, variance, and covariance of the images, and "c1, c2" are the stabilizing constants. SSIM has a value between 0-1. Similar images have SSIM near to 1. Fig. 13 illustrates the magnificent advantages of SSIM over MSE.



(a) (b) (c)

(d) (e) (f)

Fig. 13. Comparison of MSE and SSIM. All altered images have the same MSE=210 (a) Original image (b) SSIM=0.9168 (c) SSIM=0.9900 (d) SSIM=0.6949 (e) SSIM=0.7052 (f) SSIM=0.7748. (Wang et. al., 2004)

The MATLAB code is available on authors' webpage.[2]

4.2 Robustness

The robustness of a watermark method can be evaluated by performing attacks on the watermarked image and evaluating the similarity of the extracted message to the original one.

[2] http://www.cns.nyu.edu/~lcv/ssim/ssim_index.m

4.2.1 Compression attack

The most used image compression is definitely JPEG. In MATLAB, for compressing an image to different quality factors, the image should be created from a matrix and be reread:

```
imwrite(A,'wm_lena.jpg','Mode','lossy','Quality',75);
A = imread ('wm_lena.jpg');
```

4.2.2 Noise attack

Adding noise in MATLAB is simply done by "imnoise" command. Gaussian, Poisson, Salt & Pepper, and Speckle are among the noises that could be used here. Fig. 14 shows the result of the code:

```
Lena = imread('lena.tif');
Lena = imnoise(Lena,'salt & pepper',0.02);
imshow(Lena);
```

Fig. 14. Salt & Pepper noise

4.2.3 Croppinga

Cropping attack is simply cutting off parts of the image. If the algorithm is non-blind, it is better to bring back those parts from the original image for a better recovery of the message, as depicted in Fig. 15.

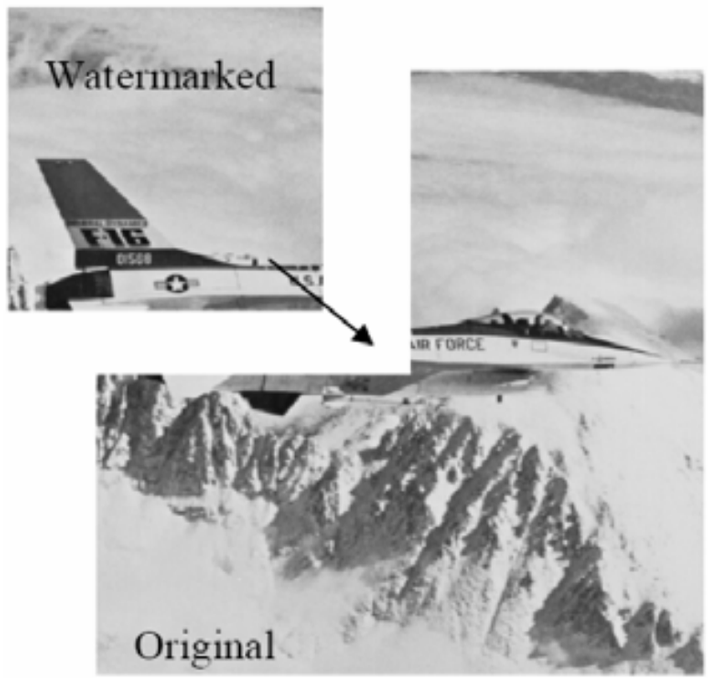


Fig. 15. Recovery from a cropping attack

4.3 Capacity

The capacity of the watermark method can be easily tested by increasing the length of the watermarking message. Any watermarking method is not capable of holding more than a certain length of message or it will endanger its imperceptibility.

5. Conclusion

In this chapter, implementation of basic digital watermarking methods in MATLAB is described. Fundamental methods in spatial, spectral, and hybrid domains are described and sample codes are given. Finally, some solutions for qualifying the watermarking method are described.

6. Acknowledgment

The author wants to thank Prof. Nasiri Avanaki who introduced the world of watermarking to students in University of Tehran.

7. References

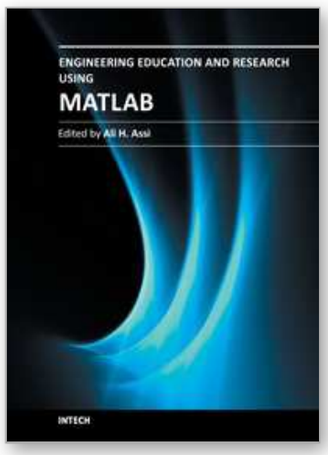
Cox, J.; Miller, M. L.; Bloom, J. A.; Fridrich J. & Kalker T. (2008). *Digital Watermarking and Steganography*, Morgan Kaufmann Pub., Elsevier Inc.

Barni M. & Bartolini F. (2004). *Watermarking Systems Engineering*, Marcel Dekker Inc., Italy

Cox, J.; Kilian, J.; Leighton F. T. & Shamoon T. (1997). Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, Vol. 6, No. 12,(December 1997), pp. 1673-1687

Vatsa, M.; Singh, R.; Noore, A.; Houck M. M. & Morris K. (2006). Robust biometric image watermarking fingerprint and face template protection. *IEICE Electronics Express, Vol. 3, No. 2, pp. 23-28*

Wang, Z.; Bovik, A. C.; Sheikh, H. R. & Simoncelli E. P. (2004).Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600-612.



Engineering Education and Research Using MATLAB

Edited by Dr. Ali Assi

ISBN 978-953-307-656-0

Hard cover, 480 pages

Publisher InTech

Published online 10, October, 2011

Published in print edition October, 2011

MATLAB is a software package used primarily in the field of engineering for signal processing, numerical data analysis, modeling, programming, simulation, and computer graphic visualization. In the last few years, it has become widely accepted as an efficient tool, and, therefore, its use has significantly increased in scientific communities and academic institutions. This book consists of 20 chapters presenting research works using MATLAB tools. Chapters include techniques for programming and developing Graphical User Interfaces (GUIs), dynamic systems, electric machines, signal and image processing, power electronics, mixed signal circuits, genetic programming, digital watermarking, control systems, time-series regression modeling, and artificial neural networks.

INTECH

open science | open minds

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen