AWGN Watermark in Images and E-Books – Optimal Embedding Strength

Vesna Vučković[1] and Bojan Vučković[2]
[1]*Faculty of Mathematics, University of Belgrade*
[2]*Serbian Academy of Sciences and Arts, Belgrade*
Serbia

1. Introduction

During the last twenty years, a lot of methods for grayscale images robust watermarking were proposed.

One important watermarking algorithms class is based on spread spectrum technique (Cox et al., 1997, 2008; Feng et al., 2005, 2006; Mora-Jimenez & Navia-Vazquez, 2000; Perez-Freire & Perez-Gonzalez, 2009; Ruanaidh & Pun, 1998; Ruanaidh & Csurka, 1999) – embedding by spreading the information about each message bit across several image pixels, or across the whole image. Watermark embedding in such manner is usually performed by adding of some pseudorandom data vector to the image.

Among such algorithms an interesting class form those based on additive white Gaussian noise (AWGN) embedding, with blind detection, based on correlation between image and embedded message. In (Cox et al., 2008) authors devoted a lot of space to these techniques. In our text for them we use the term *AWGN watermarks*.

Important property of digital watermark, dedicated to copyright protection, is its *robustness*. The watermark needs to remain detectable after common image operations (lossy compression, contrast/brightness changing, cropping, rotation...). In this text we call these operations (*image*) *modifications* (or *distortions*).

AWGN watermarking algorithm exhibits good robustness characteristics against many image modifications.

Clearly, **not every individual watermark embedded by robust technique – robust:** for this, it needs to be embedded **strongly enough**. However, strongly embedded watermark will damage image quality. It is important to find a real measure for embedding strength.

This is our text theme: for AWGN watermarking algorithm, **we determine** *optimal* **embedding strength** - minimal one that ensures message detectability after expected modification.

Important example is determining optimal strength if modification is lossy compression.

Namely, saving watermarked image in lossy compressed format inevitably destroys part of watermark data. That's why the big part of this text is dedicated to lossy compression.

Practically all results and derived formulas here are originally ours, and they are published already in (Vučković, 2008, 2010a, 2010b, 2011).

Sections 2-4 are dedicated to the simplest AWGN algorithm – *spatial domain* embedding. In section 2 we briefly outline AWGN watermark algorithm in spatial domain, described in (Cox et al., 2008). In section 3 we discover optimal strength for effective embedding in spatial domain. Section 4 deals with determining necessary embedding strength for message to survive the expected lossy compression.

In section 5 we consider the same algorithm within the transform domain. We show that optimal strength is equal in spatial and transform domain if watermark is embedded over the whole image. Then, we briefly speak about embedding in some image coefficients, in block DCT transform domain.

In section 6 we give a short analysis of embedding strength determining for other image modifications.

Section 7 deals with color images and e-books watermarking.

Section 8 concludes this text.

It is assumed that our text reader has some prior knowledge – especially from linear algebra, image processing – including lossy compression, and probability theory – normal distribution.

2. AWGN watermark algorithm

Here, we briefly outline AWGN watermark algorithm, as it is described in (Cox et al., 2008) (spatial domain embedding).

Later, we will analyze other variants of this algorithm too (embedding in different domains and image coefficients). However, this stays in essence the same algorithm: we add white Gaussian noise to image (or subimage) in some domain; we test the correlation in the same domain in which the watermark is embedded.

2.1 Embedding

Inputs into embedding program are the original image, the message bit and the watermark key. The *watermark key* is a secret number, same for the embedder and the detector. Embedder output is the watermarked image.

We present the grayscale image c_0, of $m \times n$ pixels, with one $m \times n$ matrix or (same in fact) with one mn vector. In this text we will use terms *image vector*, *image matrix* and *image* as synonyms (if this doesn't make confusion).

We embed one message bit (binary one or zero) into the image according to formulas:

$$\begin{aligned} c_{w1} &= c_0 + \alpha \cdot r_w \\ c_{w0} &= c_0 - \alpha \cdot r_w \end{aligned} \tag{1}$$

Image vectors c_{w1} and c_{w0} arise from embedding of binary one or binary zero into the image, c_0 is the original image vector before embedding, α is the embedding strength coefficient ($\alpha>0$) and r_w is the reference pattern.[1]

The *reference pattern* is pseudorandom vector chosen in accordance with N(0,1) (*standard normal distribu1tion*), of the same dimension as c_0. In its generation, watermark key is used as the pseudorandom generator seed.

2.2 Detection

Inputs to the detector are the image and the watermark key (the same as in the embedder). Its output is the detected watermark message. As a detection measure, we use a linear correlation between the image and the reference pattern:

$$\mathrm{lc}(c, r_w) = \frac{c \bullet r_w}{\|r_w\| \cdot \|r_w\|} \tag{2}$$

($\bullet$ denotes the dot product, and $\| \ \|$ - the vector norm).

The detector compares the computed linear correlation value with the threshold value τ, set in advance, and replies that it is embedded:

$$\begin{array}{llll} - & \text{binary one} & \text{if} & \mathrm{lc}(c, r_w) \geq \tau \\ - & \text{binary zero} & \text{if} & \mathrm{lc}(c, r_w) \leq -\tau \\ - & \text{nothing} & \text{if} & |\mathrm{lc}(c, r_w)| < \tau \end{array} \tag{3}$$

2.3 Embedding of longer message

We embed a k bits message into the image by repeatedly (k times) embedding one bit. We name this *embedding of message bits one over another*.

An alternative solution is *embedding into subimages*: we divide the image into disjoint subimages, and then we embed one bit message into each subimage. Under *subimage* term we mean any subset of image pixels set. Subimages may, but need not, be composed of adjacent pixels.

Different variants of the stated methods are also possible. For example, we may embed several message bits into every subimage.

2.4 Optimal embedding strength

It is not easy to produce good imperceptible digital watermark, because it needs to satisfy two opposite requests:

[1] More precisely, resultant values in matrices must be from permissible set of image pixels values. Therefore, it is more precise to say $c_{w1}=[c_0+\alpha\cdot r_w]_8$ and $c_{w0}=[c_0-\alpha\cdot r_w]_8$. Sign $[\ ]_8$ designates "squeezing" of the result vector coordinates into 8-bits values, i.e. into values from the set {0,1,2...255}. However here, for the purpose of simplicity, we will ignore this (small!) diference.

- *Detectability* – in any case it needs to be detectable
- *Fidelity* – it needs to be imperceptible for casual observer.

These two demands are mutually confronted. If watermark is embedded stronger, then it's more likely it will be detectable. In the other hand, if watermark is strongly embedded, it will be more noticeable. Detectability is surely more important condition – "essential requirement". Thus, we need to determine *optimal* embedding strength – minimal one that guaranties detectability.

In some situations it may be expected that watermarked image will be subjected (from embedding to detection moment) to some modification. Under these circumstances it is also useful to know optimal strength – minimal one that ensures watermark detectability after this (expected) modification.

Two concepts are closely related to watermark detectability: effective and robust embedding.

Watermark is embedded *effectively* if it is detectable immediately after embedding.

If message is detectable in digital image that is after embedding subjected to some modification, we say the watermark is *robust* against undergone modification.

3. Embedding effectiveness

Embedding strength coefficient α and detection threshold τ directly influence the embedding effectiveness.

Coefficient α needs to be big enough (effective embedding), but not too big (quality request).

If we reduce the detection threshold τ (set it to be close to, or maybe equal to zero), embedding will be more effective (the detector will report in higher percent of embedding cases the image as watermarked). But, if the threshold τ is too small, the detector may reply that the image is watermarked, even when this is not true.

It is very important to find the real measure – to set the threshold τ and the coefficient α, in a way that *false negative* probabilities (non-effective embedding) and *false positive* ones (case when the detector reports that image is watermarked when this is not true) are acceptably low. Also, watermark must not be embedded too strongly to have the fidelity affected.

In subsection 3.1, we determine optimal value of coefficient α for effective embedding of one bit message. In subsection 3.2, we determine it for a longer message.

3.1 Effective embedding of one bit message

3.1.1 Deviation of $Ic(c_0,r)$ from zero – parameter σ_{Ic}

In searching for optimal embedding strength, we begin with well known facts for normal distribution:

- If X_1 and X_2 are normally distributed random variables: $X_1 \sim N(\mu_1,\sigma_1^2)$, $X_2 \sim N(\mu_2,\sigma_2^2)$, then their linear combination is also normally distributed:

$$aX_1+bX_2\sim N(a\mu_1+b\mu_2,a^2\sigma_1^2+b^2\sigma_2^2) \quad (4)$$

- For distribution $N(\mu, \sigma^2)$, the *68–95–99,7 rule* (*empirical rule, 3-sigma rule*) states that nearly all values drawn from it lie within 3 standard deviations of the mean:

$$\begin{aligned} &- \text{ within interval } (\mu-\sigma,\mu+\sigma) && \text{are about 68\% of values,} \\ &- \text{ within interval } (\mu-2\cdot\sigma,\mu+2\cdot\sigma) && \text{are about 95\% of values,} \\ &- \text{ within interval } (\mu-3\cdot\sigma,\mu+3\cdot\sigma) && \text{are about 99,7\% of values} \end{aligned} \quad (5)$$

- If X_1, X_2, ..., X_k are independent *standard normal* random variables (i.e. $X_i\sim N(0,1)$, $i=1,\ldots,k$), then the sum of their squares is χ^2 (chi-square) variable with k degrees of freedom. Its mathematical expectation is k.

In following text, we'll use symbols for reference patterns and images:

- r – an arbitrary reference pattern
- r_w – reference pattern that is to be embedded
- c_0 – original image (or image where r_w is not embedded)
- c – image – input into the detector (r_w may, but need not be embedded in it)

All coordinates of $r=(r(1),r(2),\ldots,r(mn))$ are drawn from distribution N(0,1). Mathematical expectation for $\|r\|$ is $\sqrt{mn}$. For image $c_0=(c_0(1),c_0(2),\ldots,c_0(mn))$ and reference pattern r, linear correlation is

$$lc(c_0,r)=\frac{c_0\bullet r}{\|r\|^2}=\frac{\sum_{i=1}^{mn} c_0(i)\cdot r(i)}{mn} \quad (6)$$

Therefore $lc(c_0,r)$, as linear combination of random variables $r=(r(1),r(2),\ldots,r(mn))$ has values from distribution $N(0,\sigma_{lc}^2)$, where:

$$\sigma_{lc}=\frac{\sqrt{(c_0(1))^2+(c_0(2))^2+\ldots+(c_0(mn))^2}}{mn}=\frac{\sqrt{E(c_0)}}{mn} \quad (7)$$

Value $E(c_0)= (c_0(1))^2+(c_0(2))^2+\ldots+(c_0(mn))^2$ is called *energy of image* c_0.

Then,

- in interval $(-\sigma_{lc}, \sigma_{lc})$ are 68% of linear correlation values,
- in interval $(-2\cdot\sigma_{lc},2\cdot\sigma_{lc})$ are 95% of linear correlation values,
- in interval $(-3\cdot\sigma_{lc},3\cdot\sigma_{lc})$ are almost all of linear correlation values.

3.1.2 Parameter α setting

According to empirical rule, it is $|lc(c_0,r_w)|\leq 3\cdot\sigma_{lc}$ with probability 0.997 ("almost 1").

Parameter σ_{lc} - the standard deviation of a sample (linear correlations between the original image and reference patterns) **is the basis for correct setting of parameters α and τ**, for effective embedding.

The biggest embedding strength is needed

- when $lc(c_0,r_w)=-3\cdot\sigma_{lc}$ and we embed binary one, and
- when $lc(c_0,r_w)=3\cdot\sigma_{lc}$ and we embed binary zero

Thus, if we choose

$$\alpha=3\cdot\sigma_{lc}+\tau \tag{8}$$

then almost every reference pattern will be effectively embedded with strength α.

Such embedding, in which α is set in advance, and the reference pattern is embedded with this strength (without consideration for linear correlation value between the image and the reference pattern that **is to be embedded**), we call a *fixed strength embedding*.

The fixed strength embedding is quite bigger than necessary. Better result we get if we consider the linear correlation between the image and the pattern that is to be embedded. In this case, we talk about an algorithm with the *embedding strength adjustment*.

Let $l_0=lc(c_0,r_w)$ and we embed the binary one. If $l_0<\tau$, we set $\alpha=\tau-l_0$. If $l_0\geq\tau$, we set $\alpha=0$ (not necessary to embed anything). Thus,

$$\begin{aligned}&- \quad \text{if we embed binary one, then } \alpha=\max\left(\tau-l_0,0\right)\\ &- \quad \text{if we embed binary zero, then } \alpha=\max\left(\tau+l_0,0\right)\end{aligned} \tag{9}$$

Embedding with strength α makes in image *mean square error*

$$\mathrm{MSE}(c_0,c_w)=\frac{1}{mn}\sum_{i=1}^{mn}[c_w(i)\text{-}c_0(i)]^2=\frac{1}{mn}\sum_{i=1}^{mn}\alpha^2\left[r_w(i)\right]^2=\alpha^2 \tag{10}$$

3.1.3 Detection threshold τ setting

To avoid false positive error, τ needs to be bigger than the biggest correlation between the image and not embedded reference patterns.

If we set $\tau=3\cdot\sigma_{lc}$, the linear correlation between the original image (c_0) and reference pattern would be in interval $(-\tau,\tau)$, almost with probability p=1. In other words, we could be almost 100% sure that the detector would not respond with a false positive error.

The threshold should not be larger than $3\cdot\sigma_{lc}$: this is not only needless, but also affects the fidelity (a stronger embedding would be needed for the detector to recognize the message).

We have to consider the possibility of a false positive error when the message is quite short (as here, the message being one bit long). If it is longer, the false positive error is not a big problem (we will explain more about this topic in subsubsection 3.2.1), and τ could be set quite smaller.

3.1.4 Parameter σ_{lc} dependence on the image dimension

The image c_k derived from k equal images, c, with energies E(c), has energy $E(c_k)=k\cdot E(c)$ and dimension $\dim(c_k)=k\cdot\dim(c)$. Its σ_{lc} parameter is

$$\sigma_{lc}(c_k) = \frac{\sqrt{E(c_k)}}{\dim(c_k)} = \frac{\sqrt{k \cdot E(c)}}{k \cdot \dim(c)} = \frac{\sqrt{E(c)}}{\sqrt{k} \cdot \dim(c)} = \frac{1}{\sqrt{k}} \cdot \sigma_{lc}(c) \tag{11}$$

So, parameter σ_{lc} is lower for larger images. Therefore, in an image with larger dimension, we have to embed our message with a lower strength. In the previous case, it is

$$\alpha_k = \alpha/\sqrt{k} \text{ and } \tau_k = \tau/\sqrt{k} \tag{12}$$

Taking the fidelity of image into account, it is obvious that in larger image we can embed a longer message.

3.2 Effective embedding of longer message

3.2.1 Threshold τ setting for longer message

The threshold τ for a longer message may be smaller than $3\cdot\sigma_{lc}$ (value recommended in case of one bit message). For example, for $\tau=\sigma_{lc}$, for roughly 68% of possible reference patterns, the linear correlation between the image and the reference pattern is inside the interval $(-\tau,\tau)$. So, the false positive may appear in 'every third' case. If we insist that every message bit must be detected for confirmation of message presence, it will be almost impossible to detect the message if it is not embedded (for detection of non-existing message, it is necessary for all reference patterns to arise a false positive, and in a longer message this is almost impossible).

So, in the case of longer message it is permitted a quite smaller τ value.

3.2.2 Message bits embedding one over another

Due to the fact that reference patterns are mutually uncorrelated, embedding of new pattern will not substantially change linear correlation between the image and previously embedded patterns (linear correlation is resistant against white Gaussian noise):

$$lc(c_0 + r_1 + r_2, r_1) = \frac{(c_0 + r_1) \bullet r_1 + r_2 \bullet r_1}{\|r_1\| \cdot \|r_1\|} \approx \frac{(c_0 + r_1) \bullet r_1 + 0}{\|r_1\| \cdot \|r_1\|} = lc(c_0 + r_1, r_1) \tag{13}$$

Hence, if we embed message bits one over another, we can take same value for α as in the case of only one information bit.[2]

We embed k bits into image c_0=($c_0(1)$, $c_0(2)$,..., $c_0(mn)$), in a way that for each bit we add (or subtract) the corresponding reference pattern r_j=($r_j(1)$, $r_j(2)$,..., $r_j(mn)$) (j=1,2,...,k), multiplied by embedding strength α_j. The resultant image is c_w=($c_w(1)$, $c_w(2)$,..., $c_w(mn)$). We obtain each image pixel with (sign '+' is for binary one, '-' for binary zero embedding):

$$c_w(i) = c_0(i) + \sum_{j=1}^{k} (\pm\alpha_j \cdot r_j(i)) \quad (i=1,2,...,mn) \tag{14}$$

[2] (Of course, if we neglect the fact that for one bit embeddind we get $\tau=3\cdot\sigma_{lc}$, and presume that we can take here smaller value of τ, as well).

Such embedding is localized in space. Only corresponding coordinates of the original image and reference pattern influence on resultant pixel value.

Reference patterns coordinates take values from distribution N(0,1). Therefore, their linear combination

$$R(i) = \sum_{j=1}^{k} (\pm\alpha_j \cdot r_j(i)) \quad (i=1,2,...,mn) \tag{15}$$

takes values from $N(0,\sigma^2)$, where

$$\sigma = \sqrt{\alpha_1^2 + \alpha_2^2 + ... + \alpha_k^2} \tag{16}$$

Also, coordinates of vector $r_s=(r_s(1), r_s(2),..., r_s(mn))$, where

$$r_s(i) = \frac{R(i)}{\sqrt{\alpha_1^2 + \alpha_2^2 + ... + \alpha_k^2}} \quad (i=1,2,...,mn) \tag{17}$$

are liable to N(0,1) distribution, and therefore, **r_s is reference pattern**.

Thus, **embedding of k reference patterns r_j one over another, with strengths** α_j (j=1,2,...,k), **is equal with embedding one reference pattern, r_s, with strength** $\beta = \sqrt{\alpha_1^2 + \alpha_2^2 + ... + \alpha_k^2}$, or

$$c_w = c_0 + \beta \cdot r_s \tag{18}$$

Mean square error of this embedding is

$$MSE(c_0,c_w)=\beta^2=\alpha_1^2+\alpha_2^2+...+\alpha_k^2 \tag{19}$$

If all patterns are embedded with the same strength, $\alpha_1=\alpha_2=...=\alpha_k=\alpha$, then the total embedding strength is $\beta = \sqrt{k} \cdot \alpha$.

Example 1: If k=2, $\alpha_1=3$, $\alpha_2=4$, then $r_s = \frac{3 \cdot r_1 + 4 \cdot r_2}{5}$. (Embedding of two patterns with strengths 3 and 4 makes the same MSE as embedding of one pattern, with strength $\beta = \sqrt{3^2 + 4^2} = 5$).

Example 2: If $\alpha=2$ and we embed the message of k=25 bits (one over another, with fixed strength embedding), then $\beta = \sqrt{k} \cdot \alpha = 10$, i.e. embedding of 25 patterns with strength 2 is equivalent to embedding of one pattern with strength 10.

3.2.3 Embedding into subimages

We divide image c_0 of mn pixels into k disjoint subimages $c_0^1, c_0^2, ..., c_0^k$, of $mn^1, mn^2,..., mn^k$ pixels, respectively ($mn = \sum_{j=1}^{k} mn^j$). For each subimage, the coefficient σ_{lc}^j (j=1,2,...,k) is

$$\sigma_{lc}^{j} = \frac{\sqrt{E(c_0^j)}}{mn^j} \tag{20}$$

If it is possible to divide this image into k parts of equal dimensions and energies, then all subimages will have the same σ_{lc}^{j} parameter value:

$$\sigma_{lc}^{j} = \frac{\sqrt{E(c_0^j)}}{mn^j} = \frac{\sqrt{E(c_0)/k}}{mn/k} = \sqrt{k} \cdot \frac{\sqrt{E(c_0)}}{mn} = \sqrt{k} \cdot \sigma_{lc} \tag{21}$$

In this case, the overall strength for k bits message, when embedding into k subimages, is equal to the strength for one bit message, multiplied by $\sqrt{k}$. Embedding strength in this case is equal as if the message is embedded one bit over another.

4. Robustness against lossy compression

Next we give an estimate of the strength coefficient α for watermark, to survive the lossy compression.

In subsection 4.1 we consider the case of one bit, and in subsection 4.2 – of a longer message. For the purpose of simplicity, we only discuss the case of binary one embedding. The contents of this section may be used with small changes for the case of binary zeros.[3]

4.1 Robustness against compression – one bit message

4.1.1 Embedding strength parameter after compression (α')

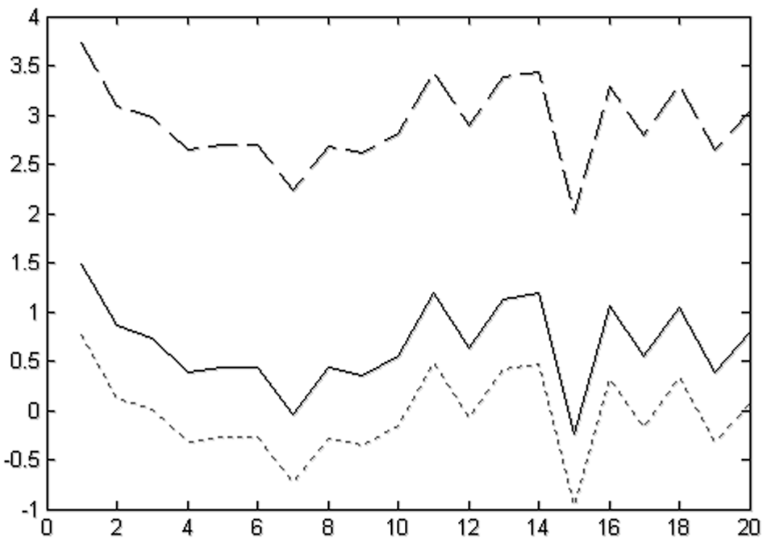
With next experiment we try to find which part of watermark will be destroyed by lossy compression.

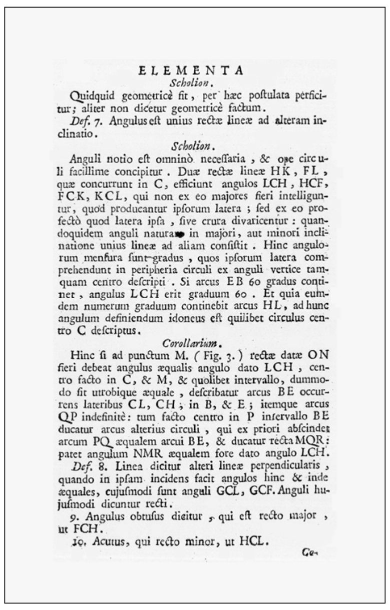
1. We embed k reference patterns r(1), r(2),..., r(k) (one by one) into the image c_0 with strength α (value α is the same for all of them). In a way, we get k watermarked images (with binary one embedded) $c_w(1), c_w(2),..., c_w(k)$.
2. Then, we subject each of these k images to expected lossy compression (same for all k images), and we get k compressed watermarked images $c_{wn}(1), c_{wn}(2),..., c_{wn}(k)$.
3. For each of them we calculate linear correlation with corresponding reference pattern (the one that is previously embedded in it).

In Fig. 1, linear correlation values of these images with reference patterns are shown (c_0 is the first page scan of the Ruđer Bošković's book '*Elementa geometriae*', α=3, k=20, compression is DjVu Photo, made by program DjVu Solo 3.1 – LizardTech, Inc). Graphic abscissa presents arrays indexes, and ordinate – their values.

The dotted line presents linear correlation values between the **original image** and reference patterns. The dashed line presents linear correlation values between **watermarked images** and corresponding reference patterns. The solid line presents the linear correlation values between **compressed watermarked images** and corresponding reference patterns.

[3] We can observe case with binary zero as the opposite pattern embedding. Opposite pattern of r_w is $-r_w$, and it also possesses all of reference pattern properties (takes values from distribution N(0,1)).





ELEMENTA

Scholion.

Quidquid geometricè fit, per hæc poſtulata perficitur; aliter non dicetur geometricè factum.

Def. 7. Angulus eſt unius rectæ lineæ ad alteram inclinatio.

Scholion.

Anguli notio eſt omninò neceſſaria, & ope circuli facillime concipitur. Duæ rectæ lineæ HK, FL, quæ concurrunt in C, efficiunt angulos LCH, HCF, FCK, KCL, qui non ex eo majores fieri intelliguntur, quod producantur ipſorum latera; ſed ex eo profectò quod latera ipſa, ſive crura divaricentur: quandoquidem anguli natura in majori, aut minori inclinatione unius lineæ ad aliam conſiſtit. Hinc angulorum menſura ſunt gradus, quos ipſorum latera comprehendunt in peripheria circuli ex anguli vertice tamquam centro deſcripti. Si arcus EB 60 gradus continet, angulus LCH erit graduum 60. Et quia eumdem numerum graduum continebit arcus HL, ad hunc angulum definiendum idoneus eſt quilibet circulus centro C deſcriptus.

Corollarium.

Hinc ſi ad punctum M. (Fig. 3.) rectæ datæ ON fieri debeat angulus æqualis angulo dato LCH, centro facto in C, & M, & quolibet intervallo, dummodo ſit utrobique æquale, deſcribatur arcus BE occurrens lateribus CL, CH; in B, & E; itemque arcus QP indefinitè: tum facto centro in P intervallo BE ducatur arcus alterius circuli, qui ex priori abſcindet arcum PQ æqualem arcui BE, & ducatur recta MQR: patet angulum NMR æqualem fore dato angulo LCH.

Def. 8. Linea dicitur alteri lineæ perpendicularis, quando in ipſam incidens facit angulos hinc & inde æquales, cujuſmodi ſunt anguli GCL, GCF. Anguli hujuſmodi dicuntur recti.

9. Angulus obtuſus dicitur, qui eſt recto major, ut FCH.

10. Acutus, qui recto minor, ut HCL.

Ge-

Fig. 1. Linear correlation between the image (original, watermarked and compressed watermarked) and corresponding reference patterns (left), for the first page of the book "*Elementa geometriae*" (right)

We can see from this graphic, that for each reference pattern r(i), (i=1,2,...,k) stands:

- Embedding of binary one with strength α raises linear correlation value between the image and the reference pattern by α
- Lossy compression reduces linear correlation by constant, i.e. it erases constant part of watermark. Thus, after lossy compression, it remains the constant part of watermark.

(Clearly, binary zero embedding reduces linear correlation value for α, and lossy compression raises it by constant – it erases constant part of watermark).

We introduce new concept: *embedding strength coefficient after compression*, α'. α' **is part of** α **that survives after lossy compression.**

For all of twenty reference patterns, the effect is the same: for our image and α=3, after DjVu Photo compression, it survived α'=0.75.

After repeating our experiment for other compression techniques (for various intensities, for JPEG compression and compression used in PDF and DjVu files making), we get the same conclusion:

For given image and compression intensity, α' is constant for fixed α coefficient (it doesn't depend on reference pattern embedded, nor on linear correlation value of original image with it).

Surely, for given α coefficient, value α' is smaller in more intensive compression. For example, α' is bigger for JPEG 70% compression than for JPEG 40%.

4.1.2 Coefficient α setting for a robust watermark

If we wish the watermark to be robust against lossy compression, we need to modify statements (8) and (9) (subsubsection 3.1.2: *Parameter α setting*) and replace each α with α':

- in the case of fixed embedding strength: $\alpha' = 3 \cdot \sigma_{lc} + \tau$
- in the case of embedding strength adjustment: $\alpha' = \max(\tau - I_0, 0)$ (for binary one), $\alpha' = \max(\tau + I_0, 0)$ (for binary zero embedding)

The procedure of deriving α' from α is "natural", because it matches to the events chronology: α and α' are "cause and effect":

1. We embed watermark with strength α;
2. We compress the image;
3. We read α'.

In order to derive α from α', we have to try with different α values, by using watermarks made from **arbitrary (but only one) reference pattern**.

Steps for α setting:

1. For our image, we set τ and (desired value of) α'
2. In our image we embed watermark for several α values; to get several watermarked images.
3. We expose these images to expected lossy compression.
4. The detector calculates α' for each compressed watermarked image. Then, we compare obtained α' values with desired value. α that corresponds α' value, closest to desired one, is appropriate.

4.2 Robustness of a longer message

If we embed k patterns into the image, each with own embedding strength coefficient α_i (i=1, 2... k), we have:

$$c_w = c_0 + \sqrt{\alpha_1^2 + \alpha_2^2 + \ldots + \alpha_k^2} \cdot r_s = c_0 + \beta \cdot r_s \tag{22}$$

$$r_s = \frac{\pm\alpha_1 r_1 \pm \alpha_2 r_2 \pm \ldots \pm \alpha_k r_k}{\beta} \tag{23}$$

After compression, we have:

$$c_{wn} \approx c_0 + \beta' \cdot r_s \,, \quad \beta' = \sqrt{(\alpha_1')^2 + (\alpha_2')^2 + \ldots + (\alpha_k')^2} \tag{24}$$

The overall strength coefficient remaining after compression is β' (the coefficient that corresponds to β for our image and compression technique).

$$p = \beta' / \beta \,, \quad \beta' = p \cdot \beta = \sqrt{(p \cdot \alpha_1)^2 + (p \cdot \alpha_2)^2 + \ldots + (p \cdot \alpha_k)^2} \tag{25}$$

Thus, $\frac{\alpha_1'}{\alpha_1} = \frac{\alpha_2'}{\alpha_2} = ... = \frac{\alpha_k'}{\alpha_k} = p$

The procedure for coefficients α_i (i=1,2,...,k) setting is:

1. For our image, we set parameters τ and α_i' (i=1,2,...,k);
2. We calculate $\beta' = \sqrt{(\alpha_1')^2 + (\alpha_2')^2 + ... + (\alpha_k')^2}$
3. We obtain β from β' and calculate $p = \frac{\beta'}{\beta}$
4. $\alpha_i = \frac{\alpha_i'}{p}$ (i=1,2,...,k)

Example 3: Let α_1'=3, α_2'=4. Then $\beta' = \sqrt{3^2 + 4^2} = 5$

If for given image and compression technique, to coefficient β'=5 corresponds β=10, then

$$p = \frac{\beta'}{\beta} = \frac{1}{2}, \qquad \alpha_1 = \frac{\alpha_1'}{p} = 6, \qquad \alpha_2 = \frac{\alpha_2'}{p} = 8$$

Example 4: We embed k=16 message bits (binary ones) into the image c_0='*Fishingboat*' (512×512 pixels).

$$\sigma_{lc} = \frac{\sqrt{E(c_0)}}{(512 \cdot 512)} = 0.25 \qquad \text{We take } \tau=0.25$$

α'= [0, 0.18, 0, 0.17, 0.41, 0.54, 0.19, 0.58, 0.42, 0.58, 0.64, 0.66, 0.14, 0.57, 0.31, 0.05]

(using embedding strength adjustment)

β'=1.65 β=5.4 p=0.3 $\alpha_i=\alpha'_i/p$ (i=1,2,...,k)

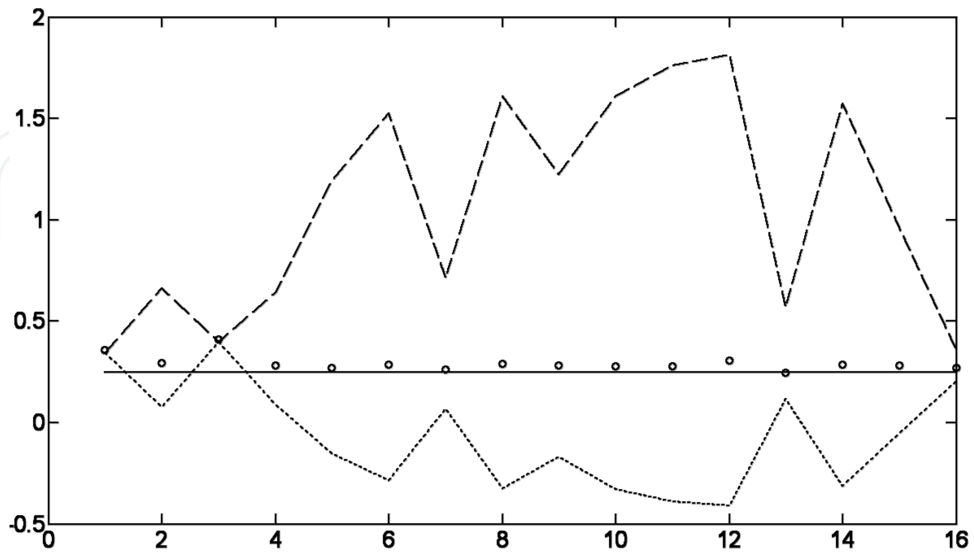


Fig. 2. Linear correlation between the image (original, watermarked and compressed watermarked) and k=16 embedded reference patterns

Fig. 2 shows linear correlation values between the image and each of embedded patterns. The dotted line presents linear correlation values for the original image; the dashed line is for the watermarked image. Roundels present linear correlation values between the compressed watermarked image and embedded reference patterns. We can see that their ordinates are near the threshold τ value (solid line). So, they are the smallest values needed for the detector to recognize after compression the image as watermarked.

Example 5: The same test is done for 64 bits message. The watermark also survives DjVu Photo compression. Overall embedding strength is β=6.94. In Fig. 3, the compressed original (without watermark) and compressed watermarked image could be seen.

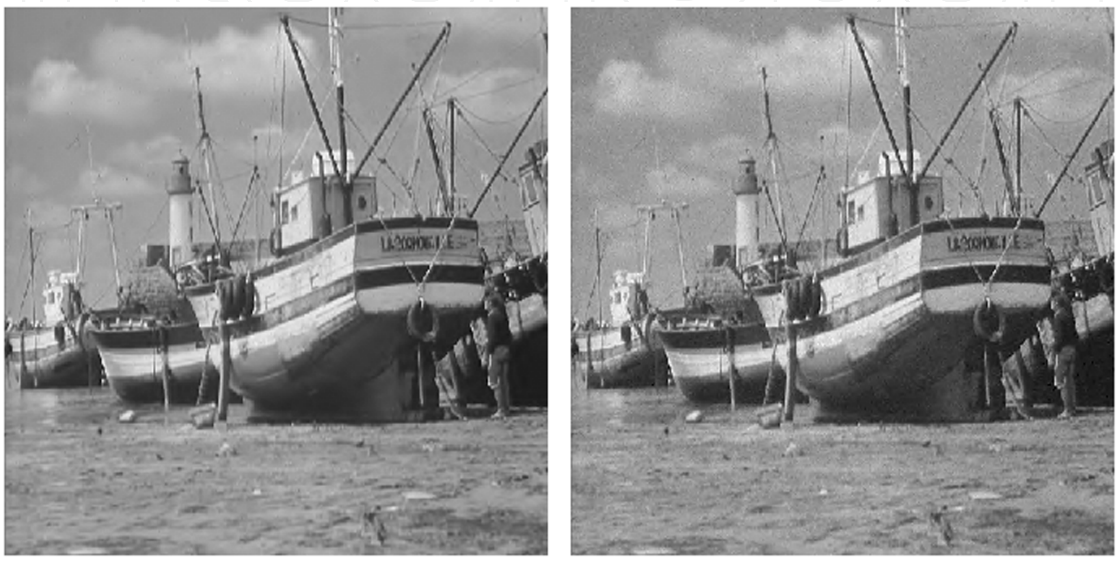


Fig. 3. Compressed original and compressed watermarked image ("*Fishingboat*", 512×512 pixels, k=64 message bits, DjVu Photo compression)

5. AWGN watermark in transform domain

Many authors suggest watermark embedding in transform, instead of spatial domain. In addition, they propose using transform domain that will be used during lossy compression. Goal is to achieve greater watermark robustness against expected compression.

5.1 Optimal embedding strength in transform domain

In this subsection we apply the described watermarking algorithm in transform domain (embedding and detection occur in transform, instead of spatial domain). We show that there is no difference from the viewpoint of effective embedding of AWGN watermark, between watermarking in spatial and in transform (DCT, block DCT, Fourier or arbitrary wavelet) domain.

First of all, most of nowadays used transforms in image processing and in compression are orthogonal (or at least unitary) linear transforms.

Linear transform f respects addition and scaling. Therefore,

$$f(c_0+\alpha\cdot r)=f(c_0)+\alpha\cdot f(r) \tag{26}$$

Orthogonal transform f preserves the inner product and consequently, it preserves vectors lengths and angles between vectors. Therefore,

$$\|c_0\| = \|f(c_0)\| \qquad \text{and} \qquad \|r\| = \|f(r)\| \tag{27}$$

$$\cos(c_0,r)=\cos(f(c_0),f(r)) \tag{28}$$

So, we have

$$lc(c_0,r)=lc(f(c_0),f(r)) \tag{29}$$

Also, for orthogonal (and unitary as well) linear transforms applies *Parseval's (energy) identity*:

$$\text{If} \quad b_0=f(c_0), \quad \text{then} \quad E(b_0) = \sum_{i=1}^{mn}(b_0(i))^2 = \sum_{i=1}^{mn}(c_0(i))^2 = E(c_0) \tag{30}$$

Orthogonal transform maps reference pattern $r=(r(1),r(2),...,r(mn))$ (i.e. vector with coordinates taken from distribution N(0,1)), into vector $f(r)$ from distribution N(0,1), i.e. **orthogonal transform maps reference pattern to reference pattern**.

Standard deviations from zero of linear correlations between vector c_0 and reference patterns in spatial and transform domain are equal:

$$\sigma_{lc}(c_0)=\sigma_{lc}(f(c_0)) \tag{31}$$

It is all the same if we embed AWGN watermark into an image with strength α in spatial or in transform domain. **Correlation values and MSE too, will be the same in both domains.**

Also, there is no difference between spatial and transform domains in AWGN watermark robustness against lossy compression.

5.2 AWGN watermark embedding into subimage in transform domain

As in spatial domain, it is possible to embed watermark in some **part of coefficients** in transform domain (here also, we call it *embedding into subimage*).

Technically, there is not any difference in optimal strength setting, speaking of subimage and of the whole image. Effective embedding strength we determine by dimension and energy of image (or image part) in that AWGN watermark is to be embedded. Robustness against lossy compression we determine by these coefficients properties in the regard to expected compression.

However, for images in spatial and in transform domain, there is a great difference in **energy distribution**. While in spatial domain, image energy is mainly distributed evenly over entire matrix, in transform domain it is concentrated in some of its elements. Illustration of this fact may be seen in Fig. 4, on example of block DCT (transform used in JPEG compression). Here, for presentation of matrices in DCT domain, we use solution proposed in (Vučković, 2008):

- matrix elements with value 0 are presented in black color
- positive elements are presented in nuances from black to white

- negative elements are colored in nuances from black to yellow
- because of big difference between image elements in block DCT domain (second and fourth objects in Fig. 4), they are presented with logarithmed magnitudes

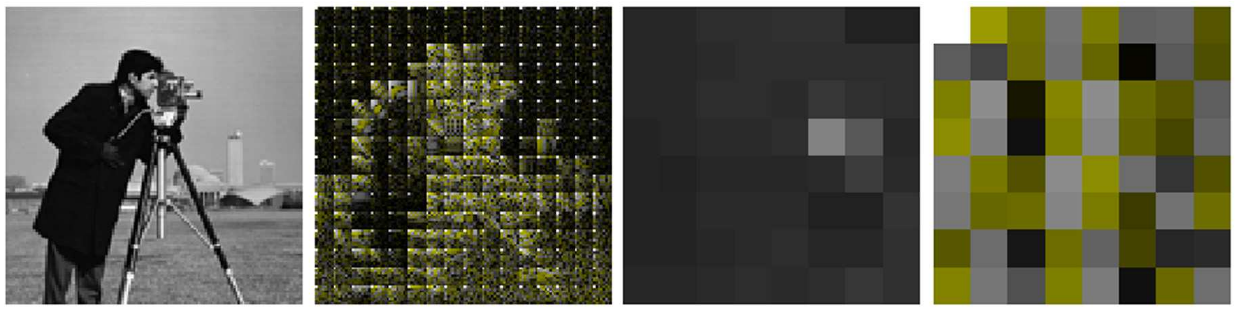


Fig. 4. Image '*Cameraman*' in spatial and in block DCT domain; one 8×8 block in spatial and in block DCT domain

As we can see, in block DCT domain, energy of each block is concentrated in its upper left corner. On the other hand, JPEG compression damages elements in lower right corner much more intensively.

Block DCT domain, which is the basis of JPEG compression, is used frequently in watermark embedding. At description of such embedding, we use term *image subchannel* (Eggers and Girod introduced it in (Eggers & Girod, 2001)). *Subchannel* is vector that in block DCT domain has for coordinates – elements, with same index in blocks. Subchannels are ordered according to zigzag order (Fig. 5). Thus, subchannel 1 consists of all DC blocks elements; subchannel 10 consists of all elements that are in the position 10 in the block (zigzag order).

So, the image of dimension m×n in (8×8) block DCT domain may be presented with 64 subchannels:[4]

$$s_j=(s_j(1),s_j(2),\ldots,s_j(nbl)),\ (j=1,2,\ldots,64)),\ \text{where } nbl=mn/64 \qquad (32)$$

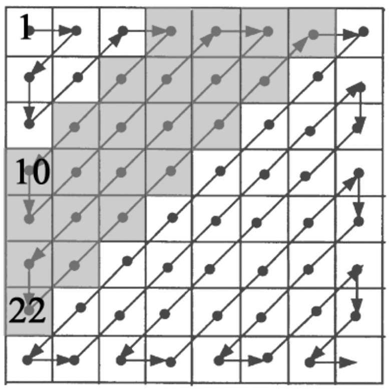


Fig. 5. Zigzag order; coefficients 1, 10, 22 in zigzag order; region of subchannels 10-22 (colored in gray)

[4] Here, for the purpose of simplicity, we assume that image dimensions are multiples of 8, in order to make the partition in 8×8 blocks possible.

In (Vučković, 2010a, 2010b) we described in detail our investigations results in connection with embedding into subchannels in block DCT domain. We analyzed case of embedding into group of (first 32) subchannels and embedding into some of individual subchannels (specifically, subchannels 1, 10 and 22). Here, we briefly outline conclusions obtained.

In block DCT domain, image energy is concentrated in upper left corner of the block. For example, in image '*Cameraman*', even 99.76% of whole image energy is concentrated in its first 32 subchannels. Therefore, with described algorithm, it is needed twice higher strength for effective embedding in first 32 subchannels than into the whole image. In this way we embed in 50% image coefficients, so the mean square error in this case is two times higher comparing with embedding into all of coefficients.

Embedding in lower right block corner is effective even with very small strength. However, even at very low compression, these coefficients will be destroyed. Therefore, embedding into those coefficients is not recommended.

Embedding in vicinity of subchannel 1 is not good solution, because of extremely high energy which requires big embedding strength (and this badly influences image quality).

Our explorations (Vučković, 2010a, 2010b) confirmed earlier (Eggers-Girod, 2001), that for effective and robust (against JPEG compression) and imperceptible watermark, it is the best to embed it in several coefficients in upper left block part (somewhere in the region of subchannels 10-22 – see Fig. 5). Such embedding showed also good robustness against compression techniques different from JPEG (for example, against DjVu compression).

6. AWGN watermark and other image modifications

Image modifications are organized (Cox et al., 2008) in two classes - valumetric and geometric distortions.

6.1 Valumetric distortions

Valumetric distortions are simpler than geometric ones. They change individual pixels values. These modifications include additive noise, amplitude changes, linear filtering and lossy compression.

6.1.1 Additive noise

This modification has effect of a random signal adding. For watermarked image, additive noise adding is defined by:

$$c_{w1} = c_w + s = c_0 + \alpha \cdot r + s, \tag{33}$$

where s is a random vector chosen from some distribution, independently of c_0 and r.

Clearly,

$$lc(c_{w1},r) = lc(c_w,r) + lc(s,r) \approx lc(c_w,r) \tag{34}$$

(because s is uncorrelated with r).

Thus, an additive noise does not affect AWGN watermark (AWGN algorithm is robust against this image modification).

AWGN watermark is also robust against **brightness changing** ($c_{w1}=c_w+n\cdot J$, where J is matrix of ones and n is integer).

6.1.2 Amplitude changing

It may be presented by the formula

$$c_{w1}=\nu\cdot c_w, \quad (35)$$

where $\nu>0$ is scaling factor. Such operation causes **brightness and contrast changing**.

After this modification, linear correlation value is equal to starting one, multiplied by factor ν. Depending on factor ν, watermark detectability will increase (if $\nu>1$), or diminish (if $\nu<1$).

6.1.3 Linear filtering

It is given by

$$c_{w1}=c_w*f, \quad (36)$$

where f is a filter, and $*$ designates convolution. Many common image operations are performed using linear filters. Examples of them are **blurring** and **sharpening** effects.

Linear filtering and lossy compression are more complex operations than the previous two, because changes that they cause are not strictly localized (pixel change depends on certain number of surrounding pixels too).

6.2 Geometric distortions

This class of image modifications includes many image distortions (rotation, spatial scaling, translation, skew or shear, cropping, perspective transformation, and changes in aspect ratio).

These modifications are more complicated than valumetric, because they displace information about pixels in image matrix. They usually change matrix dimensions as well. Therefore, here is not possible readily to detect the watermark. Yet, with these modifications, information about embedded message is not lost, but only "**masked**". For each geometric distortion, before detection, we need to perform one **correction procedure**.

For illustration, we depict one geometric modification – image rotation by an angle φ. Many inferences of this analyze could be applied to other modifications.

With modifications that erase a part of image data (for example crop operation, or other operation with erasing several columns or rows), after correction procedure, we'll have original image matrix with locally erased data (some elements are erased, but other are untouched). Then, it is possible to calculate accurately parameter σ_{lc} for untouched image part. For example, if prior to watermark embedding we know that image will be subjected to crop operation with reducing complete image area to its quarter, the watermark needs to be embedded twice stronger, to be detectable after cropping.

6.2.1 Robustness against rotation

In the case of image rotation, one of several different correction procedures may be used prior to detection. If watermarked image is rotated by angle φ, to make watermark detection possible, we may do one of the following:

- Compare (using linear correlation) rotated image with the reference pattern that is also rotated by φ
- Before detection, rotate the rotated image by angle -φ and crop it to its original dimension; and then, compare it with reference pattern, that is also rotated by angles φ and -φ and cropped to original dimension
- Image rotated by φ and then by -φ (and then cropped to original image dimensions), (we may) compare with the original reference pattern.

Linear correlation values for image and reference pattern (unrotated and rotated) are presented in Fig. 6, for image "*Cameraman*". Reference pattern r_w is embedded with strength α=5. Then, image is subjected to rotation by angle $\varphi=10^\circ$.

In each row we present image and pattern for which the correlation is calculated. So, for each example it is specified which image and pattern (and their dimensions) we deal with, and also the linear correlation value for them. In the first figure row, watermarked image and embedded reference pattern are presented. Second, third and fourth row contain information about three previously stated solutions of correction and detection.

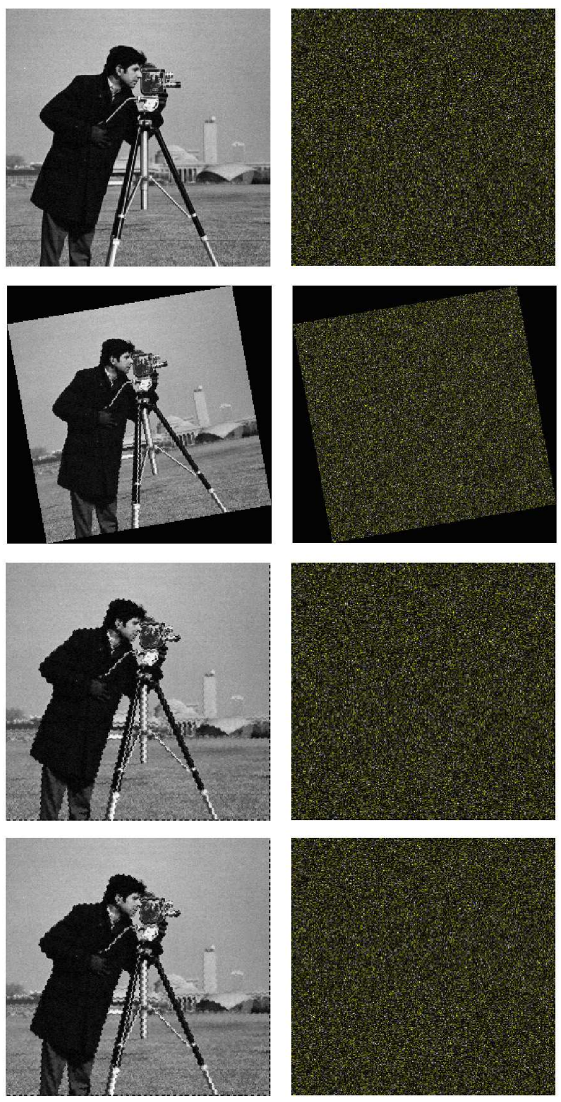
The rotation practically does not reduce linear correlation value, if it is calculated for image and reference pattern that are rotated in the same manner. However, if we compare the rotated image (after restoration to original position by rotation in opposite direction and cropping to original dimension) with the original reference pattern, certain watermark amount will be lost (α′ value will be considerably less than embedding strength value α).

It should be noted that in cases where not all details of distortion occurrence are known, the latest solution is usually only possible.

Experiments have confirmed (Vučković, 2010a) that stated inference for lossy compression (subsubsection 4.1.1: *Embedding strength parameter after compression (α′))* stays also for rotation (and other distortions too). For our image and embedding strength α, after given modification, remaining strength α′ **doesn't depend on the reference pattern nor on its correlation with the original image**. Hence, to set embedding strength α needed that after expected modification remains α′**, it is sufficient to experiment with only one reference pattern**.

The procedure of determining the necessary embedding strength α may be performed in the next steps:

1. We determine α' using procedure given in subsubsection 4.1.2 (*Coefficient α setting for a robust watermark*)
2. By experimenting with only one reference pattern, we determine the necessary embedding strength α, that after expected modification, remaining strength is at least α′.



c_w(256,256) watermarked image ('*Cameraman*')

r_w(256,256) – embedded reference pattern

$lc(c_w,r_w)$= 5.787

c_{wr}(297,297) – watermarked image, rotated by φ (=10°)

r_{wr}(297,297) – reference pattern, rotated by φ

$lc(c_{wr},r_{wr})$= 5.815

c_{wr1} (256,256) – watermarked image, rotated by φ and -φ, after cropping to original dimensions

r_{wr1} (256,256) – reference pattern, rotated by φ and -φ, after cropping to original dimensions

$lc(c_{wr1},r_{wr1})$= 5.592

c_{wr1} (256,256) – watermarked image, rotated by φ and -φ, after cropping to original dimensions

r_w(256,256) – (embedded) reference pattern

$lc(c_{wr1},r_w)$= 1.953

Fig. 6. Linear correlation values for image and reference pattern (unrotated and rotated)

7. Color images and e-books

7.1 Digital watermark and color image

Each color image can be regarded as three grayscale images combination (one for each of red/ green/ blue components). Therefore, it is not too complicated to generate results for grayscale images to use for color images. Of course, for better results we need to consider human visual system (HVS) characteristics:

- Human eye notices changes in green color nuances best, a bit worse it discerns changes in red, and changes in blue it discerns worst. That's why some authors recommend watermark embedding mostly in blue color channel (because human eye is the least sensitive there).
- Human eye better notices changes in brightness than in color nuances. Hence, JPEG standard uses different compression intensities for those components. For good watermark algorithm, this fact should be considered.

7.2 Watermark for e-books

Electronic book (e-book) is the digital equivalent of a printed book. In our *Virtual library* (Mijajlović & Vasiljević, 2008), e-books are in PDF format. On Internet, DjVu format is used often as well.

PDF books in Virtual library, from the viewpoint of digital watermark embedding, may be classified as:

- E-books produced by **scanning**; their pages are images (book pages scans). Watermark may be embedded into them in the same manner as in other images.
- E-books originated from **DOC or TeX files**; their image pages are not bitmapped images. Hence before embedding, they need to be converted to bitmaps.

7.2.1 PDF e-books produced by scanning

Embedding procedure for PDF e-books:

1. We scan book pages – results are bitmapped (e.g., TIFF) images.
2. In each book page (TIFF file) we embed watermark – result is watermarked (TIFF) page.
3. We combine watermarked (TIFF) pages into one PDF file – watermarked e-book.

Detection:

1. Book page in which we search for watermark we extract from PDF file and save as (e.g.) TIFF image.
2. We detect the watermark message in this image.

7.2.2 PDF e-books originated from DOC and TeX files

These PDF pages need at first to be converted to bitmaps; this can be done by conversion of PDF pages into (e.g.) TIFF images. Then we may proceed with just described procedure (7.2.1).

However, by converting pages into images, we lose possibility of text retrieving in the file. This problem can be overcome by using some OCR procedure.

7.2.3 Problem with black-white books

Proposed algorithm is not suitable for simple black-white images (as book pages usually are). In big white image regions (for example, page margins) a large part of embedded data will be destroyed by "squeezing" operation: values that, after adding of white noise to image matrix become greater than 255, after "squeezing" return to 255. Also, it is very easy to remove watermark data from these image regions (for example, with eraser in Photoshop).

Pages in e-books produced by scanning usually aren't black-white, but grayscale (or color) images; even their margins are not completely white, and watermarking is possible for them. Problem is emphasized with second mentioned e-books class. Watermarking makes sense only if these books aren't purely textual files but they contain some amount of pictures, in which watermark could be embedded.

With black-white text pages there exists another problem, which makes them especially inconvenient for watermarking: watermark can be removed simply by text retyping from them.

8. Conclusion

This text contains integrated results of several earlier papers.

For given grayscale image and AWGN watermark, we described procedure for optimal embedding strength setting. We analyzed optimal strength for effective embedding, and also watermark robust against expected image modification.

We analyzed AWGN watermark embedding cases, into the whole image and into images, in spatial and transform domains and robustness of such embedding against expected compression.

For other image modifications, procedures for optimal strength setting are similar as in the case of compression. For each geometric modification, however, we need prior detection to perform one correction procedure.

Obtained results we applied to color images and e-books. AWGN algorithm is not applicable on black-white e-books that originate from DOC or TeX files, because watermark may be removed from them by simply retyping of text.

9. Acknowledgment

The work presented here was supported by the Serbian Ministry of Education and Science (project III44006).

10. References

Cox, I.J., Kilian, J., Leighton, F.T. & Shamoon, T. (1997): Secure spread spectrum watermarking for multimedia, *IEEE Transactions on Image Processing,* 6(12), pp. 1673-1687

Cox, I . & Miller, M., Bloom, J., Fridrich, J. & Kalker, T. (2008): *Digital Watermarking and Steganography*, Morgan Kaufmann Publishers, ISBN 978-0-12-372585-1

Eggers, J. & Girod, B. (2001): Quantization effects on digital watermarks, *Signal Processing* 81 (2001) pp. 239–263

Feng, G., Jiang, L., Wang, D. & He, C. (2005): Quickly tracing detection for spread spectrum watermark based on effect estimation of the affine transform, *Pattern Recognition,* Volume 38 Issue 12, December, 2005, pp. 2530 – 2536

Feng, G., Jiang, L., He, C. & Xue, Y. (2006): Chaotic spread spectrum watermark of optimal space-filling curves, *Chaos, Solitons and Fractals* 27, pp. 580–587

Mijajlović, Ž. & Vasiljević, D. (2008): Web page *Virtual library eLibrary*, Available from http://elibrary.matf.bg.ac.rs/

Mora-Jimenez, I. & Navia-Vazquez, A. (2000); A new spread spectrum watermarking method with self-synchronization capabilities,, *Proceedings 2000 International Conference on Image Processing*, pp. 415 - 418 vol. 1 , Print ISBN: 0-7803-6297-7

Perez-Freire, L. & Perez-Gonzalez, F. (2009): Spread-Spectrum Watermarking Security, *IEEE Transactions on Information Forensics and Security,* March 2009, Vol. 4, Issue: 1, pp. 2 – 24, ISSN: 1556-6013

Ruanaidh, J.J.K.O. & Pun, T. (1998): Rotation, scale and translation invariant spread spectrum digital image watermarking, *Signal Processing* 66 No. 3, May 1998, pp. 303-317, ISBN: 0818681837

Ruanaidh, J.J.K.O. & Csurka, G. (1999): A Bayesian approach to spread spectrum watermark detection and secure copyright protection for digital image libraries, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)* - Volume 1, 1999, pp. 1207-1212, Print ISBN: 0-7695-0149-4

Vučković, V. (2008): Image and its matrix, matrix and its image, *Review of the National Center for Digitization*, Issue: 12, pp. 17-31, ISSN: 1820-0109, http://www.ncd.matf.bg.ac.yu/casopis/12/NCD12017.pdf

Vučković, V. (2010a): Embedding strength criteria for AWGN watermark, robust against expected distortion*, Computing and Informatics*, Vol. 29, no. 3 (2010), pp. 357–387, ISSN 1335-9150

Vučković, V. (2010b): *AWGN watermark optimal strength (Optimalna snaga žiga belog Gausovog šuma)* – doctoral dissertation, Faculty of Mathematics, Belgrade, 2010, http://elibrary.matf.bg.ac.rs/bitstream/handle/123456789/1073/PhDVVuckovic.pdf

Vučković, V. (2011): Digital watermark in digital images and e-books, *Review of the National Center for Digitization* 19, pp. 1–6, ISSN: 1820-0109, http://elib.mi.sanu.ac.rs/files/journals/ncd/19/ncd19001.pdf