Content Based Watermarking for Securing Color Images

Gregory Lo-Varco, William Puech, Michel Dumas

Content Based Watermarking for Securing Color Images

G. Lo-varco[1] W. Puech[2] M. Dumas[1]

[1] CEM2 Laboratory, UMR CNRS 5507, University of Montpellier II

Pl. Gabriel Péri, 30021 Nîmes Cedex 1, France

[2] LIRMM Laboratory, UMR CNRS 5506, University of Montpellier II

161, rue Ada, 34392 Montpellier Cedex 5, France

lovarco@cem2.univ-montp2.fr puech@lirmm.fr dumas@univ-montp2.fr

Abstract

In this paper a region based color image watermarking algorithm is presented. The objective of the method is to embed a particular message in each region of interest (ROI) in the image. The embedding message has to be detected after image manipulations such as cropping, rotation and color JPEG compression. As a basis for the watermarking, a segmentation algorithm is used. Based on the extracted regions, characteristic features are estimated by using shape information. The embedding message is synchronized with each ROI on each color component Y, Cr and Cb. To embed information non-rounded DCT coefficients are watermarked. Experimental results show the performance of the algorithm against spatial and frequential attacks.

Key Words

Robust objects labelling, Color DCT-based watermarking method, Content-based watermarking method, Non-rounded DCT coefficients, Region of interest (ROI), Principal component analysis (PCA).

1 Introduction

Several techniques have been proposed in the literature to embed information in digital images [1, 6, 7, 16]. These techniques provide various degrees of robustness. Watermarking applications include copyright protection, authentication, embedded and hidden information. Firstly, watermarking systems that are intended for copyright protection require a very high degree of robustness. Then, watermarking process for authentication belong to the fragile class of schemes. Slightest change in the image completely destroys the mark. Finally watermarking for embedding information requires resistance against moderate level of modification due to routine image processing such as compression or cropping. Our color watermarking method belongs to this last group of watermarking systems. Furthermore, in our application, the length of embedded data can relatively be important. In each region of interest (ROI), we embed a number of bits which fluctuates between 5 and 200 bits according to the ROI size.

The current paper presents a technique for color watermarking of images [4], based on non-rounded DCT-coefficients [3, 5, 10]. The hidden data are synchronized with the ROIs. This method mainly protects against interaction with visual content, geometrical manipulations [11, 12, 14] and JPEG compression. The paper is organised as follows. In Section 2 we introduce the ROIs and the Principal Component Analysis (PCA) is presented. Section 3 describes our color DCT-based watermarking method. In Section 4, we present and analyze some concluding results.

2 ROIs extraction and descriptors

Section 2.1, we present the ROIs detection by segmentation. From these ROIs, a binary mask is obtained. In Section 2.2, each ROI is analyzed and defined by several descriptors. Section 2.3, we describe the synchronization between hidden data and image.

2.1 Content-mask obtention

2.1.1 Segmentation process

Our objective is to use image regions, which may not completely correspond to an object, but are generated by a simple segmentation algorithm [8]. The image content is described by all unconnected ROIs. These ROIs are required to have a minimum size to be considered for watermarking, otherwise they are merged with neighbouring regions. We took the minimum size of a region to be 3000 pixels in total area so as to prevent the watermark relying on too small area for data hiding.

First, the color image is converted to grey-level format. Then a region-growing algorithm is used. From each pixel, every adjacent pixel is added to the same region if their difference is minor. The strict definition of a minor difference is left to the discretion of the user, but usually we consider that two pixels are similar if the color difference is less than 20 grey levels (on a scale of 256 grey levels). Each ROI consists of a subset of pixels forming a shaped area of the image. To illustrate this process, segmentation is applied on the original image “Fish”, Figure 5.a, and we obtain the associated content-mask, illustrated Figure 5.b.

2.1.2 Works on the ROIs

After segmentation, the image is a set of convex areas. Each area has an outline made of boundary pixels. These pixels, by definition, would move to the ROI when there are geometrical modifications like rotations. Consequently, to increase the robustness, these pixels are not used to define the ROI.

Indeed, the ROI is reduced by a set of erosion and dilatation [15]. We loose information about the outline but we increase the robustness. If a deformation completely modifies the ROI outline, the erosion and dilatation decrease its effect, and we obtain the same simplified shape of ROI.

2.1.3 Robust labelling of ROIs

After the reduction of ROIs, the image is a set of simplified and regular shapes. Each ROI corresponds to an area in the binary mask. Now the region-labelling algorithm indicates all the ROIs. This method is based on the Rosenfeld labelling operator [13]. Each pixel obtains a label, which depends on two characteristics: the intensity of pixel and its local neighbourhood. This technique is applied here on 4-connexity. This type of parallel labelling methods is used because the image is read only one time. After an initialization, a table of equivalences between the labels is built. Finally, each pixel of ROI gets the specific ROI label L_{ROI}. After ROI extraction, an analysis is necessary to base correctly the hiding data.

2.2 ROI features

In an attempt to detect the embedded data after geometrical modifications, we insert information in a frame of reference depending on each ROI shape. To that, we define three features of ROI: the size descriptor, the position descriptor and last the orientation and shape descriptor.

2.2.1 A Size descriptor: the ROI area

The ROI size $S(ROI)$ is the first calculated parameter. It is represented by the number of pixels which have the same ROI label L_{ROI}. Accordingly, a first classification of ROIs is possible. A quantization is made with a standard size of ROI S_s. We finally obtain a factor of size:

$$F_S(ROI) = \frac{S(ROI)}{S_s}. \quad (1)$$

S_s generally is equal to 10% of the original image size i.e. 15000 pixels with an image consisting of an array 400×350 of pixels like "Fish" image. In this way, the detection of hidden data should be possible after a scaling operation.

2.2.2 A Position descriptor: the spatial centre

To indicate the position of ROIs in the image, we use the spatial centre G_{ROI}. The moments of first degree noted $\mu_\imath(ROI)$ and $\mu_\jmath(ROI)$ precisely locate this singular point of ROI. The average $\imath$-coordinate of ROI pixels is $\mu_\imath(ROI)$. The average $\jmath$-coordinate of ROI pixels is $\mu_\jmath(ROI)$. To calculate them, we have:

$$\begin{cases} \mu_\imath(ROI) = \dfrac{1}{S(ROI)} \displaystyle\sum_{k=0}^{S(ROI)-1} \imath(k) \\ \\ \mu_\jmath(ROI) = \dfrac{1}{S(ROI)} \displaystyle\sum_{k=0}^{S(ROI)-1} \jmath(k), \end{cases} \tag{2}$$

where $\imath(k)$ and $\jmath(k)$ respectively are the vertical and horizontal coordinates of the pixel k. We finally obtain:

$$G_{ROI} = G[\mu_\imath(ROI), \mu_\jmath(ROI)]. \tag{3}$$

This centroid G_{ROI} is the origin of a specific frame of reference. This frame will be used to embed the data.

2.2.3 An orientation and shape descriptor: the principal component analysis

Our embedding scheme uses a frame of reference depending on the ROI shape. To build this frame, one point and two directions are necessary. The point G_{ROI} has been calculated in the previous section. Now we have to determinate these two specific directions of the ROI. We employ the PCA in an attempt to obtain the principal directions of the ROI. To determinate them, the ROI moments of the second degree are calculated. We obtain the horizontal variance $V_x(ROI)$, the vertical variance $V_y(ROI)$, and

the covariance $V_{xy}(ROI)$ given by:

$$\begin{cases} V_x(ROI) &= \dfrac{1}{S(ROI)} \displaystyle\sum_{k=0}^{S(ROI)-1} (\imath(k) - \mu_\imath(ROI))^2 \\ V_y(ROI) &= \dfrac{1}{S(ROI)} \displaystyle\sum_{k=0}^{S(ROI)-1} (\jmath(k) - \mu_\jmath(ROI))^2 \\ V_{xy}(ROI) &= \dfrac{1}{S(ROI)} \displaystyle\sum_{k=0}^{S(ROI)-1} [\imath(k) - \mu_\imath(ROI)][\jmath(k) - \mu_\jmath(ROI)]. \end{cases} \tag{4}$$

This set of coefficients is represented by a 2×2 covariance matrix $C_0(ROI)$ noted:

$$C_0(ROI) = \begin{bmatrix} V_x(ROI) & V_{xy}(ROI) \\ V_{xy}(ROI) & V_y(ROI) \end{bmatrix}. \tag{5}$$

An analysis of this matrix gives two eigenvalues $\lambda_1(ROI)$ and $\lambda_2(ROI)$ and two associated eigenvectors $\vec{V_1}(ROI)$ and $\vec{V_2}(ROI)$ [2]. These vectors denote directions providing maximal variance of the ROI. In this context, these couples $\{\lambda, \vec{V}\}$ represent the major and minor axes of the ROI. These two axes are illustrated in Figure 1. Associated with the centroid G_{ROI}, they form a frame of reference adapted to receive the hidden data. The hidden data consequently are oriented and synchronized according to each frame of reference.

2.3 Synchronization of hidden data according to the ROIs

Our watermarking method uses unitary blocks of N pixels. These blocks are differently definite according to the ROI shape and the ROI orientation. Moreover they are built in a specific order to keep the information integrity.

2.3.1 Specific definition of ROI unitary blocks

Our watermarking method implies several constraints on the unitary blocks. Firstly, the block size is normalized in order to use an unique and fast detection process. Then, the block is generated according

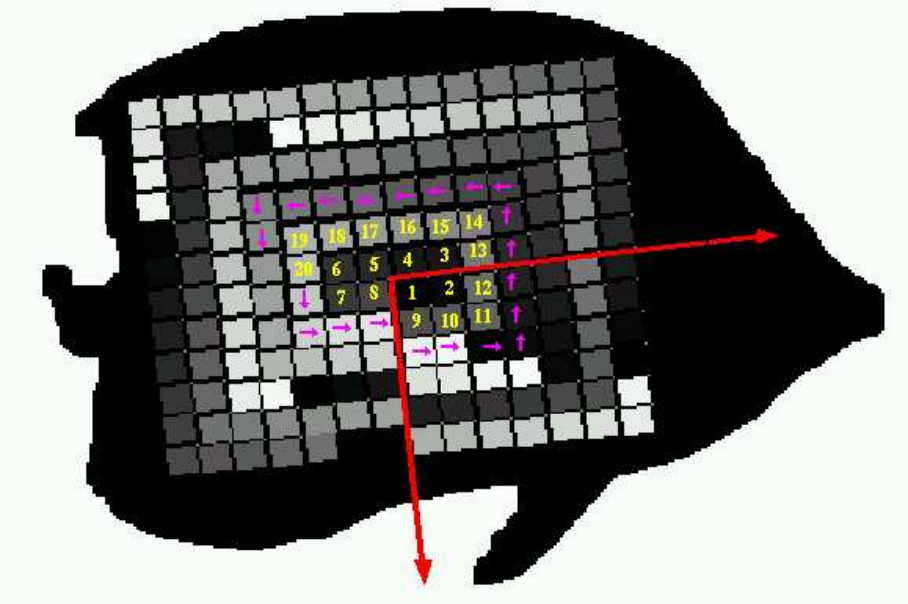


Figure 1: The appropriate frame of reference with the building order of blocks.

to the ROI shape. The eigenvectors $\vec{V_1}(ROI)$ and $\vec{V_2}(ROI)$ are the two oriented sides of block. In this way, the block is built with all pixels $p_b(x, y)$ such that:

$$\begin{cases} 0 \leq x < ||\vec{V_1}(ROI)|| \\ 0 \leq y < ||\vec{V_2}(ROI)|| \end{cases}, \text{ and } ||\vec{V_1}(ROI)|| = ||\vec{V_2}(ROI)|| = \sqrt{N}, \tag{6}$$

where $||\vec{u}||$ is the norm of vector $\vec{u}$ and (x, y) the coordinates of pixels in the ROI reference frame. The block is also a set of N 2D variables $p_b(x, y)$ (pixels of block) with $0 \leq b < N$. The unitary block is specifically oriented and adapted to each ROI.

To obtain the coordinates $(\imath, \jmath)$ of pixels p_b in the image, we change the frame of reference and we obtain:

$$\begin{cases} \imath = x + \imath_G = x + \mu_\imath(ROI) \\ \jmath = y + \jmath_G = y + \mu_\jmath(ROI). \end{cases} \tag{7}$$

2.3.2 Building order of watermarking block

To embed data, an order is followed. We use one more time the couples $\{\lambda, \vec{V}\}$ and the centroid G_{ROI} to determinate the insertion path, noted Ip_{ROI}. Ip_{ROI} is a tidy set of m starting pixels sp_k with $0 \leq k < m$ and m the number of embedded bits in the ROI. Each point sp_k is the origin of a unitary

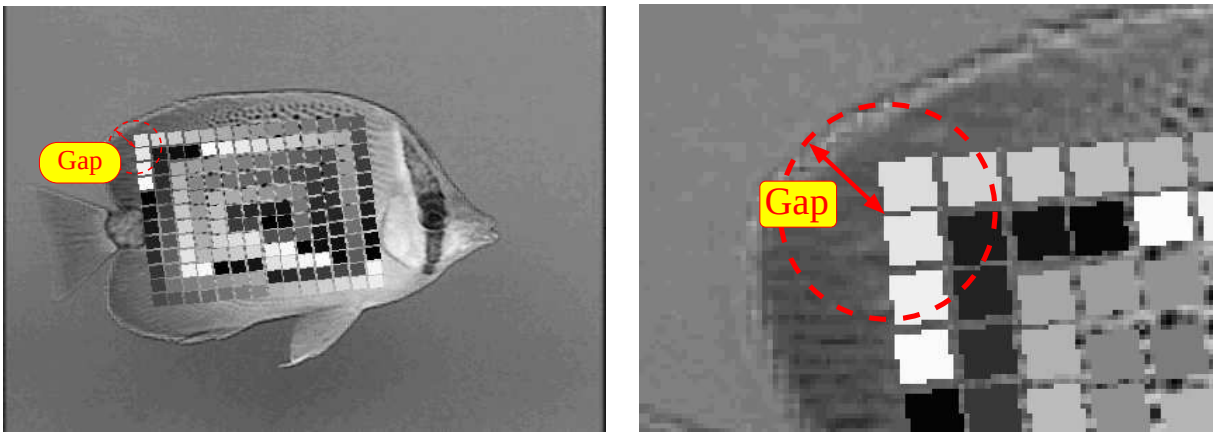


Figure 2: a) Minimal gap between the outline and the blocks, b) Enlarging of the outline.

block. The coordinates (x_{sp_k}, y_{sp_k}) of these pixels sp_k are added to each $p_b(x, y)$. We also obtain a set of m tidy blocks built with N pixels, illustrated Figure 5.c. In other hand, Ip_{ROI} has an eccentric growth. The first pixels sp_k are near the centre of gravity G_{ROI} and the last pixels are near the ROI outline. A minimal gap is kept between the outline and the blocks. It corresponds to the minimal distance between the outline and any starting pixel sp_k. This gap is left to the discretion of user but for our tests we considered a gap equal to $\frac{\sqrt{N}}{2}$, i.e. the major part of block is in the ROI. The development of Ip_{ROI} is illustrated in Figure 1. The gap is described Figures 2.a and 2.b.

In this section, we have described how to use the content of color image. The following section presents in details how to embed data by watermarking.

3 Color DCT-Based Watermarking method

To detect the data after JPEG compression, we have created a method that is adapted to the main stages of JPEG algorithm [17]. In this way, the embedded data is not eliminated during compression process. The Section 3.1 shows how to use the color conversion to increase the redundancy. Then, in Section 3.2, we present the interest of DCT coefficients in JPEG algorithm and so in our method. Finally, in Section 3.3, we present the originality of our method, the watermarking by induction.

3.1 Color conversion and redundancy

Color conversion is a part of the redundancy removal process in JPEG algorithm. JPEG handles colors as separate components. Therefore, it can be used to compress data from different color spaces such RGB, YCrCb, and CMYK. However, the best compression results are achieved if the color components are independents (no correlated) such as in YCrCb. In this color space, most of information is concentrated in the luminance (Y) and less in the chrominance (Cr and Cb). RGB color components can be converted via a linear transformation into YCrCb components as the equations below show:

$$\begin{cases} \mathbf{Y} &= (0.2989 \times \mathbf{R}) + (0.5866 \times \mathbf{G}) + (0.1145 \times \mathbf{B}) \\ \mathbf{Cr} &= (0.7132 \times \mathbf{R}) - (0.7132 \times \mathbf{Y}) + 128 \\ \mathbf{Cb} &= (0.5647 \times \mathbf{B}) - (0.5647 \times \mathbf{Y}) + 128. \end{cases} \tag{8}$$

Another advantage of using the YCrCb color space comes from reducing the spatial resolution of the Cb and Cr chrominance components. Because chrominance does not need to be specified as frequently as luminance: every other Cr and Cb elements can be discarded. As a consequence, a data reduction of 3:2 is obtained by transforming RGB (4:4:4) format into YCrCb (4:2:2) format. The conversion in color space is a first step toward image compressing.

To resist against this first compression process, our method uses color conversion in order to increase the redundancy of messages. The messages, specific to each ROI, are embedded three times, one time by component. All the bits of the message are embedded on each component according to the synchronization described Section 2.3. Consequently, three watermarked channels Y', Cr' and Cb' are obtained. Then, an inverse transformation of color space is done. Three new watermarked components are obtained R', G' and B'. Finally a color watermarked image is built with these three watermarked channels.

3.2 The function of DCT coefficients

In JPEG compression algorithm, after the stage of color conversion, each channel is transformed from the spatial domain into the frequency domain. This process consists of dividing the luminance and chrominance information into square (typically 8×8) blocks and applying a two-dimensional Discrete Cosine Transform (DCT) to each block. Concerning our method, the used block are obtained after the PCA of the ROIs, presented Section 2.2. The DCT converts each block of spatial information into an efficient frequency-space representation that is better suited for compression. Specifically, the transformation produces an array of coefficients for real-valued basis functions that represent each block of data in frequency space. For each block k of the image, we obtain the follow DCT continuous component:

$$F_k(0,0) = \frac{1}{n}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p_k(i,j) = \frac{1}{n}\sum_{x=0}^{N-1} p_k(x), \tag{9}$$

where $p_k(x)$ is the intensity of pixel x in the block k, n the block side and N the pixels number of the block.

In the third stage of JPEG algorithm, each block of DCT coefficients is quantized. Each $F_k(0,0)$ coefficient is divided by $q(0,0)$ which is the first coefficient of quantization table. The result is rounded. We also have:

$$F'_k(0,0) = \left[\frac{F_k(0,0)}{q(0,0)}\right], \tag{10}$$

where $[\ ]$ represents the rounded value, the nearest integer.

The originality of our method is in this stage. We do not round the quantized DC component. We obtain:

$$F'_k(0,0) = \frac{F_k(0,0)}{q(0,0)}, \tag{11}$$

where $F'_k(0,0)$ is a floating value and not either an integer value. Contrary of JPEG algorithm, the floating part is kept then used during our watermarking process.

3.3 Inductive watermarking

Taking into account the constraint of robustness with regard to compression, in particular JPEG, several methods have been developed in the coefficients issued from the DCT transform [9]. Our method follows this JPEG-based approach. But the use of non-rounded DCT coefficients is new and original. The watermarking scheme becomes inductive.

Firstly, we evaluate the nearest even integer which is inferior or equal to $F'_k(0,0)$. Then the difference between $F'_k(0,0)$ and this integer is noted $R_{F'_k(0,0)}$. We obtain:

$$R_{F'_k(0,0)} = F'_k(0,0) - \lfloor \frac{F'_k(0,0)}{2} \rfloor \times 2 \quad \text{and} \quad 0.0 \leq R_{F'_k(0,0)} < 2.0 \ , \tag{12}$$

where $\lfloor \ \rfloor$ represents the nearest inferior integer. $R_{F'_k(0,0)}$ is the value which is modified in order to embed data. Algorithm is normalized to detect the embedded bit b_k as follows:

$$b_k = \begin{cases} \mathbf{0} & \text{if } 0.0 \leq R_{F'_k(0,0)} < 1.0 \ , \\ \mathbf{1} & \text{if } 1.0 \leq R_{F'_k(0,0)} < 2.0 \ . \end{cases} \tag{13}$$

Two values allow a maximum variance without error: the middles of intervals i.e. 0.5 and 1.5. Indeed with these values, b_k is correctly detected even if $R_{F'_k(0,0)}$ varies of 0.5. Consequently the watermarked value of $R_{F'_k(0,0)}$ is equal to:

$$R_{F'_{kw}(0,0)} = b_k + 0.5, \tag{14}$$

where $R_{F'_{kw}(0,0)}$ is the stable value of the remainder $R_{F'_k(0,0)}$ specific to the embedded bit b_k.

The objective of our watermarking method is to modify pixels in order to have $R_{F'_k(0,0)} = R_{F'_{kw}(0,0)}$. To this end, we define d_k as the difference between these values. We also have:

$$\begin{aligned} d_k &= R_{F'_{kw}(0,0)} - R_{F'_k(0,0)} \\ &= b_k + 0.5 - R_{F'_k(0,0)}. \end{aligned} \tag{15}$$

This difference is proportional to the number of modified pixels in the block k. Indeed if the pixel intensities are modified, $F_k(0,0)$ and consequently $F'_k(0,0)$ are equally modified. Then this variation

of $F'_k(0,0)$ modifies $R_{F'_k(0,0)}$. The pixels number to be modified is N_{d_k}:

$$N_{d_k} = \lfloor n \times q(0,0) \times |d_k| \rfloor, \tag{16}$$

where n is the side of the block k and q(0,0) the first coefficient of quantization table.

We underline that only a part of pixels of the block is modified. But if $N_{d_k} > n^2$ then $(N_{d_k} - n^2)$ pixels should be modified twice. The modified pixels are selected in order to reduce the variance in the block. In this way, the pixels modifications are invisible. Their intensities only change one or two grey level. The following equation describes this transformation:

$$p'_k(x) = p_k(x) + sign(d_k), \tag{17}$$

where $p'_k(x)$ is the intensity of modified pixel x.

Finally, we obtain a watermarked block composed of modified and original pixels. To detect the embedded bit in the block k, we calculate the quantized and watermarked DCT Direct component $F'_{kw}(0,0)$. Thus, we have:

$$F'_{kw}(0,0) = \frac{1}{n \times q(0,0)} \left(\sum_{x=0}^{N_{d_k}-1} p'_k(x) + \sum_{x=N_{d_k}}^{N-1} p_k(x) \right). \tag{18}$$

Then we read the LSB of $F'_{kw}(0,0)$. Figure 3 gives the complete scheme of our watermarking method. It shows the PCA, the two symmetric color conversions and the DCT-based watermarking method.

Let us now provide an example of our watermarking method. Take the central area of the first block, which has been built in “Fish” image Figure 5.a. On the Y luminance component, it corresponds

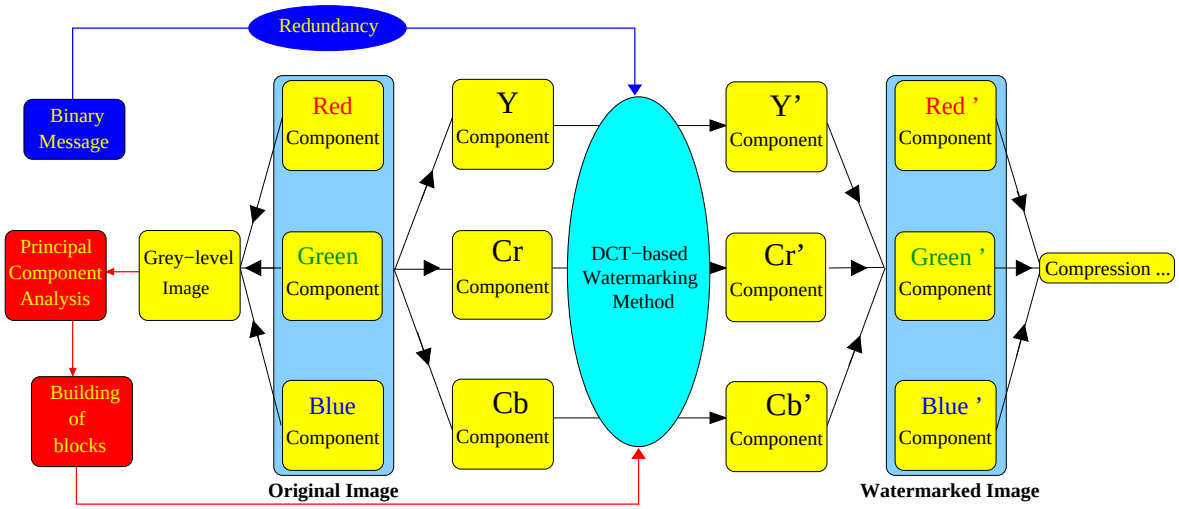


Figure 3: DCT-based watermarking method.

to the following 8×8 matrix:

$$M = \begin{bmatrix} 190 & 152 & 136 & 158 & 176 & 191 & 202 & 223 \\ 225 & 212 & 204 & 212 & 220 & 221 & 227 & 235 \\ 244 & 245 & 245 & 245 & 245 & 244 & 243 & 241 \\ 244 & 242 & 243 & 237 & 237 & 242 & 241 & 238 \\ 240 & 230 & 212 & 188 & 193 & 215 & 236 & 225 \\ 223 & 204 & 172 & 133 & 127 & 157 & 196 & 212 \\ 222 & 195 & 152 & 100 & 90 & 125 & 179 & 208 \\ 230 & 206 & 163 & 114 & 109 & 144 & 194 & 215 \end{bmatrix},$$

where each coefficient gives the grey-level of pixels. We calculate the DCT continuous component with the equation (9) and we obtain $F_1(0,0) = 1608.625$. Then $F_1(0,0)$ is quantized by $q_Y(0,0) = 16$ which is the first coefficient in the luminance quantization table. So we have $F'_1(0,0) = 100.539$. According to the equation (12), the rest $R_{F'_1(0,0)}$ is equal to 0.539. The bit to be embedded can be a **0** or an **1**. Let us carry out the insertion of a bit $b_1 = 0$. In this case $R_{F'_{1w}(0,0)} = 0.5$, then from the equation (15) we have $d_1 = 0.5 - 0.539 = -0.039$. The value of d_1 gives the variation which should be applied to $F'_1(0,0)$ to obtain the desired stable remainder $R_{F'_{1w}(0,0)}$. We note $F'_{1w}(0,0)$ this

corresponding value of DCT continuous coefficient. To obtain $F'_{1w}(0,0)$, from the equation (16) we have to modify $N_{d_1} = 5$ pixels of block. These 5 pixels decrease by one grey-level. They are selected in order to reduce the variance in the block. From the equation (18) we get:

$$\begin{aligned} F'_{1w}(0,0) &= \frac{1}{8\times 16}\left(\sum_{x=0}^{5-1} p'_k(x) + \sum_{x=5}^{8^2-1} p_k(x)\right) \\ &= \frac{1}{8\times 16}\left(\sum_{x=0}^{4}(p_k(x)-1) + \sum_{x=5}^{63} p_k(x)\right) \\ &= \frac{1}{8\times 16}\left(\sum_{x=0}^{63} p_k(x) + \sum_{x=0}^{4}(-1)\right) \\ &= \frac{F_1(0,0)}{16} - \frac{5}{128} = 100.5. \end{aligned} \tag{19}$$

Then we calculate $F_{1W}(0,0)$:

$$\begin{aligned} F_{1w}(0,0) &= q(0,0) \times F'_{1w}(0,0) \\ &= 16 \times 100.5 = 1608.0 \end{aligned} \tag{20}$$

Finally the watermarked matrix M_W is:

$$M_W = \begin{bmatrix} 190 & 152 & 136 & 158 & 176 & 191 & 202 & 223 \\ 225 & 212 & 204 & 212 & 220 & 221 & 227 & 235 \\ 244 & 245 & 245 & 245 & 245 & 244 & 243 & 241 \\ 244 & 242 & 243 & 237 & 237 & 242 & 241 & 238 \\ 240 & 230 & 212 & 188 & 193 & 215 & 236 & 225 \\ 223 & 204 & 172 & 133 & 127 & 157 & 196 & 212 \\ 222 & 195 & 152 & \mathbf{99} & \mathbf{89} & \mathbf{124} & 179 & 208 \\ 230 & 206 & 163 & \mathbf{113} & \mathbf{108} & 144 & 194 & 215 \end{bmatrix},$$

where the values written in **bold** decreased by only one grey-level.

To extract the bit from the value $F_{1w}(0,0)$, we calculate $F'_{1w}(0,0) = 1608/16 = 100.5$. Consequently we obtain $R_{F'_{1w}(0,0)} = 100.5\%2 = 0.5$. The detected bit b_1 is 0. With this example, we

illustrate the inductive aspect of our watermarking method.

4 Results

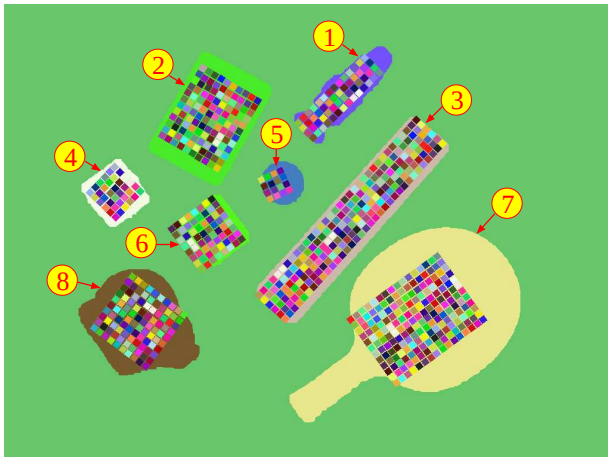
In this section we apply the proposed technique on two images with ROIs, “Fish” (429×347) Figure 5.a and “Objects” (1013×760) Figure 4.a. Then several processing attacks are tested on these two images. In Section 4.1, we describe the detection scheme with the image “Objects”. The detected data are detailed to be used as references in the check of robustness. Then our algorithm has been confronted with three treatments. Section 4.2 shows some results obtained after an image cropping. In Section 4.3, a rotation is applied on the watermarked image. After these geometrical tests, we evaluate in Section 4.4 the robustness of our method against JPEG color compression.

4.1 Detection of data in a standard image

ROI	Message length (bits)	Number of embedded bits by component	block size (pixels)	ROI size (pixels)	Embedding rate
1	28	56	113	10581	59,8%
2	60	120	117	26441	53,1%
3	102	204	113	35261	65,4%
4	18	36	113	7833	51,2%
5	8	16	106	4138	40,9%
6	32	64	113	8620	83,9%
7	96	192	113	68474	31,7%
8	50	100	113	24940	45,3%

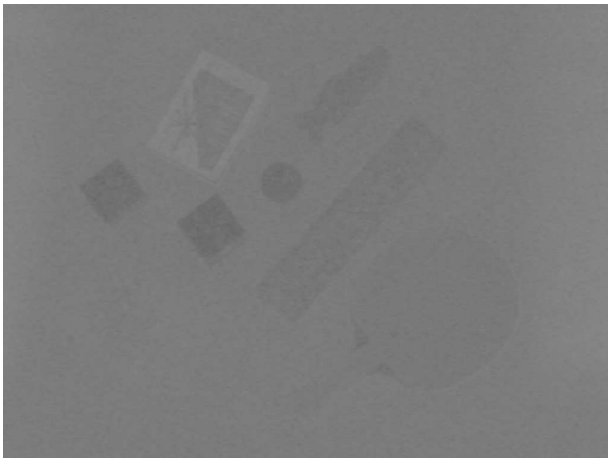
Table 1: Embedded messages and embedding rate for each ROI.

(a)



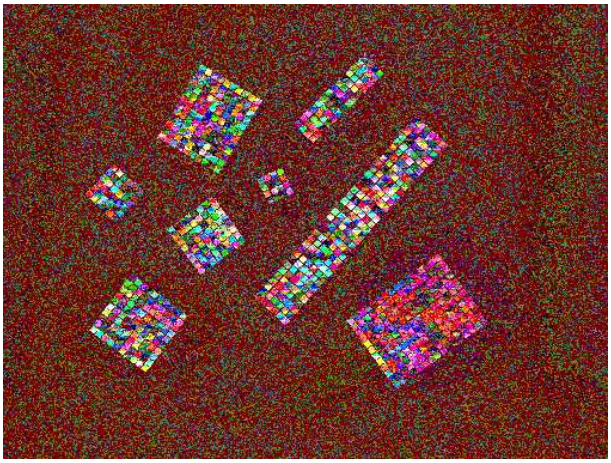
(b)

(c)



(d)

(e)



(f)

Figure 4: a) Original image, b) Detection path and watermarked blocks, c) Y watermarked component, d) Cr watermarked component, e) Color watermarked image, f) Difference between original and watermarked Image.

In this part, we work on the original image “Objects”, which is presented in Figure 4.a. There are 8 objects on a clear background. After segmentation, each object becomes a ROI where a message can be embedded. We regard all regions under 3000 pixels as being too small. Then the picture with the ROI shapes is analyzed to obtain some characteristics such as principal directions. These data are necessary to detect blocks of watermarked pixels. The detection path Dp_{ROI} uses equally these characteristics. The Figure 4.b shows the ROI shapes and the watermarked blocks built according to the Dp_{ROI}. The specific labels associated with each ROI are equally shown.

After this analyze, the watermarking method can start. To increase the robustness, we have chosen to embed many times the messages. The redundancy is double. First the bits of the message are repeated twice in each ROI. Then the color information is used to embed the information three times: one time by color component. The chosen color decomposition space is YCrCb because it is used in color JPEG algorithm. The Figures 4.c and 4.d present respectively the Y and the Cr watermarked components. Finally the watermarked color image is built with the three watermarked channels. The embedded data are invisible as the Figure 4.e shows. But if we make the difference pixel by pixel between the original image Figure 4.a and the watermarked image Figure 4.e we visualize the embedded blocks in Figure 4.f. The PSNR between these two images is equal to 50.52 dB.

The Table 1 gives some information about the embedded data in each ROI: the message length, the number of embedded bits by component, the size of detected block, the ROI size and the embedding rate. A first analyze shows that the message length depends on the ROI size. Indeed if the ROI size increases, the number of pixel blocks increases equally. Consequently, more bits can be embedded. For example, in the first ROI, 56 bits are detected and in the third ROI, 3 times larger, the number of detected bits is multiplied by 4. Then we observe that the number of detected bits is twice larger than the message length. It corresponds to the first degree of redundancy. For example, in the fifth ROI, the binary message is 00110101, and the detected bits are $\mathbf{00110101}00110101$. Each bit is repeated. It is the result of the embedding, which is made in two different places in the ROI. Therefore the detection

results are improved. In other hand, the column 4 of Table 1 presents the block sizes used in each ROI. We observe that the size varies slightly, around the 6%. As our method needs square blocks, the input data is the size of the block edge if this one is aligned with the edges of the image. Then this size is modified according to the ROI orientation and the desired size of block. Finally each region obtains this specific definition of block. By using the block size and the ROI size, we evaluate the embedding rate of ROIs:

$$E_r(ROI) \quad = \quad \frac{ROI \ block \ size \ \times \ Number \ of \ embedded \ bits \ in \ the \ ROI}{ROI \ size}. \qquad (21)$$

This rate varies from $31,7\%$ to $83,9\%$ according to the ROI shape. It shows the disadvantage to use square blocks and the limit of the detection path Dp_{ROI}. Indeed there are still many pixels to watermark information.

To validate more precisely our method, another image is used to embed information. The Figure 5.d shows the color watermarked image. The length of watermarked information is 112 bits and the number of embedded bits is 224. To visualize the embedding data, the difference pixel by pixel between the original image, Figure 5.a, and the watermarked image is shown Figure 5.e.

4.2 Robustness against image cropping

This subsection shows how our method resists to a particular geometrical deformation: the image cropping. The results of detection are presented then analyzed. We evaluate thus the robustness of our method.

The Figure 6.a shows the results of cropping on the watermarked image"Objects". Only 25 % of image could be recovered. In this case, the number of ROIs decreases and only 4 ROIs are detected. The PCA is made and the detection paths are calculated. We obtain the detected blocks illustrated in Figure 6.b. We can observe that only 3 ROIs appear on the picture. In fact, when a ROI touches the picture boundaries, we supposed that it is truncated and the detected message will be naturally wrong.

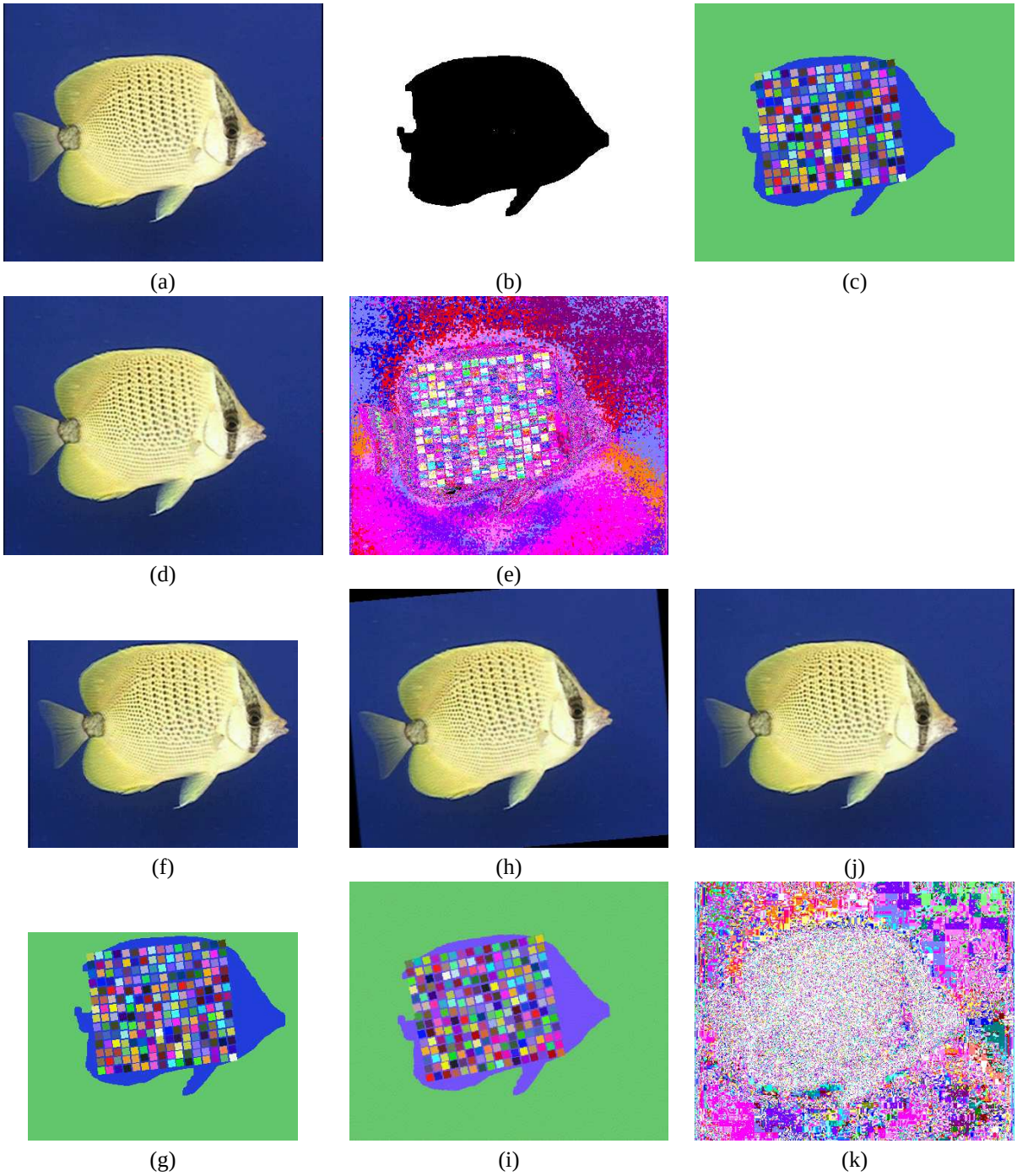
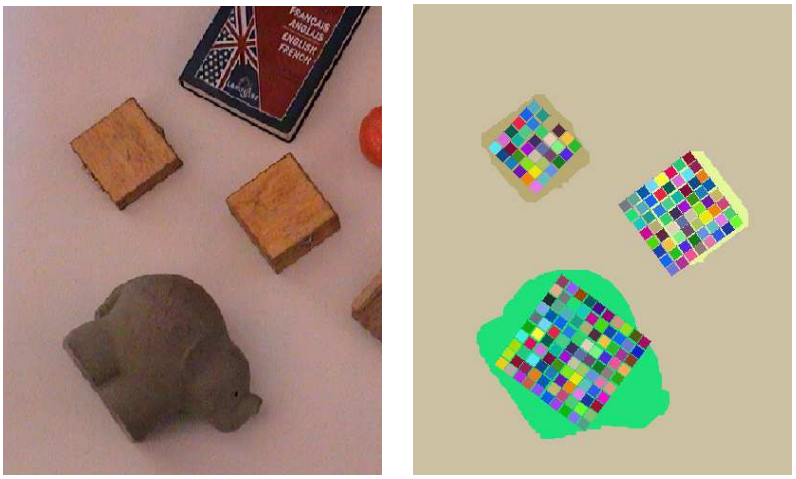


Figure 5: a) The original image “Fish”, b) The associated binary mask, c) “Fish” color label image with blocks, d) Color watermarked image, e) Difference between original color image and color watermarked image, f) Crop of watermarked image, g) Detection and visualization of watermarked blocks, h) 5-degree of rotation on color watermarked image “Fish”, i) “Fish” color label image with watermarked blocks obtained after rotation, j) Compressed color watermarked image with QF = 80%, k) Difference between original and compressed color watermarked image with QF = 80%.



(a) (b)

Figure 6: a) Crop of watermarked image, b) Detection and visualization of watermarked blocks.

ROI	% of right bits (right bits/embedded bits)			
	on Y component	on Cr component	On Cb component	after voting
1	——	——	——	——
2	——	——	——	——
3	——	——	——	——
4	88,9% (32\36)	94,4% (34\36)	91,7% (33\36)	100% (18\18)
5	——	——	——	——
6	75,0% (48\64)	93,7% (60\64)	93,7% (60\64)	100% (32\32)
7	——	——	——	——
8	76,0% (76\100)	90,0% (90\100)	94,0% (94\100)	100% (50\50)

Table 2: Results of bits detection after image cropping on Y, Cr and Cb watermarked components and results after voting.

So the detection does not start in this ROI.

The results of detection after cropping image are given in Table 2. The ROIs, which are disappeared, are of course not detected and their messages are lost. In the recovered ROIs some errors are detected. they are explained by the color space transformation. But after the vote, all bits are rightly detected.

The Figure 5.f shows the results of cropping on the watermarked image"Fish". We obtain the detected blocks illustrated in Figure 5.g. After the vote all bits are also rightly detected. We can conclude that the synchronization resists to the cropping image. Our method is robust against this kind of geometrical modification.

4.3 Robustness against Rotations

(a) (b)

Figure 7: a) 5-degree of rotation on color watermarked image "objects", b) "Objects" color label image with watermarked blocks obtained after rotation,

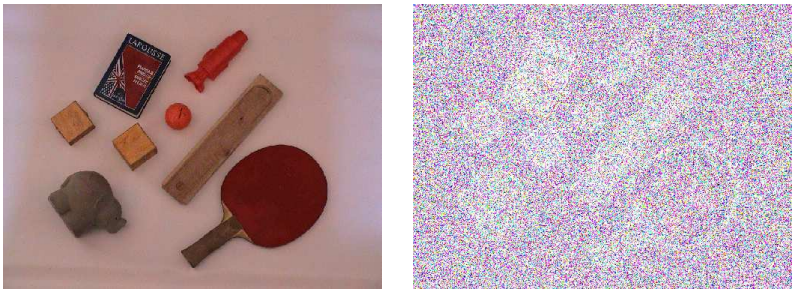
In this subsection, another geometrical modification is used to evaluate the robustness of our method. A 5-degree rotation is applied on two watermarked image Figures 4.e and 5.d. Two rotated images are obtained Figures 7.a and 5.h. With this modification, we check the robustness of synchronization between image and hidden data. After segmentation, we obtain the rotated ROIs and we start the PCA. Then the detection paths and the watermarked blocks are detected, illustrated Figures 7.b and 5.i. We can observe that the ROI principal axes are just rotated of 5 degrees. So the detection path

ROI	% of right bits (right bits/embedded bits)			
	on Y component	on Cr component	on Cb component	after voting
1	92,9% (52\56)	94,6% (53\56)	89,3% (50\56)	100% (28\28)
2	87,5% (105\120)	84,2% (101\120)	94,2% (113\120)	100% (60\60)
3	75,0% (153\204)	86,7% (177\204)	93,6% (191\204)	100% (102\102)
4	87,5% (14\16)	93,7% (15\16)	93,7% (15\16)	100% (8\8)
5	80,6% (29\36)	86,1% (31\36)	88,9% (32\36)	100% (18\18)
6	56,2% (36\64)	93,7% (60\64)	92,2% (59\64)	100% (32\32)
7	81,2% (156\192)	94,3% (181\192)	87,5% (168\192)	100% (96\96)
8	57,0% (57\100)	87,0% (87\100)	94,0% (94\100)	100% (50\50)

Table 3: Results of bits detection after 5-degree of rotation on each color component Y, Cr and Cb.

stays usable. In other hand, if the principal axes change, the shapes of watermarked blocks change equally. It is explained by the discrete structure of images. Moreover the discretization makes another problem: if the image is turned, several pixels that compose the boundaries of blocks can change and the corresponding direct DCT components can be consequently modified.

It is one of reasons of the detected errors, which are illustrated in Table 3. This Table gives the percentage of right bits in each ROI, on Y, Cr and Cb components. With the double redundancy, the bits are watermarked 6 times that is 2 times by channels. So a process of vote is started and the results are presented in the last column of Table 3. In each ROI, the message is correctly detected. Therefore, our synchronization resists to rotations.



(a) (b)

Figure 8: a) Compressed color watermarked image with QF = 80%, b) Difference between original and compressed color watermarked image with QF = 80%.

4.4 Detection results after Compression

In this subsection we applied the color JPEG algorithm on the watermarked image to test the robustness against compression. The Figure 8.a shows the compressed watermarked image with a quality factor equal to 80 %. To decrease the image size, the JPEG compression algorithm modifies the color pixels. And if the quality factor is small, the modifications are important and the modified pixels are numerous. With our method, the detection is correct if the quality factor is superior to 75 %. Below this value, the noise is too significant. The Table 4 gives the percentage of right detected bits in each ROI on three components. We observe that the messages are correctly detected, thanks to the voting use.

The Figure 8.b shows the difference between the compressed and watermarked image Figure 8.a and the original image Figure 4.a. With this difference image, the invisibility of watermarking is illustrated, the PSNR is equal to $43,19$ dB.

The Figure 5.j shows the compressed watermarked image “Fish” with a quality factor equal to 80 %. The Figure 5.k shows the difference between the compressed and watermarked image Figure 5.j and the original image Figure 5.a. The message is correctly detected thanks to the voting use. The modifications made by our watermarking method are weak against compression modifications.

ROI	% of right bits (right bits/embedded bits)			
	on Y component	on Cr component	on Cb component	after voting
1	69,6% (39\56)	87,5% (49\56)	75,0% (42\56)	100% (28\28)
2	75,8% (91\120)	87,5% (105\120)	81,7% (98\120)	100% (60\60)
3	75,5% (154\204)	88,2% (180\204)	88,7% (181\204)	100% (102\102)
4	72,2% (26\36)	94,4% (34\36)	83,3% (30\36)	100% (18\18)
5	68,8% (11\16)	75,0% (12\16)	87,5% (14\16)	100% (8\8)
6	50,0% (32\64)	81,3% (52\64)	82,8% (53\64)	100% (32\32)
7	75,5% (145\192)	76,0% (146\192)	81,25% (156\192)	100% (96\96)
8	50,0% (50\100)	82,0% (82\100)	88,0% (88\100)	100% (50\50)

Table 4: Results of bits detection after compression on each color component Y, Cr and Cb.

5 Conclusion and perspectives

In this paper, we have presented a color DCT-based watermarking method, which uses the content of images. To obtain the synchronization between the message and the image, an analysis is made and several ROIs are created. The content of image is used to synchronize message and image. Then the three color components Y, Cr and Cb are used to embed three times the message. It is the first degree of redundancy. What is more, each bit is duplicated and embedded two times. It corresponds to the second degree of redundancy. As a result, the robustness is largely improved. Finally, our watermarking method is inductive because we modify the non-rounded DCT coefficients. The watermarking is made by anticipating the quantization.

The different results illustrate the fact the robustness of our watermarking method depends on the image color segmentation. But the primary objectives were realized. Our method is robust to a

great variety of processing such as rotations, cropping or color JPEG compression. The embedded information stays invisible. Moreover the watermarking impact on the image is weak compared with others modifications like compression.

All the results have been obtained with a watermarked block size around to $\mathbf{11} \times \mathbf{11}$. If the size block is smaller, the number of embedded information increases but the robustness decreases. On contrary if the size block increases, the robustness is improved but the hidden data is very small. To improve the quantity of embedded data, we intend to adapt the watermarked block shape to the ROI shape. All the block would not square and the embedding rate increases. As new research orientation, we would like to change the size of block according to the ROI size in order to be robust against zoom.

References

[1] P. Bas, J.M Chassery, and B.Macq. Image watermarking: an evolution to content based approaches. *Pattern Recognition*, (35):545–561, 2002.

[2] P. Bas and B. Macq. A new video-object watermarking scheme robust to object manipulation. In *Proc. of ICIP'01, Tessaloniki, Greece*, pages "526–529", 2001.

[3] A.G. Bors and I. Pitas. Image watermarking using block site selection and DCT domain constraints. *Optics Express*, 3(12):512–522, 1997.

[4] G. Chareyron and A. Tremeau. Watermarking of color images based on a multi-layer process. In *CGIV'02, Poitiers, France*, pages 77–80, 2002.

[5] I.J. Cox, J. Killian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Trans. on Image Processing*, 6(12):1673–1687, 1997.

[6] F. Deguillaume, S. Voloshynovskiy, and T. Pun. Hybrid robust watermarking resistant against copy attack. In *EUSIPCO'02, Toulouse, France*, 2002.

[7] F. Hartung and M. Kutter. Multimedia watermarking techniques. In *IEEE Proceeding 87*, pages 1079–1107, 1999.

[8] G. Healey. Using Color for Geometry-Insensitive Segmentation. *Journal of the Optical Society of America-A*, 6(6):920–937, June 1989.

[9] A. Bors I. and Pitas. Image watermarking using dct domain constraints. In *Proceedings ICIP, vol. 3, Lausanne, Switzerland*, pages 231–234, 1999.

[10] E. Koch and J. Zhao. Towards robust and hidden image copyright labeling. In *IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Greece*, pages 452–455, 1995.

[11] M. Kutter. Watermarking resisting to translation, rotation and scaling. In *Proceeding of SPIE Multimedia Systems and Applications,vol 3528, Boston, USA*, pages 423–431, Nov 1998.

[12] M.P. Queluz. Content-based integrity protection of digital images. In *SPIE Proceeding 3657*, pages 85–93, 1999.

[13] A. Rosenfeld and J. Pfaltz. Distance Functions in Digital Pictures. *Pattern Recognition*, 1:33–61, June. 1968.

[14] J.J.K. O Ruanaidh and T. Pun. Rotation, scale and translation invariant spread spectrum digital image watermarking. In *Signal Processing 66*, pages 303–317, 1998.

[15] J. Serra. *Image Analysis and Mathematical Morphology, vol. 2*. London: Academic Press, 1988.

[16] G. Voyatzis and I. Pitas. The use of watermarks in the protection of digital multimedia products. In *IEEE Proceeding 87*, pages 1197–1207, 1999.

[17] G. Wallace. The JPEG still picture compression standard. *Communication of the ACM*, 34(4):31–44, Apr. 1991.