Watermark Resynchronization based on Elastic Graph Matching

Gwenaël Doërr, Christian Rey and Jean-Luc Dugelay

Eurécom Institute
Multimedia Communications Department
2229 route des Crêtes – B.P. 193 – 06904 Sophia-Antipolis - France
{**doerr, rey, dugelay**}**@eurecom.fr**

Abstract: Most of the watermarking algorithms can still be defeated by geometric distortions today. If the weakness against global distortions can almost be considered as solved, local geometric distortions such as the ones introduced by StirMark remain a major issue. In this paper, an original resynchronization method is presented as a potential countermeasure against such attacks. The basic idea consists in interlacing resynchronization bits with the bits carrying the payload during the watermark embedding. During the extraction, those bits are used as anchor points to estimate and compensate for both small local and global geometric distortions. This registration procedure is performed using an Elastic Graph Matching (EGM) approach.

Keywords: Digital watermarking, robustness, resynchronization

1 Introduction

Digital watermarking was introduced in the early 90's as a complementary protection technology. Encryption alone is indeed not enough to protect multimedia data: sooner or later, encrypted data is decrypted to be viewed/listened by human beings and can be perfectly duplicated and redistributed at a large scale. Inserting imperceptible watermarks surviving to several signal processing primitives has consequently received an increasing interest. There exists a trade-off between several conflicting parameters and most research initiatives have been dedicated to better understand it: perceptual models have been exploited to make embedded watermarks less perceptible, benchmarks have been released to evaluate robustness, channel models have been considered to obtain a theoretical bound for the embedding capacity. Nevertheless, progress in security fields is usually an iterative process. Hackers create new attacks to beat down security systems and system designers introduce new countermeasures to survive to new attacks. Robustness has been thus considered for a long time as a key parameter in digital watermarking. However, if most of the watermarking algorithms are robust against usual image processing primitives such as filtering or lossy compression, they are still weak against geometric distortions.

An overview of spatial geometric distortions relevant in image watermarking is given in Section 2. Additionally, alternative countermeasures to compensate for geometric distortions are briefly reviewed. This article focuses then on the robustness against local geometric distortions. In Section 3, a baseline watermarking scheme developed by the Eurécom Institute and based on fractal image coding is presented. Next, a resynchronization framework is introduced in Section 4. It relies on the insertion of control bits during the embedding step so that they can be used as anchor points to compensate for geometric distortions during the detection procedure. Two alternative resynchronization processes are investigated: Block Matching (BM) on one side and Elastic Graph Matching (EGM) on the other side. Both systems are finally compared in Section 5 in terms of robustness against the StirMark attack.

2 Background

A distinction is usually made between two brands of attacks when watermarking systems are benchmarked. On one side, synchronous attacks simply modify the *sample values*. Typical examples include noise addition, filtering, quantization, lossy compression. On the other side, asynchronous (or geometric) attacks modify the *sample positions*. In this case, the embedded watermark is not removed but the detector is unable to retrieve it since the synchronization convention shared by both the embedder and the detector is no longer valid. A brief overview of geometric distortions is given in Subsection 2.1 since robustness to such desynchronization attacks will be the main focus of this article. Next, several alternative countermeasures are presented in Subsection 2.2.

2.1 Geometric Distortions

In real life, geometric distortions usually result either from a physical manipulation, e.g. the print and scan attack, or from a digital manipulation. Geometric transformations basically map each image pixel $\mathbf{p} = (x, y)$ to a new location $\mathbf{p}' = (x', y')$. With this definition in mind, people usually differentiate global and local distortions. Global transformations can be described using a model with a reasonable number of fixed parameters. On the other hand, it is not possible to model local distortions with a unique model and a fixed set of parameters.

2.1.1 Global geometric distortions

A geometric transformation is said to be global if the field of pixel displacements is simple enough so that it can be described using a unique model with a limited number of parameters, or degrees of freedom, having fixed values. The more parameters in the model, the more complex can be the distortions that it describes. A few examples of such models are presented below and the associated displacements are illustrated in Figure 1. Of course, one can also combine those different models to obtain even more sophisticated geometric transforms.

Affine transform. The coordinates mapping can be described with the following equation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} \quad (1)$$

where the six degrees of freedom a, b, c, d, e and f control the zoom, translation, rotation and shearing. This distortion preserves the relative distance between points and parallelism.

Bilinear transform. This transform is slightly more generic and is used to model the distortions due to a misalignment between the display and capture device e.g. the handy cam attack during movie projection in theater [11]. It has eight degrees of freedom and can be expressed as follows:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} xy + \begin{pmatrix} g \\ h \end{pmatrix} \quad (2)$$

This transformation can be seen as moving the corners of the image and mapping the other points so that their relative positions remain the same.

Curved transform (or bending). This model is used to approximate the optical transformations due to the lens when deformation amplitudes are small. The transform between the old and new coordinates is given by the following expression:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} [(1-\alpha)a + \alpha b]\sin(\beta\pi) \\ [(1-\beta)c + \beta d]\sin(\alpha\pi) \end{pmatrix} \quad (3)$$

where a, b, c, d are the focal parameters and $0 \leq \alpha, \beta \leq 1$ are the normalized coordinates in the image.

High-frequency sinusoidal transforms. Those transformations are similar to the curved transform except that higher frequencies ($\omega_x, \omega_y > \pi$) are assigned to the sinusoidal function [40]. This results in two different types of distortions. The sinusoidal stretch and shrink, defined as follows:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a\sin(\omega_x\alpha) \\ b\sin(\omega_y\beta) \end{pmatrix}, \quad (4)$$

distorts the image by locally stretching and shrinking the image. Such distortions may not be perceptually disturbing depending on the image content. Alternatively, pixels can be locally shifted to the left/right are upwards/downwards:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a\sin(\omega_x\beta) \\ b\sin(\omega_y\alpha) \end{pmatrix} \quad (5)$$

This kind of distortions can be regarded as some sinusoidal jitter and rapidly becomes visible when the parameters a and b grow.

2.1.2 Local geometric distortions

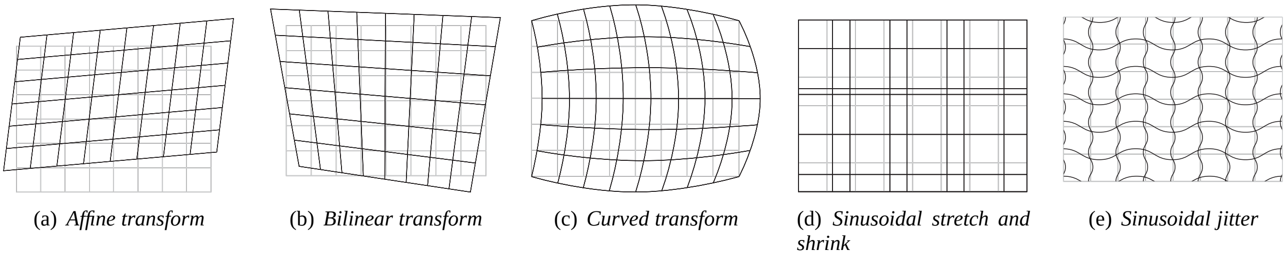
As an alternative to global geometric distortions, one can divide the image in many subregions and consider a transformation with specific parameters for each subregion. With such a local approach, a very wide class of transformations can be modeled. In fact, the number of degrees of freedom is now proportional to the number of subregions. A continuity constraint could be imposed for adjacent regions. However, purely uncorrelated local geometric distortions can also be considered. For instance, the random jitter attack basically consists in changing the pixel locations by a small random amount [31]. To date, StirMark or the Random Bending Attack (RBA) is the reference attack when local geometric distortions are considered [37, 28]. It can be seen as a complex global transformation involving many local transformations. This attack is described in further details in Subsection 5.1.

2.2 Countermeasures

Robustness to geometric distortions is a great challenge in digital watermarking since the watermark detector usually assumes implicitly a perfect synchronization with the embedded watermark. A small misalignment can result in a drastic loss in performances. A significant part of the research effort has consequently been devoted to design efficient countermeasures which enable the watermark detector to be immune against geometric distortions. A brief overview of those methods is given below.

2.2.1 Non-blind detectors

When the original non-watermarked image is available on the detector side, the undergone transformation can be estimated and inverted prior to watermark detection. The procedure usually consists in estimating the displacement in some points, e.g. using block-matching, and then in computing the parameters for a given model which best describe the estimated displacement field [44, 25, 7, 35, 11, 33]. Other techniques compute some geometric characteristics in both images



(a) *Affine transform* (b) *Bilinear transform* (c) *Curved transform* (d) *Sinusoidal stretch and shrink* (e) *Sinusoidal jitter*

Figure 1. *Impact of alternative geometric transforms on a regular grid.*

to be able to estimate the parameters of the transformation [3]. As an alternative approach, a regular tessellation can be applied on both images and the goal is then to find some slight shifts for the vertices of the attacked image so that the quadratic error between corresponding triangles is minimized [9, 14]. This latter approach enables to cope with local geometric transforms. However, in order to avoid storing all the original documents, blind watermarking detectors are prefered.

2.2.2 Exhaustive search

In this brute force perspective, each potential geometric transformation that might have been applied to the watermarked image is inverted and the watermark detector checks whether it can find any underlying watermark [27, 23, 1]. Obviously, such an approach is only feasible for a restricted subset of geometric transformations. As the set of hypothetical geometric transformation is enlarged, the method rapidly becomes computationally too expensive. Furthermore, performing successive detections can significantly increase the false positive probability [30].

2.2.3 Geometric transformation inversion

Currently, a common resynchronization technique consists in inserting an additional watermark which is often referred to as template, registration pattern or pilot watermark. This template is then basically used as a reference to detect and compensate for geometric distortions such as affine transforms [20, 46]. To do so, one can embed a small watermark patch several times in the spatial domain according to a predefined pattern e.g. a grid [26, 24, 45, 43, 10]. It results in local peaks in the autocorrelation or in the Fourier transform of the image which can be exploited to identify and invert the undergone geometric transformation. Alternatively, one can also create local peaks directly in the frequency domain [36, 4]. Anyway the main drawback of those techniques is that they rely on the presence of local peaks which can be easily detected. Thus, a malicious party can remove those peaks e.g. in the frequency domain and deprive the detector of any means of registration [47].

2.2.4 Immune embedding space

Another solution consists in embedding the watermark in a subspace which is immune to geometric distortions. In other terms, if the watermarked image is submitted to a geometric transformation in the spatial domain, it has no impact in the invariant subspace i.e. the watermark is still synchronized. Alternative ways of building such invariant subspaces have been proposed in the literature.

Image moments. Geometric image moments can be considered to normalize an image. For instance, the resulting image can be made invariant to rotation, scaling and flipping [2, 15]. As a result, if the watermark is embedded in this normalized space, it is robust to any combination of the above mentioned attacks. Moments can also be used to normalize video objects before watermark embedding [6].

Properties of the transform domain. The distribution of pixel values usually remains quite stable against geometric distortion. A robust watermark can thus be embedded by specifying the shape of the image histogram [8]. Similarly, the average grey level of an image is not modified by geometric transformations and can be used to convey information [21]. In another fashion, the Fourier transform and in particular its magnitude has many properties which can be exploited to design a robust watermark. For instance, a watermark can be embedded in a ring covering middle frequencies. This ring is then separated in different sectors and the same watermark is embedded in each sector. The resulting watermark can then be proven to be robust against translation, cropping, scaling and some rotations [41]. Alternatively, the log-polar mapping of the magnitude of the Fourier transform can be averaged along the log-radius axis to obtain a signal which is invariant to translation and scaling. A rotation results then in a cyclical shift of the signal which can be easily compensated with a simple search [32]. The properties of the Fourier transform can even be further exploited by considering the Fourier-Mellin transform which maps scalings and rotations to simple translations. Embedding a watermark in this specific domain consequently enables to survives those transformations [39]. However, implementation difficulties due to interpolation seem to have hampered further work in this direction.

Image features. Another way to obtain an invariant embedding space is to consider the intrinsic features of the image. For instance, corners are likely to remain corners even after a geometric transform. Identifying such feature points enables to design highly robust schemes. One can embed small watermark patches at those specific locations [42]. Alternatively one can also use those feature points to define a partition of the image e.g. a Delaunay tessellation and then watermark each element of this partition in a normalized space [5, 13]. The nature itself of the document to be protected can be considered. For example, with face images, the position of the eyes, the nose and the mouth can be used for normalization [34]. It should be noted that the main concern of such methods is usually the stability of the feature extractor with respect to the possible distortions. Furthermore, the extracted features should be chosen in a pseudo-random fashion.

Finally it should be noted that this immunity against geometric distortions usually comes with a reduction of the capacity i.e. the number of bits that can be embedded. The broader is the range of geometric distortions that the embedding space is invariant to, the fewer bits can be hidden.

3 Eurémark

Eurécom watermarking algorithm [16, 18] exploits fractal image coding theory [19] and in particular the notion of self-similarities. The image is considered as a collection of local similarities modulo an affine photometric compensation and a pool of geometric transformations. The underlying idea is then to use invariance properties of fractal coding such as invariance to affine photometric transformations to ensure watermark robustness. Furthermore, the extraction process is performed in a blind fashion i.e. the original image is not required.

3.1 Watermark embedding

The embedding process can be divided in three different steps. First, a *fractal approximation* $\mathbf{I}_\mathrm{o}^\mathrm{IFS}$ of the original image $\mathbf{I}_\mathrm{o}$ is computed. Second, the payload is properly formatted and encrypted to obtain the watermark $\mathbf{W}$ to be embedded. Finally, the watermark is merged with the cover $\mathbf{I}_\mathrm{o}^\mathrm{c} = \mathbf{I}_\mathrm{o} - \mathbf{I}_\mathrm{o}^\mathrm{IFS}$ according to a sign rule.

3.1.1 Cover generation

The input image is scanned block by block. Those blocks $\mathbf{R}_i$ are labeled as *range blocks* and have a dimension $r \times r$ e.g. 8×8 pixels. The goal is then to find for each block a *domain block* $\mathbf{D}_i$ taken from a pool of blocks which is similar according to the Mean Square Error (MSE) criterion defined below:

$$\mathrm{MSE}(\mathbf{R}, \mathbf{D}) = \frac{1}{r^2} \sum_{\mathbf{p} \in [1,r] \times [1,r]} \big(\mathbf{R}(\mathbf{p}) - \mathbf{D}(\mathbf{p})\big)^2 \quad (6)$$

where $\mathbf{p}$ is a bi-dimensional spatial index used to address the pixels of the blocks $\mathbf{R}$ and $\mathbf{D}$. By analogy with fractal image coding theory, for each range block, a search window is defined and the blocks $\mathbf{Q}_j$ lying in it are collected to initialize a codebook. Each block is then scaled to match the dimensions $r \times r$ of the range blocks. Next, the codebook is enlarged by building k geometrically transformed blocks $\mathrm{T}_k(\mathbf{Q}_j)$ e.g. identity, 4 flips and 3 rotations. An affine photometric compensation is then performed for each transformed block to minimize the Mean Square Error with the range block $\mathbf{R}_i$ i.e a photometric scaling s and an offset o are computed to minimize $\mathrm{MSE}(s.\mathrm{T}_k(\mathbf{Q}_j) + o, \mathbf{R}_i)$. Finally, the range block $\mathbf{R}_i$ is substituted by the transformed block $s.\mathrm{T}_k(\mathbf{Q}_j) + o$ which has the lowest MSE. The whole matching process is depicted in Figure 2. The cover $\mathbf{I}_\mathrm{o}^\mathrm{c}$ is simply obtained by computing the signed difference between the original image and its fractal approximation:

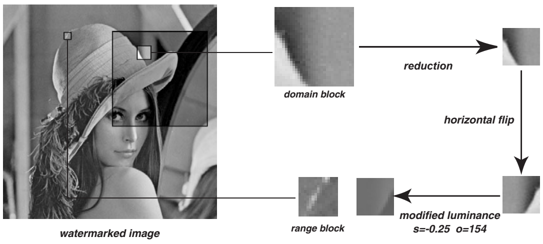


Figure 2. *Self-similarities: an example of association between range and domain blocks modulo an affine photometric compensation and a pool of geometric transformations.*

$$\mathbf{I}_\mathrm{o}^\mathrm{c} = \mathbf{I}_\mathrm{o} - \mathbf{I}_\mathrm{o}^\mathrm{IFS} \quad (7)$$

3.1.2 Watermark formatting

The payload to be hidden (a string or a logo) is first converted into a binary mark[1]. Then it is duplicated to ensure robustness against small modifications of the cover. On one hand, the binary mark is over-sampled by a scaling factor to produce a low-frequency watermark more resilient to low-pass filtering and lossy compression. On the other hand, this over-sampled mark is tiled horizontally and vertically up to the size of the image. This spatial repetition enables to compensate loss of information due to local image manipulations. At this point, the final binary watermark $\mathbf{W}$ is obtained by encrypting the over-sampled tiled binary mark with a binary over-sampled pseudo-random sequence using a XOR operator. The XOR operation removes repetitive patterns and thus reduces the psycho-visual impact of the watermark. Nevertheless, using an over-sampled sequence permits to retains the low-frequency nature of the encrypted binary mark. Additionally, the XOR operation secures the hidden payload, typically against collusion attacks.

3.1.3 Modulation

Modulating the watermark $\mathbf{W}$ with the cover $\mathbf{I}_\mathrm{o}^\mathrm{c}$ basically consists in zeroing some cover samples depending on their sign and the corresponding watermark bit

[1] An Error Correcting Code (ECC), typically a block turbo code [38], can be inserted before any other formatting to further improve robustness against photometric attacks.

to hide. More formally the following rules are applied:

$$\mathbf{I}_{\mathrm{o}}^{\mathrm{w}}(\mathbf{p}) = \begin{cases} \mathbf{I}_{\mathrm{o}}^{\mathrm{c}}(\mathbf{p}), & \text{if } \mathbf{W}(\mathbf{p}) = 1 \text{ and } \mathbf{I}_{\mathrm{o}}^{\mathrm{c}}(\mathbf{p}) > 0 \\ & \text{or } \mathbf{W}(\mathbf{p}) = 0 \text{ and } \mathbf{I}_{\mathrm{o}}^{\mathrm{c}}(\mathbf{p}) < 0 \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $\mathbf{I}_{\mathrm{o}}^{\mathrm{w}}$ is the watermarked cover. It should be noted that, in average, only one pixel out of two is modified. Furthermore, for visibility reasons, high valued samples should not be zeroed. A threshold τ_{high} is consequently introduced to discard systematically high valued samples as follows:

$$\mathbf{I}_{\mathrm{o}}^{\mathrm{w}}(\mathbf{p}) = \mathbf{I}_{\mathrm{o}}^{\mathrm{c}}(\mathbf{p}) \text{ if } \left|\mathbf{I}_{\mathrm{o}}^{\mathrm{c}}(\mathbf{p})\right| > \tau_{\mathrm{high}} \tag{9}$$

Finally, the watermarked cover is added to the fractal approximation to produce the watermarked image $\mathbf{I}_{\mathrm{w}} = \mathbf{I}_{\mathrm{o}}^{\mathrm{IFS}} + \mathbf{I}_{\mathrm{o}}^{\mathrm{w}}$. By default, the threshold τ_{high} is chosen so that the embedding process results in a distortion of 38 dB in terms of Peak Signal to Noise Ratio (PSNR).

3.2 Watermark extraction

The extraction process is somewhat dual to the embedding. In a first step, a fractal approximation is computed. Then the embedded payload is retrieved according to some extraction rules and a detection score is computed.

3.2.1 Cover extraction

As during the embedding process, a fractal approximation $\mathbf{I}_{\mathrm{w}}^{\mathrm{IFS}}$ of the watermarked image is computed and the associated cover $\mathbf{I}_{\mathrm{w}}^{\mathrm{c}} = \mathbf{I}_{\mathrm{w}} - \mathbf{I}_{\mathrm{w}}^{\mathrm{IFS}}$ is extracted. A basic assumption is that fractal coding is stable enough so that $\mathbf{I}_{\mathrm{w}}^{\mathrm{IFS}} \approx \mathbf{I}_{\mathrm{o}}^{\mathrm{IFS}}$ and thus $\mathbf{I}_{\mathrm{w}}^{\mathrm{c}} \approx \mathbf{I}_{\mathrm{o}}^{\mathrm{w}}$. This cover is then decoded according to the following rule to obtain a ternary watermark $\tilde{\mathbf{W}}$:

$$\tilde{\mathbf{W}}(\mathbf{p}) = \begin{cases} 1, & \text{if } \quad \tau_{\mathrm{low}} < \mathbf{I}_{\mathrm{w}}^{\mathrm{c}}(\mathbf{p}) < \tau_{\mathrm{high}} \\ -1, & \text{if } -\tau_{\mathrm{high}} < \mathbf{I}_{\mathrm{w}}^{\mathrm{c}}(\mathbf{p}) < -\tau_{\mathrm{low}} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

Only samples whose magnitude is between the thresholds τ_{low} and τ_{high} are considered as carrying information related to the watermark. High valued samples are discarded since they are likely not to have been considered for watermarking during the embedding process. Furthermore, low valued samples are neglected since they might result from the non perfect cover stability ($\mathbf{I}_{\mathrm{w}}^{\mathrm{c}} \neq \mathbf{I}_{\mathrm{o}}^{\mathrm{w}}$).

3.2.2 Payload extraction

The binary pseudo-random sequence used during embedding is regenerated using the shared secret key. Its values are then mapped from {0,1} to {1,-1} and the resulting antipodal binary sequence is multiplied with the ternary watermark $\tilde{\mathbf{W}}$ to invert the XOR operation performed during embedding. Next, the following quantities are computed for each payload bit:

$$d_k = \sum_{\mathbf{p}\in\mathcal{R}_k} \tilde{\mathbf{W}}(\mathbf{p}) \quad \text{and} \quad s_k = \sum_{\mathbf{p}\in\mathcal{R}_k} |\tilde{\mathbf{W}}(\mathbf{p})| \tag{11}$$

where $\mathcal{R}_k$ is the set of positions where the k^{th} bit has been duplicated. The value s_k indicates how many positions have been considered as carrying information related to the watermark and d_k the difference between position voting for 1 and those voting for 0. The final value of the k^{th} payload bit b_k can then be determined with a simple *majority vote* as follows:

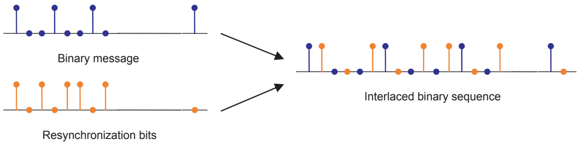


Figure 3. *Interlacing payload and resynchronization bits.*

$$b_k = \begin{cases} 0, & \text{if } d_k < 0 \\ 1, & \text{if } d_k \geq 0 \end{cases} \tag{12}$$

Right now, whatever image is given in input, a sequence of bit is extracted. The following score is consequently computed:

$$\rho = \frac{\sum_{k=1}^{K} |d_k|}{\sum_{k=1}^{K} s_k} \tag{13}$$

where K is the number of payload bits. When all the positions associated with a given bit are voting for the same bit value (watermarked image), $d_k = \pm s_k$ and $\rho = 1$. On the contrary, if the positions vote evenly for 0 and 1 (non watermarked image), then $d_k = 0$ and $\rho = 0$. As a result, the detection score ρ can be compared to a threshold τ_{detect} to assert whether a watermark has been effectively embedded or not.

4 Resynchronization process

The presented watermarking scheme implicitly assumes that the sample positions have not been modified after watermark embedding. In others terms, nothing specific has been done regarding a possible loss of synchronization and this algorithm is thus inherently weak against geometric distortions. To be robust against local geometric distortions, a resynchronization framework has been added. A pilot watermark made of control bits is inserted during embedding. Those bits are then used as anchor points to compensate for geometric distortions during detection using a registration procedure such as block matching (BM) [17] or, even better, Elastic Graph Matching (EGM).

4.1 Pilot watermark

To invert potential geometric distortions, resynchronization bits are interlaced with the payload bits during embedding. On the detector side, those bits are also *known* and can be used as anchor points. This can be regarded as some sort of template which is used to align the detector with the watermark in a blind fashion. However, this template is not a repetitive pattern which can be easily isolated in the frequency domain [47].

4.1.1 Resynchronization bits insertion

A sequence of resynchronization bits is generated pseudo-randomly using the secret key. Those bits are then regularly interlaced with the payload bits as de-

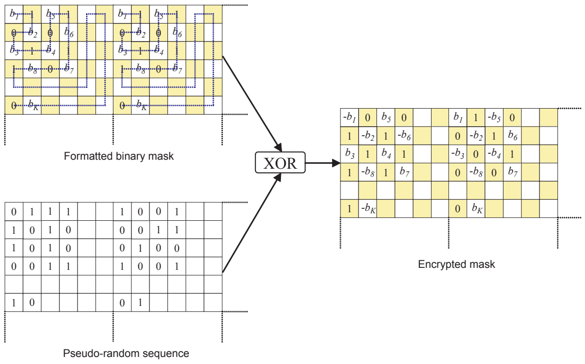


Figure 4. *Example of resynchronization mask generation.*

picted in Figure 3. This mono-dimensional binary signal is then reshaped to produce a bi-dimensional mark which is embedded as described in Subsection 3.1 i.e. over-sampling, tiling, encryption and modulation with the fractal cover. Nevertheless, there exists a trade-off regarding the ratio between the number of payload and resynchronization bits. Indeed, the higher the density of resynchronization bits, the finer are estimated the geometric distortions. However, this is counterbalanced by a loss in robustness due to the fact that each payload bit is repeated fewer times. On the other hand, without enough resynchronization bits, the resynchronization process is likely to fail and the message cannot be extracted. In practice, a relevant balance is to have the same number of resynchronization and payload bits.

4.1.2 Resynchonization mask for extraction

To use resynchronization bits as anchor points for registration, it is necessary to know their values and the way the watermark was formatted during embedding. A resynchronization mask $\mathbf{M}$ is consequently created. It precisely indicates the original layout of payload and resynchronization bits, as well as the impact of the mark encryption as depicted in Figure 4. The generation of this mask is similar to the watermark generation during the embedding process except that payload bits are replaced by their labels $\{b_1, b_2, \cdots, b_K\}$. Furthermore, those labels are signed according to the bits of the pseudo-random binary sequence used for encryption so that the XOR operation can be inverted during the payload extraction. Finally, the resynchronization bits are mapped from {0,1} to {-1,1}.

4.2 Block-matching based resynchronization

Block-matching is applied between the resynchronization mask $\mathbf{M}$ and the extracted ternary watermark $\tilde{\mathbf{W}}$ to compensate for local geometric distortions. A $n \times n$ block of a matrix $\mathbf{A}$ is indexed as follows:

$$\mathbf{A}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}(\mathbf{p}) = \mathbf{A}(n.x_b + \delta^x + x, n.y_b + \delta^y + y) \qquad (14)$$

where $\mathbf{p}_b = (x_b, y_b)$ are the horizontal and vertical block indices, $\boldsymbol{\delta} = (\delta^x, \delta^y)$ a spatial displacement and $\mathbf{p} = (x, y)$ the coordinates within the block. For sake of simplicity, the notation $\mathbf{A}_{\mathbf{p}_b}$ will be used when no displacement is considered. For each block $\tilde{\mathbf{W}}_{\mathbf{p}_b}$ of the extracted ternary watermark, a search is carried out within an $m \times m$ search window ($m \geq n$) of the resynchronization mask $\mathbf{M}$, which is centered on the current block position. Of course, there exists a trade-off between the search window dimensions and the computational complexity: the greater m is, the larger are the distortions that can be compensated. However, this also increases the computational cost. Each block $\mathbf{M}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}$ within the search window is then considered as a candidate block and the goal is then to find the one which minimizes a given cost function. Once this best candidate block has been identified ($\boldsymbol{\delta} = \boldsymbol{\delta}_{\mathbf{p}_b}$), it is recopied in the resynchronized mask $\tilde{\mathbf{M}}$ at the coordinates of the block $\tilde{\mathbf{W}}_{\mathbf{p}_b}$. The resulting mask is then exploited to extract the embedded payload from the extracted ternary watermark $\tilde{\mathbf{W}}$ as described in Subsection 3.2.

4.2.1 Cost function

Since BM is used here to estimate geometric distortions using resynchronization bits, the following cost function is used:

$$\mathrm{C}\Big(\tilde{\mathbf{W}}_{\mathbf{p}_b}, \mathbf{M}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}\Big) = \frac{1}{n^2} \sum_{\mathbf{p} \in \mathcal{M}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}} \Phi\Big(\tilde{\mathbf{W}}_{\mathbf{p}_b}(\mathbf{p}), \mathbf{M}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}(\mathbf{p})\Big)$$

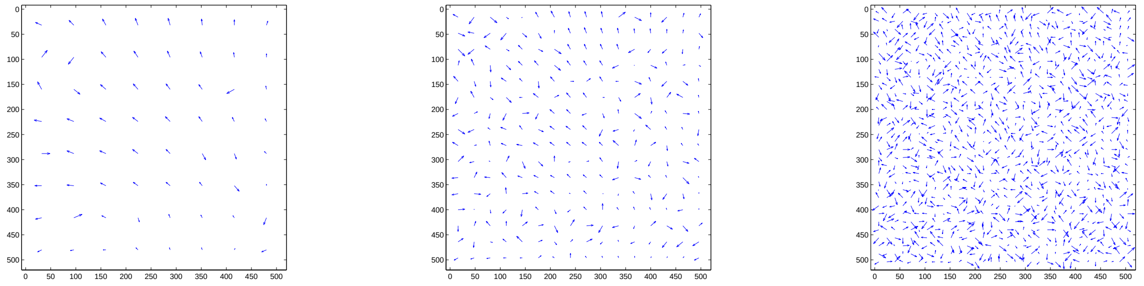
$$\text{with} \quad \Phi\big(a, b\big) = \begin{cases} 1, & \text{if } ab = -1 \\ 0.5, & \text{if } ab = 0 \\ 0, & \text{if } ab = 1 \end{cases} \qquad (15)$$

where $\mathcal{M}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}$ is the set of positions in the block $\mathbf{M}^{(\boldsymbol{\delta})}_{\mathbf{p}_b}$ associated with resynchronization bits. This formula adds a penalty each time that the resynchronization mask and the extracted watermark mismatch. Furthermore, when the watermark bit is unknown, i.e. $\tilde{\mathbf{W}}_{\mathbf{p}_b}(\mathbf{p}) = 0$, a cost of 0.5 is introduced to penalize blocks without enough valid resynchronization bits.

4.2.2 Shortcomings

As for any BM based process, the block size has a great influence on the performances of the resynchronization process. On one hand, small blocks are likely not to contain enough resynchronization bits to enable a correct registration and thus compensate for local geometric distortions. On the other hand, considering large blocks prevents from estimating finely the geometric distortions. This phenomenon is depicted in Figure 5. A 512×512 pixels image has been watermarked with Eurémark as described in Section 3. Next, the watermarked image has been submitted to the random bending attack. The resynchronization process has then been run with different block sizes and the estimated optical flows have been retrieved. It is clear that considering large blocks improves the resynchronization process. In fact, with 16×16 blocks, an almost random motion field is obtained and payload extraction is no longer possible.

The second issue with the current resynchronization process is that it operates blindly in a best match fashion. There is no constraint at all between the displacements $\boldsymbol{\delta}_{\mathbf{p}_b}$ of neighbor blocks. This is of course suboptimal. The visual quality of an image distorted by a geometric distortion is indeed determined by its homogeneity. The less homogenous the distortion, the



(a) 64×64 *blocks* (b) 32×32 *blocks* (c) 16×16 *blocks*

Figure 5. *Influence of the block size on the efficiency of the resynchronization process when BM is used. The watermarked image has been submitted to the random bending attack. It is clear that when the block size decreases too much, the estimated displacements are less relevant.*

worse the visual quality is [12]. In other terms, neighbor displacements are likely to be correlated to a certain extend so that the resulting overall distortion remains tolerable. However this knowledge is still not exploited during the matching process. As a result, nothing ensures that the estimated optical flow will even belong to the possible set of transformations. For instance, in Figure 5, with 16 × 16 blocks, the resynchronization process outputs an estimation of the optical flow which cannot have been applied in practice for visibility reasons.

4.3 Resynchronization enhancement

To overcome those shortcomings, the registration process is slightly modified to constrain the coherency of the estimated optical flow. First, considering some related work in object recognition, a rigidity parameter is introduced in the resynchronization framework to ensure the smoothness of the optical flow. Additionally, a multi-scale approach is also considered to obtain denser motion fields.

4.3.1 Elastic graph matching

In the context of object recognition, the procedure usually consists of two steps. First, for each reference object in the database, the object to be recognized is optimally aligned with the reference one. Second, the reference and aligned objects are compared to assert whether the input object is recognized as the reference one or not. This is somewhat similar to the situation in digital watermark: resynchronization can be regarded as the alignment process and watermark detection as the comparison. This analogy has consequently motivated the review of pattern recognition algorithms to see whether one of them can be easily introduced in our resynchronization framework. Elastic Graph Matching (EGM) [29] has particularly hold our attention. This algorithm performs a BM procedure with a smoothness constraint: a rigidity parameter is introduced to prevent displacements of neighbor blocks from being incoherent.

Cost function. Elastic Graph Matching aims at finding the set of block displacements $\boldsymbol{\Delta} = \{\boldsymbol{\delta}_{\mathbf{p}_b}\}$ which minimizes the following global cost function:

$$\mathrm{C}_{\mathrm{total}}(\tilde{\mathbf{W}},\mathbf{M}) = \mathrm{C}_{\mathrm{match}}(\tilde{\mathbf{W}},\mathbf{M},\boldsymbol{\Delta}) + \lambda.\mathrm{C}_{\mathrm{smooth}}(\boldsymbol{\Delta}) \qquad (16)$$

where the parameter λ controls the *rigidity* of the estimated optical flow and $\mathrm{C}_{\mathrm{match}}(.)$ and $\mathrm{C}_{\mathrm{smooth}}(.)$ are two cost functions. The first one is a matching cost function which indicates how well correspond the extracted watermark $\tilde{\mathbf{W}}$ and the resynchronization mask $\mathbf{M}$ considering only the resynchronization bits and assuming that block displacements are given by $\boldsymbol{\Delta}$. It can thus be defined as follows:

$$\mathrm{C}_{\mathrm{match}}(\tilde{\mathbf{W}},\mathbf{M},\boldsymbol{\Delta}) = \sum_{x_b=1}^{X_b}\sum_{y_b=1}^{Y_b} \mathrm{C}\left(\tilde{\mathbf{W}}_{\mathbf{p}_b}, \mathbf{M}_{\mathbf{p}_b}^{(\boldsymbol{\delta}_{\mathbf{p}_b})}\right) \qquad (17)$$

where X_b (resp. Y_b) is the number of blocks along the horizontal (resp. vertical) axis and $\mathrm{C}(.)$ the cost function defined in Equation (15). It should be noted that when the rigidity parameter λ is set to 0, the resynchronization process is equivalent to the previous BM procedure: minimizing the whole cost function $\mathrm{C}_{\mathrm{total}}(.)$ comes down to minimizing independently each block cost $\mathrm{C}\left(\tilde{\mathbf{W}}_{\mathbf{p}_b}, \mathbf{M}_{\mathbf{p}_b}^{(\boldsymbol{\delta}_{\mathbf{p}_b})}\right)$. A second term is consequently added in Equation (16) to ensure some kind of smoothness for the estimated optical flow. To this end, the sum of the distance between displacements of neighbor blocks is computed considering only the four nearest neighbors:

$$\begin{aligned} \mathrm{C}_{\mathrm{smooth}}(\boldsymbol{\Delta}) &= \sum_{x_b=1}^{X_b-1}\sum_{y_b=1}^{Y_b} \left\|\boldsymbol{\delta}_{x_b,y_b} - \boldsymbol{\delta}_{x_b+1,y_b}\right\|^2 \\ &+ \sum_{x_b=1}^{X_b}\sum_{y_b=1}^{Y_b-1} \left\|\boldsymbol{\delta}_{x_b,y_b} - \boldsymbol{\delta}_{x_b,y_b+1}\right\|^2 \end{aligned} \qquad (18)$$

where $\|.\|$ is the Euclidean distance. This smoothing cost function interferes with the BM process to permit blocks that are not the *best* matching ones to be still considered in case they are coherent with the current estimation of the optical flow $\boldsymbol{\Delta}$. Furthermore, it should be noted that when the rigidity parameter λ is set to infinity, the resynchronization process comes down to rigid alignment i.e. all the block displacements $\boldsymbol{\delta}_{\mathbf{p}_b}$ are bound to be equal.

```
1  Initialize $\mathbf{\Delta}$ with the estimation of the optical flow
   given by the BM resynchronization process
2  Set $flag = 1$ and $numIter = 0$
3  While $flag = 1$ and $numIter \leq maxIter$,
     (a) Set $flag = 0$
     (b) Scan the blocks sequentially in a random order
         i. Visit all the blocks in the search window and
            compute the associated increase of cost $\mathrm{dC}_{\boldsymbol{\delta}_{\mathbf{p}_b}}^{\boldsymbol{\delta}'_{\mathbf{p}_b}}$
         ii. If the displacement which minimizes $\mathrm{dC}_{\boldsymbol{\delta}_{\mathbf{p}_b}}^{\boldsymbol{\delta}'_{\mathbf{p}_b}}$
            decreases the global cost, update the optical
            flow $\mathbf{\Delta}$ and set $flag = 1$
     (c) Increment $numIter$
```

Table 1. *Iterative process of the Elastic Graph Matching algorithm.*

Iterative procedure. Once the cost function has been defined, a procedure has to be designed to find the optical flow $\mathbf{\Delta}$ which minimizes it. In this work, an iterative procedure is used which is sure to terminate in a local minimum of the cost function. First the BM based resynchronization process described in Section 4.2 is launched to obtain an initial estimation of the optical flow ($\lambda = 0$). The rigidity parameter is then set to a finite value and the goal is to update iteratively the optical flow so that the cost function decreases. Thus, for each iteration, all the blocks are scanned sequentially in a random order. For each block, if the current displacement estimation $\boldsymbol{\delta}_{\mathbf{p}_b}$ is updated with $\boldsymbol{\delta}'_{\mathbf{p}_b}$, this increases the total cost by a the following value:

$$\begin{aligned}\mathrm{dC}_{\boldsymbol{\delta}_{\mathbf{p}_b}}^{\boldsymbol{\delta}'_{\mathbf{p}_b}} = &\mathrm{C}\Big(\tilde{\mathbf{W}}_{\mathbf{p}_b}, \mathbf{M}_{\mathbf{p}_b}^{(\boldsymbol{\delta}'_{\mathbf{p}_b})}\Big) - \mathrm{C}\Big(\tilde{\mathbf{W}}_{\mathbf{p}_b}, \mathbf{M}_{\mathbf{p}_b}^{(\boldsymbol{\delta}_{\mathbf{p}_b})}\Big)\\ &+ \lambda\Big[\, \|\boldsymbol{\delta}'_{x_b,y_b} - \boldsymbol{\delta}'_{x_b,y_b-1}\|^2 - \|\boldsymbol{\delta}_{x_b,y_b} - \boldsymbol{\delta}_{x_b,y_b-1}\|^2\\ &\quad + \|\boldsymbol{\delta}'_{x_b,y_b} - \boldsymbol{\delta}'_{x_b,y_b+1}\|^2 - \|\boldsymbol{\delta}_{x_b,y_b} - \boldsymbol{\delta}_{x_b,y_b+1}\|^2\\ &\quad + \|\boldsymbol{\delta}'_{x_b,y_b} - \boldsymbol{\delta}'_{x_b-1,y_b}\|^2 - \|\boldsymbol{\delta}_{x_b,y_b} - \boldsymbol{\delta}_{x_b-1,y_b}\|^2\\ &\quad + \|\boldsymbol{\delta}'_{x_b,y_b} - \boldsymbol{\delta}'_{x_b+1,y_b}\|^2 - \|\boldsymbol{\delta}_{x_b,y_b} - \boldsymbol{\delta}_{x_b+1,y_b}\|^2\,\Big]\end{aligned} \tag{19}$$

This equation may even have fewer terms if the considered block lies on the border of the image. Considering all the blocks in the search window associated with the current block, the goal is consequently to find the displacement $\boldsymbol{\delta}'_{x_b,y_b}$ which minimizes the cost increase $\mathrm{dC}_{\boldsymbol{\delta}_{\mathbf{p}_b}}^{\boldsymbol{\delta}'_{\mathbf{p}_b}}$. In the worst case, no better displacement is identified and the optical flow remains untouched. Otherwise, the optical flow is updated ($\boldsymbol{\delta}'_{\mathbf{p}_b} \rightarrow \boldsymbol{\delta}_{\mathbf{p}_b}$) to obtain a motion field with a lower cost. The algorithm terminates when all the blocks have been visited without any modification or when more than $numIter$ iterations have been done. The overall process is summarized in Table 1.

4.3.2 Multi-scales approach

The algorithms which iteratively aim at minimizing some cost function all share the same drawback: the iterative process can get trapped in local minimum and the global minimum, which is usually the expected solution, can be missed. A multi-scales approach has consequently been superimposed over the current framework to be able to estimate dense optical flows. The basic idea is to start the matching process with large blocks and then to successively consider smaller blocks. Large blocks enable to find a relevant initial estimation of the optical flow since many resynchronization bits are available for each block. Then, the block size can be slowly decreased to refine the optical flow by permitting more geometric distortions. In practice, the current implementation starts with 64×64 blocks and keeps the block size constant until the EGM iterations terminates. Then, the block size is divided by 2 and the estimated optical flow $\mathbf{\Delta}$ is over-sampled since it is now twice denser. Next, the EGM algorithm is run again. This process is repeated until the EGM procedure terminates with 16×16 blocks.

Choice of $\boldsymbol{\lambda}$. The parameter λ controls the smoothness of the optical flow and has to be chosen carefully. The larger its value is, the more the block displacements $\boldsymbol{\delta}_{\mathbf{p}_b}$ are bound to be similar in the considered neighborhood. The proposed multi-scales approach calls for an adaptive rigidity parameter. The same value cannot be used for 64×64 blocks and 16×16 blocks. Indeed, in the latter case, neighbor blocks are nearer and thus neighbor displacements should be more similar i.e. the rigidity constraint should be stronger. The parameter λ should consequently be set so that it grows larger when the block size decreases. To this end, it is substituted in Equation (16) by λ/n^2 where n is the block size[2].

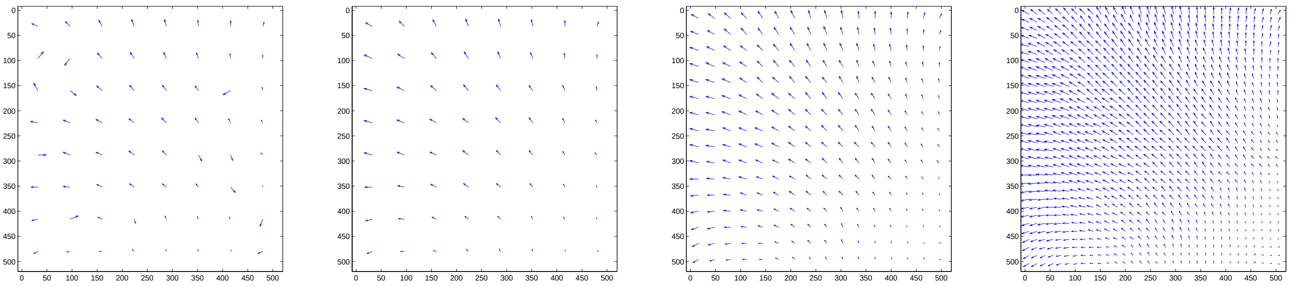
4.4 Further Possible Enhancements

The Figure 6 depicts how the estimation of the optical flow evolves during the EGM process. Looking at Figure 5 for comparison, this novel resynchronization process obviously outperforms the previous one. With EGM, the estimated optical flow is somewhat coherent with 16×16 blocks while it is almost random with BM based resynchronization. Nevertheless, a few tracks are proposed below to further investigate how to enhance the resynchronization process.

Knowledge about payload bits. Both methods consider resynchronization bits as anchor points to compensate for geometric distortions. The payload bits are completely ignored during the resynchronization process. However, at any moment, with the current estimation of the optical flow $\mathbf{\Delta}$, it is possible for each payload bit b_i to obtain an estimation of the probability p_i (resp. $1 - p_i$) that this bit is equal to 1 (resp. 0). An Expectation-Maximization (EM) framework can consequently introduced so that payload bits are also considered in the matching cost function defined in Equation (15) with respect to their current estimated probabilities.

Other pattern recognition algorithms. A relationship (even if artificial) has been established between digital watermarking and pattern recognition. This has motivated the introduction of Elastic Graph Matching to perform watermark resynchronization. However, if this

[2] The rigidity parameter has been set inversely proportional to n^2 rather than n because the *squared* Euclidean distance is considered in the smoothing cost function $\mathrm{C}_{\mathrm{smooth}}(.)$.



(a) *Iteration 1* (b) *Iteration 10* (c) *Iteration 16* (d) *Iteration 21*

Figure 6. *Illustration of the iterative estimation of the geometric distortions when EGM is used. The watermarked image has been submitted to the random bending attack.*

algorithm is recognized to be a reference one in pattern recognition, it is today 10 years old. Thus, it may be useful to review the state-of-the-art in pattern recognition to identify whether new and more efficient techniques have emerged.

Multi-scale vs. hierarchical. Currently, there is no relationship between the different scales. The estimation of the optical flow obtained at a given scale is simply used as an initialization for the next one. It may be interesting to insert some kind of dependencies across the levels. It is somewhat obvious that a block and its four children should share similar displacements. Such a hierarchical approach may enables to reduce the block size below 16×16, which is not possible now, and thus obtain a denser optical flow.

5 Experiments

In this report, upgrading the watermark detector is motivated by a potential gain in robustness against local geometric distortions. It is consequently reasonable to verify whether the improved detectors exhibit superior performances or not. Figure 6 illustrates that EGM enables to obtain a smoother estimation of the geometric distortions. Nevertheless, detection statistics have to be surveyed to investigate whether such dense, smooth displacement fields improve detection performances. Therefore, intensive benchmarking needs to be performed and in this perspective, StirMark - also referred to as Random Bending Attack (RBA) - is now recognized as an essential tool to evaluate robustness against local geometric distortions [37, 28].

5.1 StirMark

This benchmarking tool basically simulates a resampling process i.e. it introduces the same kind of errors into an image as printing it on a high quality printer and then scanning it again with a high quality scanner. To do so, each input image is processed in five steps.

Global bilinear transform: The input image is slightly stretched, sheared, shifted and/or rotated by an unnoticeable random amount. In practice, the corners of the image are moved by a small random amount in both directions and the other pixels are mapped so that their relative position remains the same. The impact of this step is guided by two parameters i and o. The first one sets the number of pixel distances that the corner of the target image is allowed to be *inside* the original image. It is set by default to 2% of the image dimensions. Similarly, o sets the number of pixel distances that the corner of the target image is allowed to be *outside* the original image. It is set by default to 0.7 and cannot be much higher since sample values taken from outside the original image are extrapolated.

Noise addition: A transfer function is applied to the image to introduce a small and smoothly distributed error into sample values. This emulates the small nonlinear analog/digital converter imperfections typically found in scanners and display devices. As geometric distortions are the main concern is this study, this step is discarded by setting the associated parameter d to zero.

Global bending: In addition to the general bi-linear distortion, a slight deviation is applied to each pixel, which is greatest at the center of the picture and almost null at the borders. The strength of the bending is given by the parameter b which sets the number of pixel displacement allowed for the center of the image. Its default value is set to 2.

Higher frequency displacement: An additional geometric distortion is added which has the form:

$$l \sin(\omega_x x) \sin(\omega_y y) + n(x, y) \qquad (20)$$

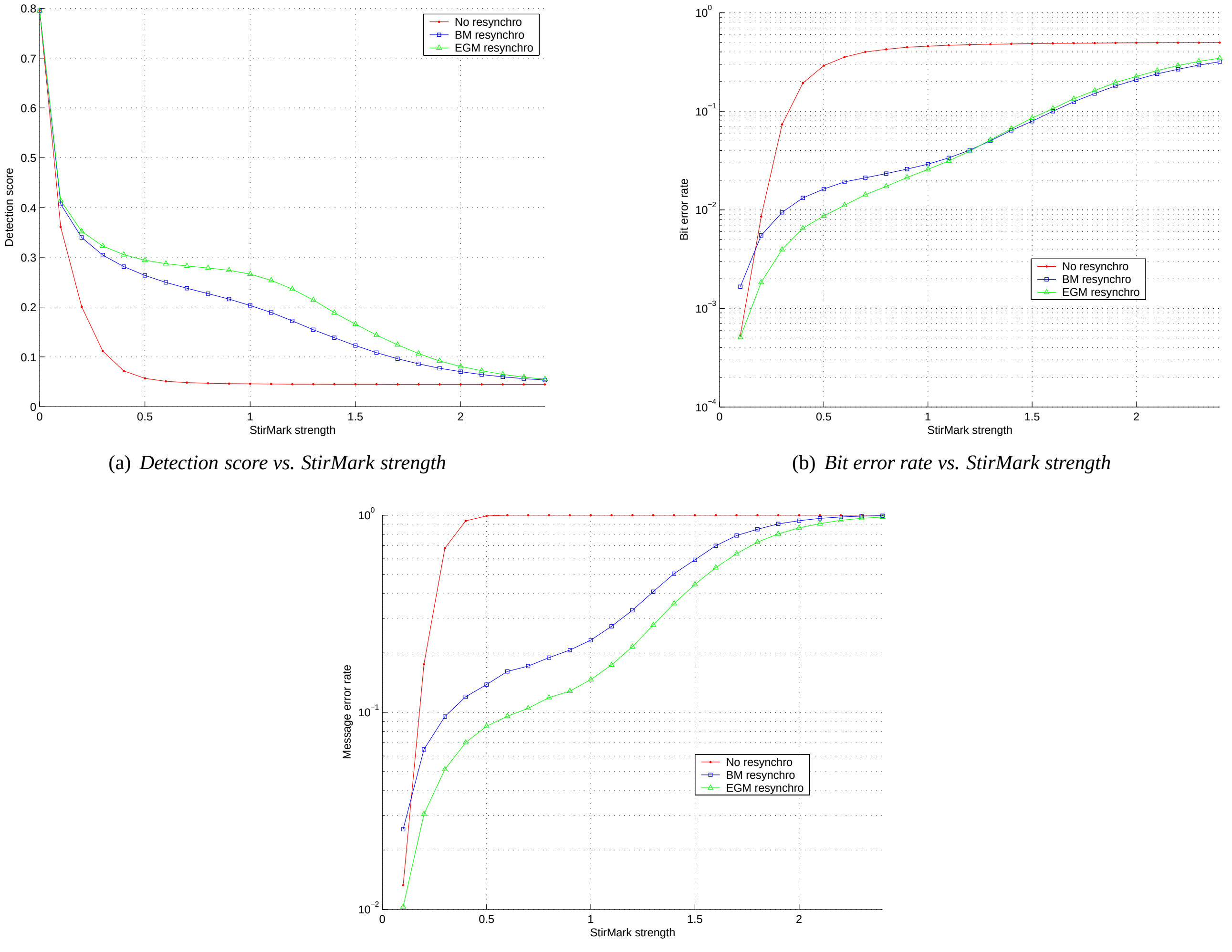
where n is a random number. This distortion is constrained by the parameter R which sets the fraction of pixel displacement allowed for any pixel. By default, the value 0.1 is used.

JPEG compression: A mild JPEG compression is then done since digital images are usually stored using this compression standard. Nevertheless, this step is also discarded because the focus of this study is geometric distortions.

In the remainder of this article, Θ_o will refer to a column vector containing the default values of the different StirMark parameters.

5.2 Experimental results

A database of 500 images of size 512×512 has been considered for experiments. It contains snapshots, syn-



(a) *Detection score vs. StirMark strength*

(b) *Bit error rate vs. StirMark strength*

(c) *Message error rate vs. StirMark strength*

Figure 7. *Robustness of the different presented detectors (no resynchronization, BM based resynchronization, EGM based resynchronization) against the random bending attack.*

thetic images, drawings and cartoons. All the images are first watermarked using the algorithm Eurémark described in Section 3. The threshold τ_{low} and τ_{high} are respectively set to 3 and 12 so that the embedding process results in a distortion equal to 38 dB. The payload is 64 bits long and randomly generated. Furthermore, 57 control bits are finely interlaced with the payload bits for resynchronization. This results in a $11 \times 11 = 121$ binary block which is duplicated before encryption and embedding. Next, those watermarked images are submitted to the StirMark attack with an increasing strength $\alpha > 0$. To this end, the attack is simply performed with the parameters $\Theta = \alpha\Theta_{\text{o}}$. On the detector side, three resynchronization methods are surveyed: no resynchronization, BM based resynchronization and EGM resynchronization. For each method and for each attacked image, the detection score defined in Equation (13), the Bit Error Rate (BER) and the Message Error Rate (MER) are computed. This experiment is performed 25 times with alternative random embedding keys. It results in $500 \times 25 = 12500$ curves which indicate the evolution of the detection score (or BER/MER) vs. the StirMark strength for a given image, a given embedding key and a given resynchronization method. All those curves are averaged and then reported in Figure 7.

As expected, Eurémark is quickly defeated when no resynchronization is performed on the detector side. The resynchronization process improves significantly the performances of the algorithm. Furthermore, the novel EGM based resynchronization module appears to slightly outperform the previous one based on BM only. Following the practice suggested in [28], the watermarking scheme is considered to be robust if at least 80% of the watermarks are correctly retrieved i.e. the MER is below 20%. The different schemes are respectively defeated for a StirMark strength equal to 0.2 with no resynchronization, 0.85 with BM resynchronization and 1.2 with EGM resynchronization. The improvement between BM and EGM resynchronization is due to two different aspects in the design of the resynchronization module. First, the rigidity constraint enables to correct incoherent estimated displacements as it has been depicted in Figure 6. Second, the multi-scales framework allows to better cope with small local distortions. For large StirMark strength, both schemes are defeated because the resynchronization procedure is limited to the size of the search window. Additionally, it is interesting to note that the curves for BM and EGM resynchronization do not vary regularly: it seems that there is a step somewhere in the middle. This reveals a weakness due to the fact that fractal coding is considered in this algorithm. Because of computational cost, the computation of the fractal cover is block based. Thus, geometric distortions disturb the alignment of the blocks and the fractal cover is not computed using exactly the same blocks. It could be possible to get round this weakness by considering overlapping blocks during cover computation. But this comes of course with additional computational cost.

6 Conclusion

A novel watermark registration technique based on EGM has been presented. It basically relies on the insertion of resynchronization bits which are finely interlaced with payload bits. Thus, those control bits can be used as anchor points to compensate for geometric distortions. Once the image has been realigned, the original watermark detection procedure can be performed. This new technique has been shown to outperform a similar previous technique which was simply based on block matching. Furthermore, since the resynchronization bits are encrypted, peaks do not appear in the frequency domain. As a result, the template cannot be removed thanks to existing attacks [47]. Future work will be devoted to the extension of this resynchronization framework to generic watermarking systems such as spread-spectrum watermarks [22]. Additionally, the potential benefits of the multi-scale approach will be further investigated to obtain denser and thus finer motion fields.

Acknowledgements

The authors want to thank Dr. Florent Perronnin for fruitful discussions on pattern recognition and valuable feedback regarding the implementation of the Elastic Graph Matching algorithm. They also want to thank Dr. Emmanuel Garcia for his appreciated comments. Finally, the authors acknowledge the European Commission for financial support through the IST Program under Contract IST-2002-507932 ECRYPT.

7 References

[1] M. Alghoniemy and A. Tewfik. Progressive quantized projection watermarking system. In *Proceedings of the ACM International Conference on Multimedia*, pages 295–298, November 1999.

[2] M. Alghoniemy and A. Tewfik. Geometric distortion correction through image normalization. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, volume III, pages 1291–1294, August 2000.

[3] M. Alghoniemy and A. Tewfik. Geometric distortions correction in image watermarking. In *Security and Watermarking of Multimedia Contents II*, volume 3971 of *Proceedings of SPIE*, pages 82–89, January 2000.

[4] M. Barni, F. Bartolini, R. Caldelli, and A. Piva. Geometric invariant robust watermarking through constellation matching in the frequency domain. In *Proceedings of the IEEE International Conference on Image Processing*, volume II, pages 65–68, September 2000.

[5] P. Bas, J.-M. Chassery, and B. Macq. Robust watermarking based on the warping of pre-defined triangular patterns. In *Security and Watermarking of Multimedia Contents II*, volume 3971 of *Proceedings of SPIE*, pages 99–109, January 2000.

[6] P. Bas and B. Macq. A new video-object watermarking scheme robust to object manipulation. In *Proceedings of the IEEE International Conference on Image Processing*, volume II, pages 526–529, October 2001.

[7] G. Braudaway and F. Mintzer. Automatic recovery of invisible image watermarks from geometrically distorted images. In *Security and Watermarking of Multimedia Contents II*, volume 3971 of *Proceedings of SPIE*, pages 74–81, January 2000.

[8] D. Coltuc and P. Bolon. Robust watermarking by histogram specification. In *Proceedings of the IEEE International Conference on Image Processing*, volume II, pages 236–239, October 1999.

[9] F. Davoine, P. Bas, P.-A. Hébert, and J.-M. Chassery. Watermarking et résistance aux déformations géométriques. In *Actes des Cinquième Journées d'Étude et d'Échanges sur la Compression et la Représentation des Signaux Audiovisuels*, June 1999.

[10] F. Deguillaume, S. Voloshynovskiy, and T. Pun. A method for the estimation and recovering from general affine transforms in digital watermarking applications. In *Security and Watermarking of Multimedia Contents IV*, volume 4675 of *Proceedings of SPIE*, pages 313–322, January 2002.

[11] D. Delannay, J.-F. Delaigle, B. Macq, and M. Barlaud. Compensation of geometrical transformations for watermark extraction in the digital cinema application. In *Security and Watermarking of Multimedia Contents III*, volume 4314 of *Proceedings of SPIE*, pages 149–157, January 2001.

[12] D. Delannay, I. Setyawan, R. Lagendijk, and B. Macq. Relevant modeling and comparison of geometric distortions in watermarking systems. In *Application of Digital Image Processing XXV*, volume 4790 of *Proceedings of SPIE*, pages 200–210, July 2002.

[13] J. Dittmann, T. Fiebig, and R. Steinmetz. A new approach for transformation invariant image and video watermarking in the spatial domain: SSP – self spanning patterns. In *Security and Watermarking of Multimedia Contents II*, volume 3971 of *Proceedings of SPIE*, pages 176–185, January 2000.

[14] P. Dong, J. Brankov, N. Galatsanos, and Y. Yang. Generic robust watermarking through mesh model based correction. In *Proceedings of the IEEE International Conference on Image Processing*, volume III, pages 493–496, September 2002.

[15] P. Dong and N. Galatsanos. Affine transformation resistant watermarking based on image normalization. In *Proceedings of the IEEE International Conference on Image Processing*, volume III, pages 489–492, September 2002.

[16] J.-L. Dugelay. Method for hiding binary data in a digital image. Pending Patent PCT/FR99/00485 (EURECOM 09-PCT), March 1999.

[17] J.-L. Dugelay and C. Rey. Method of marking a multimedia document having improved robustness. Pending Patent EP 99480075.3 (EURECOM 14 EP), May 2001.

[18] J.-L. Dugelay and S. Roche. Process for marking a multimedia document, such an image, by generating a mark. Pending Patent EP 99480075.3 (EURECOM 11/12 EP), July 1999.

[19] Y. Fisher. *Fractal Image Compression: Theory and Applications*. Springer-Verlag, 1994.

[20] D. Fleet and D. Heeger. Embedding invisible information in color images. In *Proceedings of the IEEE International Conference on Image Processing*, volume I, pages 532–535, October 1997.

[21] J. Haitsma and T. Kalker. A watermarking scheme for digital cinema. In *Proceedings of the IEEE International Conference on Image Processing*, volume II, pages 487–489, October 2001.

[22] F. Hartung and B. Girod. Watermarking of uncompressed and compressed video. *Signal Processing*, 66(3):283–301, May 1998.

[23] F. Hartung, J. Su, and B. Girod. Spread spectrum watermarking: Malicious attacks and counterattacks. In *Security and Watermarking of Multimedia Contents*, volume 3657 of *Proceedings of SPIE*, pages 147–158, January 1999.

[24] C. Honsinger and M. Rabbani. Data embedding using phase dispersion. In *Proceedings of PICS 2000: Image Processing, Image Quality, Image Capture, Systems Conference*, volume III, pages 264–268, March 2000.

[25] N. Johnson, Z. Duric, and S. Jajodia. Recovery of watermarks from distorted images. In *Proceedings of the Third International Workshop on Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 318–332, November 1999.

[26] M. Kutter. Watermarking resisting to translation, rotation and scaling. In *Multimedia Systems and Applications*, volume 3528 of *Proceedings of SPIE*, pages 423–431, November 1998.

[27] M. Kutter, F. Jordan, and F. Bossen. Digital watermarking of color images using amplitude modulation. *Journal of Electronic Imaging*, 7(2):326–332, April 1998.

[28] M. Kutter and F. Petitcolas. A fair benchmark for image watermarking systems. In *Security and Watermarking of Multimedia Contents*, volume 3657 of *Proceedings of SPIE*, pages 226–239, January 1999.

[29] M. Lades, J. Vorbrüggen, J. Buhnmann, J. Lange, C. Malsburg, R. Würtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on Computers*, 42(3):300–311, March 1993.

[30] J. Lichtenauer, I. Setyawan, T. Kalker, and R. Lagendijk. Exhaustive geometrical search and the false positive watermark detection probability. In *Security and Watermarking of Multimedia Contents V*, volume 5020 of *Proceedings of SPIE*, pages 203–214, January 2003.

[31] C. Licks, F. Ourique, R. Jordan, and F. Pérez-González. The effect of the random jitter attack on the bit error rate performance of spatial domain image watermarking. In *Proceedings of the IEEE International Conference on Image Processing*, volume II, pages 455–458, September 2003.

[32] C.-Y. Lin, M. Wu, J. Bloom, I. Cox, M. Miller, and Y. Lui. Rotation, scale and translation resilient watermarking for images. *IEEE Transactions on Image Processing*, 10(5):767–782, May 2001.

[33] P. Loo and N. Kingsbury. Motion estimation based registration of geometrically distorted images for watermark recovery. In *Security and Watermarking of Multimedia Contents III*, volume 4314 of *Proceedings of SPIE*, pages 606–617, January 2001.

[34] A. Nikolaidis and I. Pivas. Robust watermarking of facial images based on salient geometric pattern matching. *IEEE Transactions on Multimedia*, 2(3):172–184, September 2000.

[35] I. Ozer, M. Ramkumar, and A. Akansu. A new method for detection of watermarks in geometrically distorted images. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume IV, pages 1963–1966, June 2000.

[36] S. Pereira and T. Pun. Fast robust template matching for affine resistant image watermarking. In *Proceedings of the Third International Workshop on Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 200–210, September 1999.

[37] F. Petitcolas, R. Anderson, and M. Kuhn. Attacks on copyright marking systems. In *Proceedings of the Second International Workshop on Information Hiding*, volume 1525 of *Lecture Notes in Computer Science*, pages 219–239, April 1998.

[38] C. Rey, K. Amis, J.-L. Dugelay, R. Pyndiah, and A. Picart. Enhanced robustness in image watermarking using block turbo codes. In *Security and Watermarking of Multimedia Contents V*, volume 5020 of *Proceedings of SPIE*, January 2003.

[39] J. Ó. Ruanaidh and T. Pun. Rotation, scale and translation invariant digital image watermarking. *Signal Processing*, 68(3):303–317, May 1998.

[40] I. Setyawan and R. Lagendijk. Human perception of geometric distortions in images. In *Security, Steganography and Watermarking of Multimedia Contents VI*, volume 5306 of *Proceedings of SPIE*, pages 256–267, January 2004.

[41] V. Solachidis and I. Pitas. Circularly symmetric watermark embedding in 2D DFT domain. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume VI, pages 3469–3472, March 1999.

[42] K. Su, D. Kundur, and D. Hatzinakos. A novel approach to collusion resistant video watermarking. In *Security and Watermarking of Multimedia Contents IV*, volume 4675 of *Proceedings of SPIE*, pages 491–502, January 2002.

[43] P.-C. Su and C.-C. Kuo. Synchronized detection of the block-based watermark with invisible grid embedding. In *Security and Watermarking of Multimedia Contents III*, volume 4314 of *Proceedings of SPIE*, pages 406–417, January 2001.

[44] Q. Sun, J. Wu, and R. Deng. Recovering modified watemrarked image with reference to original image. In *Security and Watermarking of Multimedia Contents*, volume 3657 of *Proceedings of SPIE*, pages 415–424, January 1999.

[45] P. Termont, L. D. Stycker, J. Vandewege, M. O. de Beeck, J. Haitsma, T. Kalker, M. Maes, and G. Depovere. How to achieve robustness against scaling in a real-time digital watermarking system for broadcast monitoring. In *Proceedings of the IEEE International Conference on Image Processing*, volume 1, pages 407–410, September 2000.

[46] A. Tirkel, C. Osborne, and T. Hall. Image and watermark registration. *Signal Processing, Special Issue on Watermarking*, 66(3):377–384, May 1998.

[47] S. Voloshynovskiy, A. Herrigel, and Y. Rystar. The watermark template attack. In *Security and Watermarking of Multimedia Contents III*, volume 4314 of *Proceedings of SPIE*, pages 394–405, January 2001.