

# Robust New Method in Frequency Domain Watermarking

David Sánchez, Agustín Orfila, Julio César Hernández, José María Sierra

Computer Security Group, Computer Science Department, Carlos III University  
28911 Leganés, Madrid, Spain

{dsanchez, adiaz, jcesar, sierra}@inf.uc3m.es

**Abstract.** This article presents a new and robust watermarking method in the frequency domain that improves over the existing ones. It is robust to JPEG compression, very configurable, simple, efficient and very easy to implement. Apart from JPEG test, it shows very good results in all tests applied.

## 1 Introduction

The development of digital technologies has made possible the transmission and storage of big multimedia information amounts. This is made at low costs, without quality losses and efficiently. This good news brings also new dangers. Multimedia creators are worried about their intellectual property rights [1,6] because, nowadays, it is not only possible but also easy to make several copies of any work [7]. A clear example of this is music stored in CD's. It is possible to make a high quality copy of a CD with a PC in less than half an hour. Furthermore the cost can be less than 5% of the original. Watermarking could be an accurate solution for protecting intellectual property rights of any kind, including images, audio, and video.

## 2 Our algorithm

Our watermarking method focuses on digital images. It works in the frequency domain (in the Discrete Cosine Transformed (DCT) domain to be exact). Working in that domain makes our method more resistant to JPEG compression attacks than if we work directly over the pixels space domain. In this sense, Cox work [3,4] is remarkable because it is one of the firsts that proposes to embed watermarks in perceptually significant components of a signal, in order to get higher robustness and to avoid quality losses.

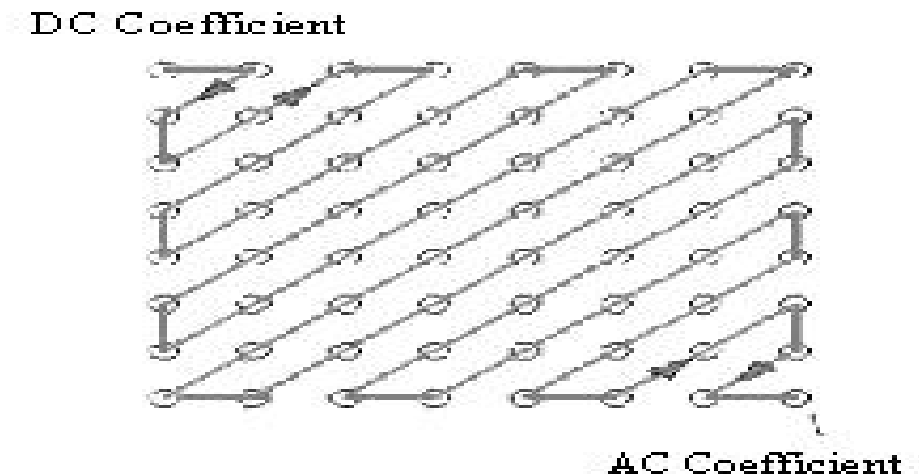
Two requirements are needed to use watermarking techniques: imperceptibility and robustness against image processing algorithms and forgery attacks [2]. Our proposal, called Sonya, tries to improve the current models. The main advantages of this new method versus famous Langelaar one [6] are:

- Better resistance to JPEG compression and to other attacks, getting a better detector response with the same quality factor  $Q$ . In order to achieve this target we use lower frequency coefficients than other approaches.
- Easy to use.
- Good execution speed

Let us explain how it works. We have an image we want to mark. We can divide it, for instance, in blocks of  $8 \times 8$  pixels. These blocks will be our  $8 \times 8$  *DCT blocks* when we transform the data to the frequency domain. The tag or watermark is made by a bit series as follows:  $L_0, L_1, \dots, L_n$ . The tag can represent information about the owner, or the input to a data base table where copyright data are related to the owner, etc.

The algorithm modifies different coefficients if the bit that is supposed to be embedded is “1” or “0”. In order to have a watermark that can resist possible attacks, low frequency coefficients are the modified ones. The way this is done is establishing its value to 0. This variation allows us to identify the marked coefficients later because under a threshold next to 0 ( $U_{\text{detection}}$ ) they are considered marked. We should choose a marking threshold ( $U_{\text{marked}}$ ) with the feature that no  $8 \times 8$  *DCT block* with a value over it is marked. Only those  $8 \times 8$  *DCT blocks* under this threshold are marked.

One of the main features of  $8 \times 8$  *DCT blocks* is that their frequencies are ordered the way the following figure shows:




**Figure 1:** Ordination of  $8 \times 8$  block of *DCT* coefficients. AC is the coefficient associated with the highest frequency and DC with the lowest

## 2.1 Marking Process

The marking process works this way:

- If the tag-bit we want to embed is “1” the shadowed coefficients in the following figure (*Diagonal\_one*), shown as an example, are modified establishing its value to 0.



583.5	-132	-2.1	-12.7	11	-2.3	-5.1	-4.5
411.9	-61.9	-33.2	-1.8	0.4	-7.2	-8.7	4.9
115.6	126.3	-47.3	5.2	-2.4	-4.2	-5.7	4.3
-43.9	135.2	44.5	-15.1	10.2	-6.7	8.3	7.8
-21	5.5	108.8	-3.7	5.5	6.8	8	-6.2
1.9	-46.7	57.9	50.2	-8.4	22.6	1.8	1.8
-10.3	-17.9	-30.7	48.3	28.9	13.8	-5.2	-8.6
-14.9	0.4	-30.3	6.7	50.7	-8.9	4.7	-4.2

**Figure 2:** Coefficients that has to be modified in order to embed  $L_i = 1$

If we establish the value  $Diagonal\_one = |C(0,1)| + |C(1,0)|$ ,

to modify the coefficients it must be true that:

$$Diagonal\_one \leq U\_marked \rightarrow C(0,1) = 0 \text{ and } C(1,0) = 0.$$

Else, we should go ahead with next  $8 \times 8$  DCT block.

Guessing an  $U\_marked = 600$

$$Diagonal\_one = 132 + 411,9 = 543,9 < 600$$

The sum of the absolute values of the 2 coefficients (*Diagonal\_one*) is lower than  $U\_marked$ , so we mark:

Before marking  $C(0,1) = -132 \rightarrow$  After marking  $C(0,1) = 0$ .

Before marking  $C(1,0) = 411,9 \rightarrow$  After marking  $C(1,0) = 0$ .

- If the bit of the tag we like to embed is “0” we modify the shadowed coefficients in the following figure (*Diagonal\_zero*) establishing its value to 0.

583.5	-132	-2.1	-12.7	11	-2.3	-5.1	-4.5
411.9	-61.9	-33.2	-1.8	0.4	-7.2	-8.7	4.9
115.6	126.3	-47.3	5.2	-2.4	-4.2	-5.7	4.3
-43.9	135.2	44.5	-15.1	10.2	-6.7	8.3	7.8
-21	5.5	108.8	-3.7	5.5	6.8	8	-6.2
1.9	-46.7	57.9	50.2	-8.4	22.6	1.8	1.8
-10.3	-17.9	-30.7	48.3	28.9	13.8	-5.2	-8.6
-14.9	0.4	-30.3	6.7	50.7	-8.9	4.7	-4.2

**Figure 3:** Coefficients that have to be modified if we want to embed  $L_i = 0$

If we establish the value  $Diagonal\_zero = |C(1,1)| + |C(2,0)|$ , in order to modify the shadowed coefficients we have to proceed this way:

If  $Diagonal\_zero \leq U\_marked \rightarrow C(1,1) = 0$  and  $C(2,0) = 0$ .

Else, we go ahead with the next 8x8 *DCT block*.

Guessing the same  $U\_marked = 600$  as we did in the last case:

$$|C(1,1)| + |C(2,0)| = 61.9 + 115.6 = 177,5 < 600$$

The sum of the absolute values of both coefficients is lower than  $U\_marked$ , so we proceed to mark:

Before marking  $C(1,1) = -61,9 \rightarrow$  After marking  $C(1,1) = 0$ .

Before marking  $C(2,0) = 115,6 \rightarrow$  After marking  $C(2,0) = 0$ .

## 2.2 Detection Process

We choose an  $U\_detection$  next to 0 under which the  $8 \times 8$  DCT block is considered marked. Now we have two possibilities:

If  $Diagonal\_one \leq Diagonal\_zero$  and  $Diagonal\_one \leq U\_detection$

→  $L_i(\text{bit to extract}) = 1$

If  $Diagonal\_zero < Diagonal\_one$  and  $Diagonal\_zero \leq U\_detection$

→  $L_i(\text{bit to extract}) = 0.$

We have only a problem to solve. Let us observe the following  $8 \times 8$  DCT block:

583.5	-132	-2.1	-12.7	11	-2.3	-5.1	-4.5
411.9	-2,7	-33.2	-1.8	0.4	-7.2	-8.7	4.9
1,3	126.3	-47.3	5.2	-2.4	-4.2	-5.7	4.3
-43.9	135.2	44.5	-15.1	10.2	-6.7	8.3	7.8
-21	5.5	108.8	-3.7	5.5	6.8	8	-6.2
1.9	-46.7	57.9	50.2	-8.4	22.6	1.8	1.8
-10.3	-17.9	-30.7	48.3	28.9	13.8	-5.2	-8.6
-14.9	0.4	-30.3	6.7	50.7	-8.9	4.7	-4.2

**Figure 4:** DCT problematic coefficients

If we like to embed a 1, we should proceed as we did in the example of figure 2 modifying  $C(0,1)$  and  $C(1,1)$ . The problem is that after opening the image for marking it, apply the DCT to it and the IDCT (inverse DCT), the values of the marked coefficients are not 0. They have been softly disturbed to a value next to zero. During detection process, if the sum of  $|C(1,1)| + |C(2,0)|$  has a high value, much higher than 0, there is no problem, but what happens if the sum of  $|C(1,1)| + |C(2,0)|$  has a value very close to 0?

The values of  $Diagonal\_one$  and  $Diagonal\_zero$  after marking will be very low, close to zero and very similar between them. This produces an ambiguous situation while detection, because it is perfectly possible to detect a 1 when a 0 was embedded and the other way round.

To solve the problem, we have raised the value of the non-modified coefficient pair in the marking process, if they are under  $U\_marked$ . We define an *increment*, and we have two possible situations:

If we mark *Diagonal\_one* and the value of *Diagonal\_zero*  $\leq U\_marked$   
 $\rightarrow \quad Diagonal\_zero = Diagonal\_zero + increment.$

If we mark *Diagonal\_zero* and the value of *Diagonal\_one*  $\leq U\_marked$   
 $\rightarrow \quad Diagonal\_one = Diagonal\_one + increment.$

Increasing the value of the non-marked diagonal when it is under the marking threshold, it is guaranteed that it goes away from 0 in the detection process, decreasing the probability of wrong positive results.

Let us show the marking algorithm more in detail.

### 2.3 Marking Algorithm Revisited

The steps of the algorithm are the following:

- We establish the values of these parameters:
  - $U\_marked$  : limit under which we proceed to mark an 8x8 *DCT* block.
  - *increment* : quantity that is added to the non marked coefficients under  $U\_marked$ .
- The counter  $i$  of the 8x8 *DCT blocks* is initialized to 0. The counter  $j$  of tag-bits is initialized to 0.
- An 8x8 *DCT block*,  $b_i$  is selected from the image  $I$  in order to embed  $L_j$ .
- If  $L_j$  is 1,

If  $|C(0,I)| + |C(1,0)| \leq U\_marked \rightarrow C(0,I) = 0$  and  $C(1,0) = 0$

If  $|C(1,I)| + |C(2,0)| \leq U\_marked \rightarrow$

If  $C(1,I) \geq 0 \rightarrow C(1,I) = C(1,I) + increment$

else,  $C(1,I) = C(1,I) - increment$

else while there are *DCT* 8x8 blocks  $i$  is increased and we go again to step 3.

- If  $L_j$  is 0,

If  $|C(1,1)| + |C(2,0)| \leq U\_marked \rightarrow C(1,1) = 0$  and  $C(2,0) = 0$

If  $|C(0,1)| + |C(1,0)| \leq U\_marked \rightarrow$

If  $C(0,1) \geq 0 \rightarrow C(0,1) = C(0,1) + increment$

else  $C(0,1) = C(0,1) - increment$

else while there are *DCT* 8x8 blocks *i* is increased and we go again to step 3.

- While there are *DCT* 8x8 blocks *j* and *i* are increased . We go again to step3.
- There are no more 8x8 *DCT* blocks to mark  $\rightarrow$  End.

## 2.4 Tag Extraction Algorithm

The steps of the extraction algorithm are the following:

1. First we establish some parameters:

- *U\_detection* : an 8x8 *DCT block* is considered marked under this limit.
- *T*: is the difference between the percentage of detected bits in a truly embedded mark and the percentage detected in one or more that are not. Normally the detector response for false positives is under 20, that is why we can use a value for *T* = 20 or 25.
- *False\_password\_0 ... False\_password\_n* : we establish the value or values for the false passwords. They generate marks that will not be on the false images.

2. We initialize the number of detected bits, *detected\_bits* = 0. The counter *i* of 8x8 *DCT* blocks is initialized to 0. The counter of tag-bit *j* is initialized to 0.

3. An 8x8 *DCT block* , *b<sub>i</sub>* is selected from the image *I* to extract *L<sub>j</sub>*.

4. If  $|C(0,1)| + |C(1,0)| \leq |C(1,1)| + |C(2,0)|$

and  $|C(0,1)| + |C(1,0)| \leq U\_detection \rightarrow L_j = 1$  *detected\_bits* = *detected\_bits* + 1

We increase *j*.

Else,

If  $|C(1,1)| + |C(2,0)| < |C(0,1)| + |C(1,0)|$

and  $|C(1,1)| + |C(2,0)| \leq U\_detection \rightarrow L_j = 0$  *detected\_bits* = *detected\_bits* + 1.

We increase j.

5. While there are  $8 \times 8$  DCT blocks,  $i$  is increased and we go to point 3.

- We have finished with  $8 \times 8$  DCT blocks. Steps from 1..5 are executed with all the wrong passwords getting the average bit number. Steps from 1..5 are also executed for the password we want to detect the mark with and we keep the value in *detected*.
- If  $100 - ((average \times 100) / detected) \geq T \rightarrow$  Watermark detected !  
else  $\rightarrow$  Watermark not detected.
- If the condition is true and the mark is detected then the difference between the number of detected bits on the tag extracted and the not embedded mark average is higher than  $T$  %. Normally, the response to a mark that is really embedded is about 50 % and the response to a mark that is not is under 20. That is why a threshold  $T$  equal or higher than 20 can be used in the detection process.

### 3 Results

We have analyzed three different images: lenna.ppm, sabatini.ppm and mandril.ppm.

Their features are different (colour scales, high frequency zones, defined borders...) so it is very useful to compare the results over them. We have figures for all the results but we are forced not to show most of them because the space restrictions of this article. Let us take only a quick look on the results:

#### 3.1 Invisibility of the watermarking

$U\_marked$  guarantees the invisibility of the watermark. If we choose this limit too high the watermark would be visible, so we have to look for a limit that adapts to the image features.  *$U\_marked$*  should allow us to modify many low frequency coefficients guaranteeing the invisibility of the mark. Other parameter that has an important influence in the visibility of the mark is *increment*. In this case a higher value of *increment* produces a higher distortion in certain coefficients making the



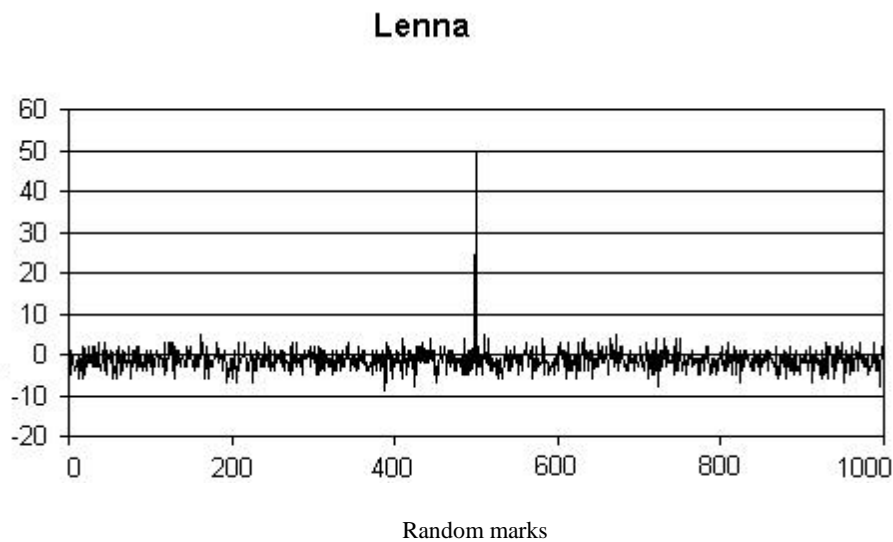


**Figure 5:** *Lenna* image marked with  $U_{\text{marked}} = 15$ ,  $\text{increment} = 15$

mark more visible. As we can see in the image above, using  $U_{\text{marked}} = 15$ ,  $\text{increment} = 15$  we get a mark completely invisible. We have gotten similar results for the other images.

### **3.2 Mark Uniqueness**

Uniqueness test is passed easily. Testing 1000 random marks we can observe that only one is really embedded. The detector's response to this mark stands out the rest. The results are shown in the following figure.



**Figure 6:** Detector Response to 1000 random watermarks, only 1 of them embedded

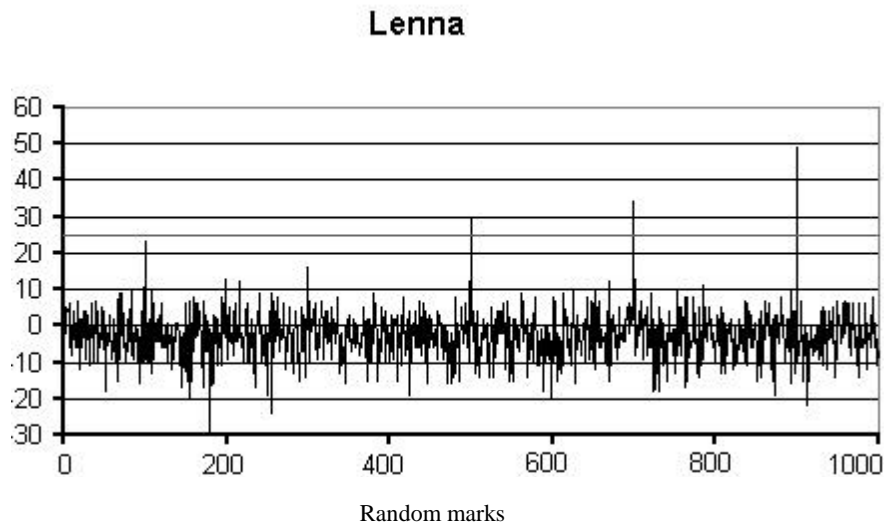
The used parameters for *Lenna* image were:  $U\_marked = 15$ ,  $Increment = 15$ ,  $U\_detection = 5$ ,  $T = 25$ . In this case the detector response is clear. While non-embedded marks have a response between 10 and  $-10$ , the response to the embedded mark is 50.

For the image *Sabatini*, the chosen parameters are  $U\_marked = 20$ ,  $Increment = 20$ ,  $U\_detection = 5$ ,  $T = 25$ . In the case of the embedded mark, 376 bits of 377 embedded bits are detected. In the rest of random marks, the response moves around 188 bits.

The results for the image *Mandrill* are also satisfactory. The parameters used with *Mandrill* are  $U\_marked = 20$ ,  $Increment = 20$ ,  $U\_detection = 5$

### 3.3 Multiple Watermark Detection

The Sonya algorithm has the same problems Langelaar's has. The process of watermarking applied many times modifies the same *DCT* coefficients, so a mark alters the coefficients that other mark has just modified. This makes very difficult the detection process. The solution is to choose different coefficients for each mark we want to embed, using other diagonals.

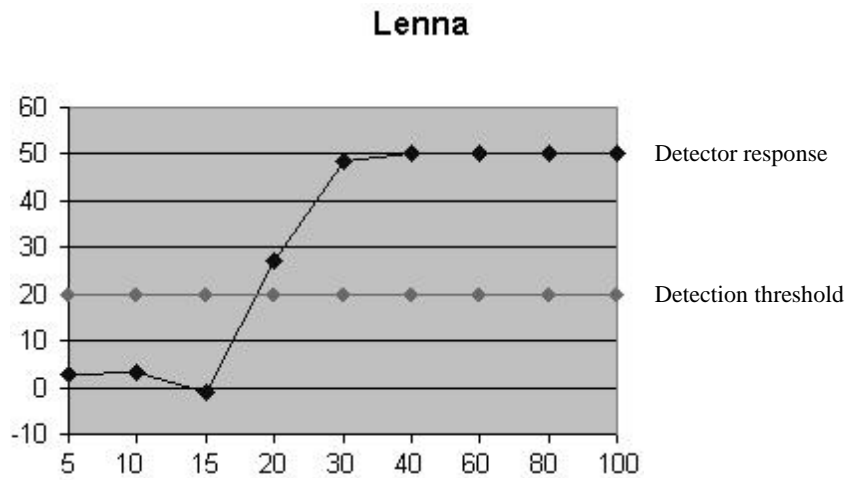


**Figure 7:** Detector response to 1000 watermarks, 5 embedded

- The response to 5 embedded marks is higher than the response to the rest of random marks. Although this is true, we must remark that there is one without a clear value because we need a very low threshold  $T = 15$  for its detection. For instance a threshold  $T = 25$  would only detect 3 of the 5 marks. Therefore we can say that embedding more than five images can get us into trouble during the detection process.
- For *Sabatini* image only 3 of the 5 five embedded marks are over a threshold  $T = 25$ . For lower  $T$  we can detect 5 embedded marks but also some that are not embedded.
- For *Mandrill* image the five embedded marks are over the threshold  $T = 25$ .

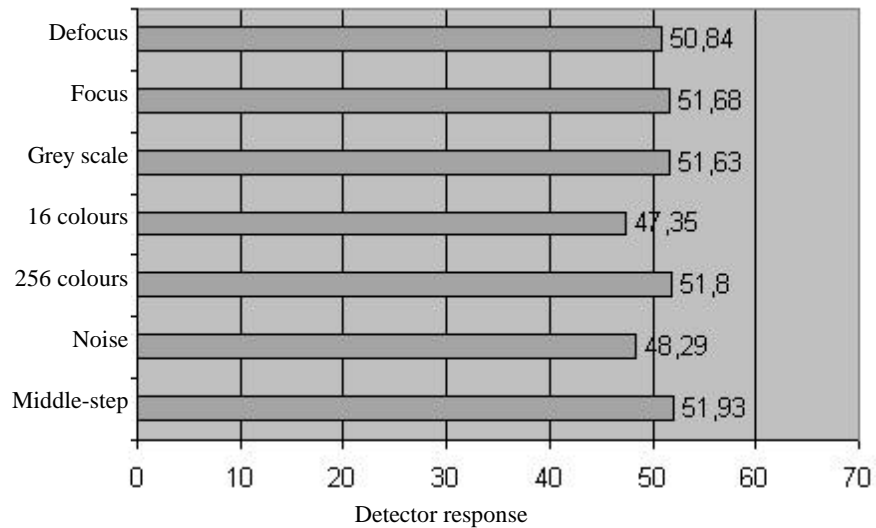
### 3.4 JPEG Compression

The results allow us to detect a watermark with quite low compression levels as it is shown in the figure. We used a threshold ( $T = 20$ ) quite safe in the preceding image. This means that marks with a response over 20% are detected. In this case the mark is detected even using a quality factor  $Q = 20$ . For lower quality factors, the detector response is too low ought to the presence of false positive and false negative results after the distortion that compression makes. Watching the figure, the mark is detected for all quality factors  $Q$  which blue line is over the pink one. In the case of *Sabatini* and *Mandrill* images there are no doubts with quality factors  $Q$  over 15.



**Figure 8:** Detector response against JPEG compression

### 3.5 Strength Against Filters



**Figure 9:** Lenna detector response after filter application

We have obtained excellent results in this test. The detector response is very high for the three images and the mark is detected in all cases with a threshold  $T = 30$ ,

excluding a negative case in 16 colour reduction in *Mandrill* image where the response is relatively low, with a value of 17,72. In order to solve this problem we can mark the blue channel of the image or look for stronger marking parameters. Let us show Lenna results:

### **3.5 Rotation, Scaled and Rescaled**

For the rotation, if we know the spinning angle, when we revert the operation the mark is clearly detected, getting the same results in the detection process as we get with the original marked image.

The rescaling test presents higher difficulties to detect the mark. In this case, the image loses quality. With our images when we reduce the images and we turn to the original size, the results have been the following:

For *Lenna* image the mark is lost after reducing it to a 60% of its original size. Nevertheless the image loses quality notably.

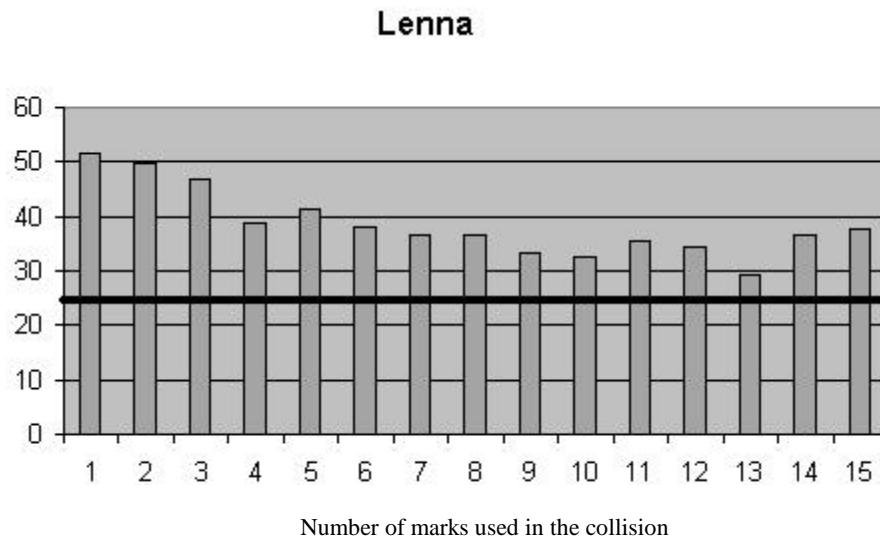
For *Sabatini* image we get excellent results. The mark is not lost until we reduce it to 30 %. But if we do that, the image remains totally deteriorated.

In all cases the image resist to an 80% of the original reduction at least.

For the cutting out test the mark is detected with a threshold over  $T = 20$  getting a response in the detector of 21,67 in *Mandrill* case. Of course the results are completely conditioned to the cut size. As bigger the cut is, smaller is the detector response.

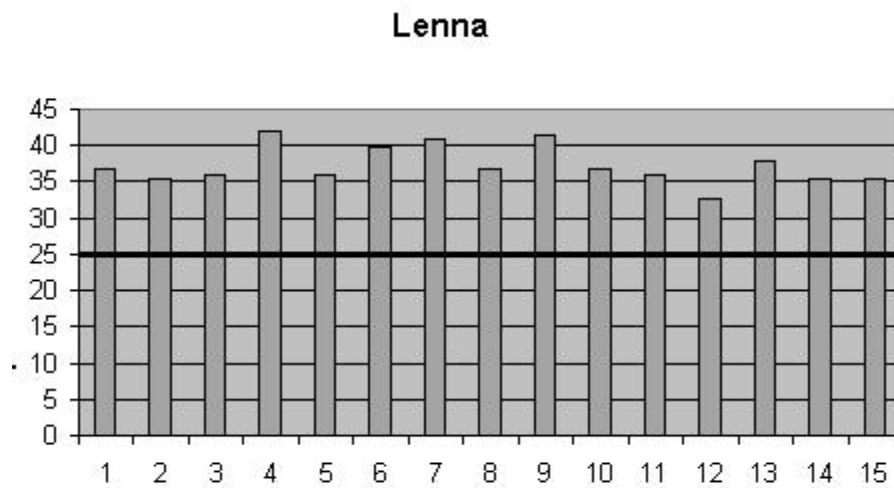
### **3.5 Watermark Collision**

We have marked 15 images with different passwords for this test. The detector response has been the following.



**Figure 10:** Detector response to different one-mark collisions

As shown, the mark is detected over a threshold  $T = 25$  for all the tested cases.



**Figure 11:** Detector response to each collision watermark

Also, when we used 15 marked images in the collision, all of them were detected.

In *Sabatini* case only a response (with value 20) is under the threshold  $T = 25$ . This means we should establish  $T = 20$ . This threshold is not as reliable as the other but it can be useful if we look at the probability of false positive results at the uniqueness test. Three marks remain under  $T = 25$ . Two of them under 20 and the other is not detected because has a 0 response. The solution is looking for other marking parameters. In the *Mandrill* case the mark is detected for  $T = 25$  in all cases.

Summing up we can say that our method gets better results for bigger images because they use to have a higher number of marking coefficients.

### 3.6 Stirmark

Although the response is quite good in most of Stirmark attacks, there are cases in which we can remove the watermark making the detector response to descend to 10. Nevertheless, the difference between the marked image and the one that is not is clearly visible, as it is shown in the following figures.



**Figure 12:** *Lenna* image marked before Stirmark application



**Figure 13:** *Lenna* image marked after Stirmark application

## 4 Conclusions

The importance of watermarking for protecting the intellectual property of multimedia data is clear. New ideas and its development appear to be essential. No current method is free from weaknesses. So we have to work in order to improve them day-by-day. Furthermore, the way data is transferred by the Internet changes very fast and the methods should be strong enough in order to adapt to these changes.

The main advantages of the exposed method are:

- Easy implementation: The marking and detection code together has less than 300 lines.
- Good efficiency: The implementation of the algorithm is very fast. As an example we can say that for a 535 KB image, the marking process consumes less than a second. (Tests made on a Pentium III, 800 MHz and 128 MB RAM).
- The code is written in standard ANSI C providing portability for Linux and Windows platforms.
- A very important difference versus other methods is that we use *blind techniques*. The original image is not required in the detection process. This is very important because we do not always have the original image. The flexibility of the presented method allows using it as a non-blind method. In this case the strength of Sonya method is still higher and we can use *Trusted*



*Third Parties (TTP)* for the custody of the original image, passwords and data related to the image owner.

- Other remarkable point is that our watermarks are *public* or *recoverable* because we can extract the mark information bit by bit. This feature is very important because we can know the data owner just in the moment we extract the mark. We do not have to look in a database. Again the flexibility of this method allows a *private* implementation in order to just detect the mark in the image. In both cases if the attacker wants to *recover/detect* the mark has to own the password used in the marking process.
- With no doubts the main achievement of our method is the good response to all the tests done.

Let us summarize our results:

- The watermark invisibility is guaranteed by the marking parameters. According to our interests we can establish if a mark will be visible or not. Obviously a visible mark that destroys the image is not interesting, but sometimes it is better a watermark slightly visible and almost imperceptible that provides us higher strength.
- Considering the JPEG compression test as one of the main ones, the results are excellent because the watermark embedded is even detected under a compression factor  $Q = 30$ .
- The uniqueness test also shows great results and the watermark is clearly detected using 1000 random marks. Therefore the false positive probability is almost zero in the tests done. This test allows us to guess which is the best value for the detection threshold. Consequently over a certain detection threshold we are completely sure that the detected mark has been really embedded, and not detected marks have not.
- Some modifications to our method are necessary in order to detect multiple watermarks. This happens because marking an image many times often needs to modify the same DCT coefficients.
- Filter application does not seem a serious threat because a strong filter produces distortion in the image when removing the mark. We have used standard filters that any attacker can use with programs that modify images.
- Rotation presents no problem if we know the spinning angle. In this case we detect the mark in all the tests we have made, after undoing the operation.

- The scale and cutting out processes highly depend on the image. They can destroy a mark but they distort the image. In these circumstances what we have to value is if the quality of the image is accurate.
- Watermark collision has the same problems multiple marks have, but nevertheless the results are good. If we realize that an attacker must have many marked images in order to remove the watermark, the results appear to be even better.
- The more difficult test in watermarking use to be the Stirmark one. We have to say that Stirmark removes our watermark but the resultant image is distorted and clearly different from the original.

We have shown that our method improves over the existing ones in what is related to the watermark of digital images. Anyway, there is still quite a lot of work to do in order to get even better results, and to apply similar ideas and philosophy to video and audio watermarking.

## References

1. M. Barni, F. Bartolini, V. Cappellini, A. Piva. *Robust watermarking of still images for copyright protection*. Proceedings 13<sup>th</sup> International Conference on Digital Signal Processing DSP97, Vol. 2, pp. 499-502, 1997.
2. F. Bartolini, M. Barni, V. Cappellini, A. Piva. *Mask building for perceptually hiding frequency embedded watermarks*. Proceedings of 5<sup>th</sup> IEEE International Conference on Image Processing ICIP'98, Vol. I, pp. 450-454, 1998.
3. I. J. Cox, J. Kilian, T. Leighton, T. Shamoan. *Secure, Robust Watermark for Multimedia*. Proceedings of First International Workshop of Information Hiding, University of Cambridge, May 1996.
4. I. J. Cox, J. Kilian, T. Leighton, T. Shamoan. *Secure Spread Spectrum Watermarking for Multimedia*. IEEE Transactions on Image Processing, vol. 6, no. 12, pp. 1673-1687, 1997.
5. M. Kutter. *Watermarking resisting to translation, rotation, and scaling*.
6. G.C. Langelaar, J.C.A. Van der Lube, J. Biemond, *Copy Protection for Multimedia Data based on Labeling Techniques*, 17<sup>th</sup> Symposium on Information Theory in the Benelux, Enschede, The Netherlands, May 1996.
7. A. Piva, M. Barni, F. Bartolini. *Copyright protection of digital images by means of frequency domain watermarking*. Mathematics of Data/Image Coding, Compression, and Encryption, Proceedings of SPIE Vol. 3456, pp. 25-35, 1998.

8. A. Piva, M. Barni, F. Bartolini, V. Cappellini. *Threshold Selection for Correlation-Based Watermark Detection*. Proceedings of COST 254 Workshop on Intelligent Communications, pp. 67-72, 1998