Twin Peaks: The Histogram Attack to Fixed Depth Image Watermarks

Maurice Maes

Philips Research Laboratories
Prof. Holstlaan 4, 5656 AA Eindhoven, The Netherlands

Abstract. In this paper we present an attack to fixed depth image watermarks. The attack is based on histogram analysis of a watermarked image. With this attack, the watermark can often be reconstructed from just a few images, without using the detector.

1 Introduction

Digital watermarking is a research area which aims at hiding secret information in digital multimedia content including images, audio, or video. It is mostly used to embed copyright information in such a way that it is imperceptible by humans, but easily detected by a computer. Watermarks should be difficult to remove, and robust to for instance image compression, noise, scaling, etc.

Many presently used techniques for watermarking of digital images boil down to adding a pseudo-noise pattern to the images. The detection of the watermark is then done by correlating the possibly watermarked image with the pseudo-noise sequence, which is generated by a secret key.

This paper deals with an attack to these watermark methods. In particular, it will be shown that if the added noise sequence does not depend on the original image, that is if it is a so-called 'fixed depth' watermark, then it is easily hacked by a malicious attacker. Fixed depth watermarks are for instance proposed in Bender et al. (1996) and Pitas et al. (1995-1996). The concepts of our attacks also apply to image dependent, 'variable depth' watermarks.

The contents of this paper is as follows. In Section 2, we present a convenient framework in which we define images and operations on images. Watermarks are being introduced in Section 2.1. We discuss the basic requirements of watermarks, and the embedding and detection details of so-called 'fixed depth' watermarks. Section 3 discusses a method to remove the watermark from an image when only an (unknown) part of the watermark is estimated correctly. Section 4 then describes our method to obtain a good estimate from an analysis of the histogram of a watermarked image.

2 Definitions

In this paper, images are rectangular arrays of pixels. Pixels are seen as points, rather than squares, so they can be seen as a subset of $\mathbb{Z}^2$. A digital image is

obtained by assigning colors or gray values to pixels. Usually these colors or gray values are represented by bounded integers. For watermark patterns, negative values are also allowed, and even though in this case the 'image' has no meaning in a visual context, such a pattern will also be called an image.

Let m and n be positive integers, representing the *width* and *height* of an image. The total number of pixels will be denoted by N, so $N = mn$. Consider the lattice $P \subset \mathbb{Z}^2$, defined by

$$P := \{p_{ij} = (i, j) \in \mathbb{Z}^2 \mid 1 \leq i \leq m, 1 \leq j \leq n\}. \tag{1}$$

The elements p_{ij} of P will be called *pixels*.

Let k be a positive integer, representing the dimension of the color space. Usually, $k = 1$ (for gray value images) or $k = 3$ (for color images like YUV or RGB). An image I is defined as a mapping

$$I : P \to \mathbb{R}^k. \tag{2}$$

We will write $I_{ij} = I(p_{ij})$, and I_{ij} represents the *color* of pixel p_{ij}.

Linear combinations and inproducts of two images are defined in a straightforward way.

2.1 Watermarks

A *watermark method* consists of

1. a *watermark embedding* algorithm, which assigns to each image I an image I_W, and
2. a *watermark detection* algorithm, which assigns to each image J a probability p that J is of the form I_W for some image I.

If an image J is of the form I_W for some image I, with probability p, we will say that in J, *the watermark is present* with probability p. The detection algorithm often simply returns 1 or 0, meaning whether the watermark is present or not. For the watermark embedding algorithm, an important requirement is that the watermark is invisible. For the detection algorithm, important requirements include the following. The detection should not need the original image: to decide whether J is of the form I_W, no a priori knowledge about possible candidates for I should be required. Furthermore, probabilities of false alarms and missed detections should be small, while the method should be robust with respect to several sorts of operations on images that in a specific application are likely to occur. Examples of such operations are data compression, noise, format conversions, geometric transformations or dedicated transformations applied by attackers.

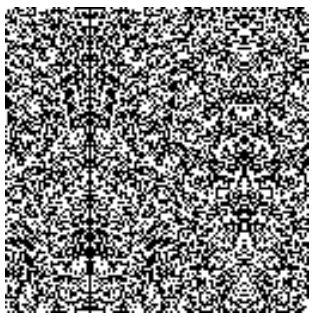

Fig. 1. An example of a watermark pattern

2.2 Fixed depth watermarks

The embedding method. A commonly used watermark embedding technique is that of adding a pseudo-noise pattern to a digital image. This pattern may or may not depend on the original image. A *fixed depth watermark* embedding method is a mapping which assigns to each image I an image I_W of the form

$$I_W = I + W, \tag{3}$$

where the *watermark pattern* W does not depend on I. The image W is said to be *of uniform depth* d if $|W_{ij}| = d$ for all i, j. The image W is called is called *DC-free* if $\sum_{i,j} W_{ij} = 0$.

For gray scale images, the watermark patterns of uniform depth d that we consider are of the form

$$W_{\text{gray}}(p_{ij}) = \pm d \tag{4}$$

for all i, j. The '+' or '−' sign is determined according to some 2-dimensional pseudo-random noise pattern. An example of such a pattern is shown in Figure 1. In this figure of size $m = n = 128$, white blocks correspond to pixels p_{ij} for which $W_{\text{gray}}(p_{ij}) = d$, and black blocks correspond to pixels with $W_{\text{gray}}(p_{ij}) = -d$. To actually watermark images that are not exactly 128 by 128 pixels, the basic pattern is tiled so as to obtain the correct dimensions.

For YUV images, the watermark pattern W_{yuv} is commonly defined such that it effects the luminance value y only:

$$W_{\text{yuv}}(p_{ij}) = (\pm d, 0, 0). \tag{5}$$

When the then obtained watermarked image is converted to an RGB image by the well-known rule

$$(r, g, b) = (y + v, y - \frac{1}{6}u - \frac{1}{2}v, y + u), \tag{6}$$

then the watermarking effect is equivalent to adding the pattern

$$W_{\text{rgb}}(p_{ij}) = \pm(d, d, d) \tag{7}$$

directly to the RGB image.

The detection method. The detection method in its basic form consists of taking the inner product of the image J with the pattern W, and this inner product is compared with some threshold value to decide whether J is watermarked or not. Consider the gray scale watermark embedding method described above. Let J be a gray scale image. Then we define the *correlation* $y_W(J)$ to be

$$y_W(J) = W.J. \tag{8}$$

This value $y_W(J)$ is then compared with a threshold value $y_{\text{thr}}(J)$, which depends on J, to decide whether J is watermarked or not. To see why this works, note that an 'arbitrary' image I is not correlated with W, so $W.I$ is small. Now if I is watermarked and we consider $J = I_W$, then the inner product becomes

$$y_W(J) = W.J = W.(I + W)) = W.I + W.W = W.I + d^2N.$$

Now d^2N will generally be large compared to $W.I$, and this enables detection of the watermark W. Of course, statistical analysis supports detection decisions.

3 Misleading the Detector

The purpose of a malicious attacker may be to obtain *full knowledge* of the watermark method, or just to *remove* the watermark from an image by *misleading the detector.*

When discussing vulnerability to malicious attackers, one should always have in mind the specific circumstances of an application. The attacker may have some knowledge of the watermark method. For instance, he may know or suspect that it is a fixed depth method. The attacker may or may not have the disposal of a detector. If he does, we assume the detector is a 'black box' to the attacker: it just says 'yes' or 'no', and no quantitative info is given. The attacker may have one or many watermarked images (with the same watermark) at his disposal.

In Section 4, we will see that incautious application of fixed depth watermarks in some cases can enable an attacker to obtain complete knowledge of the method, using just a few watermarked images! For this he doesn't even need the detector.

However, when he is not that successful, he can still exploit a partial guess by misleading the detector: he can 'remove' the watermark by subtracting an estimated watermark from the watermarked image. To illustrate this, let us assume that detection is done as described in Section 2.2. So we have

$$y_W(J) = W.I + W.W,$$

where, in general, we will have $W.I \ll y_{\text{thr}}(J)$ and $W.W > y_{\text{thr}}(J)$. Now suppose that an attacker has in some way estimated the watermark to be an image $\tilde{W}$. Let $\rho > 0$ be defined by

$$\rho = \frac{W.\tilde{W}}{W.W}. \tag{9}$$

In that case we will say that the attacker has *estimated* $\rho \cdot 100\%$ of the watermark. Now the watermark can be 'removed' from J, by subtracting $\rho^{-1}\tilde{W}$ from J. Let

$$\tilde{J} = J - \rho^{-1}\tilde{W}. \tag{10}$$

Then we have

$$\begin{aligned} y_W(\tilde{J}) &= y_W(J - \rho^{-1}\tilde{W}) \\ &= W.(I + W - \rho^{-1}\tilde{W}) \\ &= W.I + W.W - \rho^{-1}W.\tilde{W} \\ &= W.I, \end{aligned}$$

and this will generally be less than the threshold, so the watermark will not be detected in $\tilde{J}$. Let us consider the example of a uniform depth watermark.

Example 1. Suppose we have a watermark W of uniform depth $d = 1$. So $W.W = N$. Now suppose an attacker has estimated the sign of the watermark correctly for 75% of the pixels, while he has estimated it falsely for 25% of the pixels. Let $\tilde{W}$ be the estimated watermark, where $\tilde{W}_{ij} = \pm 1$ for all i and j. Then we have

$$W.\tilde{W} = \frac{3}{4}N - \frac{1}{4}N = \frac{1}{2}N,$$

so $\rho = \frac{1}{2}$, and we say that the attacker has estimated 50% of the watermark. From the above discussion, it follows that from a watermarked image J, the watermark can be removed by subtracting $2\tilde{W}$. Then, in pixels p_{ij} where the sign was estimated falsely, we have $|\tilde{J}_{ij} - I_{ij}| = 3$.

More generally, if $p\%$ of the pixels is estimated correctly, with $p > 50$, then

$$\rho = \frac{p - 50}{50},$$

so in a falsely estimated pixel, we have a maximum possible difference

$$|\tilde{J}_{ij} - I_{ij}| = 1 + \lceil \frac{50}{p - 50} \rceil.$$

Here $\lceil x \rceil$ denotes the smallest integer larger than or equal to x. Note that if ρ^{-1} is not an integer, then we cannot simply subtract $\rho^{-1}\tilde{W}$, because projection onto the color map then includes rounding for many pixels, and that can not be ignored. To circumvent this, we apply a dithering technique. Write $\rho^{-1} = k + r$ for $k \in \mathbb{N}$ and $r \in [0, 1)$. Then we 'randomly' divide the pixels into two disjoint sets P_1 and P_2 of approximate sizes $(1 - r)N$ and rN, respectively. Then we let

$$\tilde{J}_{ij} = \begin{cases} J_{ij} - k\tilde{W}_{ij} \text{ if } p_{ij} \in P_1, \\ J_{ij} - (k+1)\tilde{W}_{ij} \text{ if } p_{ij} \in P_2. \end{cases}$$

Note that $k(1 - r) + (k + 1)r = k + r = \rho^{-1}$, so the average difference between pixels values in $\tilde{J}$ and J is ρ^{-1}.

The above example shows that the quality of the image $\tilde{J}$ depends on how well the watermark is estimated. It also illustrates how to subtract $\rho^{-1}\tilde{W}$ in the case that ρ^{-1} is not an integer. The remaining question is how to determine ρ. An attacker can find out how good his estimate is, by replacing given percentages of the estimated watermark by random noise, and checking if the watermark is still detected. The same can be done for a watermarked image, and the noise percentages needed to remove the watermark in both the estimated watermark and the watermarked image can be compared.

4 Twin Peaks; The Histogram Attack

The histogram attack estimates a watermark by using only the histogram of an image. We first present some simple examples that illustrate this attack. We then introduce an operation called 'flattening' which may be applied to images to enhance the histogram attack. Next we present some experimental results, and then we discuss the method in some more detail. We conclude with a scenario how an attacker can exploit multiple images with the same watermark to come to a better estimate.

4.1 Examples

For the moment, let us assume we have gray scale images, and a uniform depth DC-free watermark of depth 1. Because the watermark pattern is DC-free, we know that 50% of the pixels will have their color decreased by 1, and 50% of the pixels will have their color increased by 1 (we ignore clipping at the values 0 and 255). Now suppose we have an original image in which an isolated color occurs. The histogram may for instance look like:

10	11	12	13	14	15	16
0	0	0	1200	0	0	0

So we have 1200 pixels with gray value 13 and neighboring colors do not occur. When this image I is watermarked, the histogram of J may look like:

10	11	12	13	14	15	16
0	0	573	00	627	0	0

For an attacker who suspects that the watermark is of depth 1 and DC-free, it is easily concluded that the 573 pixels having gray value 12, should all be pixels in which the watermark is -1. (If the watermark would have been $+1$, then the original gray value would have been 11, but in that case one would also expect pixels with gray value $11 - 1 = 10$ in the histogram of J). Similarly, the pixels with gray value 14 will all correspond to $+1$ watermark pixels with very high probability.

Isolated colors occur in images for instance when bits are suppressed or when the colors of an RGB are mapped to a limited smaller set of colors. The histogram

attack is not limited to isolated colors however, but can also be applied to histograms with a 'peaky' look, or to histograms in which many short non-zero sequences, mutually separated by 3 or more zeros, occur. This will be illustrated in the following example, in which we examine part of the histogram of a watermarked image J, from which we reconstruct the histogram of the original image I. After we have done that, probabilities can be given for groups of pixels, depending on their gray value, whether they correspond to $+1$ or -1 watermark pixels. (In the remainder we will say that 'a pixel is a $+1$ or -1 pixel'.)

Example 2. The table below shows part of the histogram of the watermarked image J, and we will try to reconstruct the histogram of the original image I. We write I^k for the number of pixels in i with gray value k, and likewise for J.

	86	87	88	89	90	91	92	93	94	95	96	97	98
I	?	?	?	?	?	?	?	?	?	?	?	?	?
J	0	0	1320	1510	3756	2187	6824	4073	6234	3364	2074	0	0

Now note that the 1320 pixels with gray value 88 can only be -1 pixels, otherwise in I they would have had value 87, but then in the histogram of J, there would have been pixels with value 86 (all this, as well as the rest of the arguments in this example, is based on probabilities). So in I, there should have been a number of pixels with gray value 89, of which about 50% (the 1320 pixels) were -1 pixels. Also, about 50% would have been $+1$ pixels, so the total estimate of I^{89} is $2 \cdot 1320 = 2640$. A similar argument can be applied to the tail of the non-zero sequence, and we then obtain the following preliminary estimate for the histogram of I, where from the histogram of J, the $2640 + 4148$ pixels that have been reconstructed, are removed (so e.g. $J^{90} = 3756 - 1320 = 2436$). We obtain:

	86	87	88	89	90	91	92	93	94	95	96	97	98
I	0	0	0	2640	?	?	?	?	?	4148	0	0	0
J	0	0	0	1510	2436	2187	6824	4073	4160	3364	0	0	0

where the added 0's in the estimate for I are obvious. We can now repeat the above step to the extreme values in the remaining non-zero sequence in the histogram of J, and obtain

	86	87	88	89	90	91	92	93	94	95	96	97	98
I	0	0	0	2640	3020	?	?	?	6728	4148	0	0	0
J	0	0	0	0	2436	677	6824	709	4160	0	0	0	0

and next

	86	87	88	89	90	91	92	93	94	95	96	97	98
I	0	0	0	2640	3020	4872	?	8320	6728	4148	0	0	0
J	0	0	0	0	0	677	228	709	0	0	0	0	0

Finally, we add $677 + 677$ to obtain the estimate for I^{92}, and we get

	86	87	88	89	90	91	92	93	94	95	96	97	98
I	0	0	0	2640	3020	4872	1354	8320	6728	4148	0	0	0
J	0	0	0	0	0	0	228	32	0	0	0	0	0

We do not bother about the remaining $228 + 32 = 260$ pixels which are less than 1% of all the pixels. It should be noted that the above method of reconstructing the histogram of I is certainly not the theoretically optimal way to reconstruct it. We will return to the reconstruction method later. Now, given the estimated histogram of I, we can compute for groups of pixels in J the probabilities that they are -1 or $+1$ pixels. This is illustrated in the following table:

	86	87	88	89	90	91	92	93	94	95	96	97	98
I est	0	0	0	2640	3020	4872	1354	8320	6728	4148	0	0	0
$p(-1)$	0	0	1	1	0.65	0.31	0.63	0.83	0.33	0	0	0	1
$p(+1)$	1	0	0	0	0.35	0.69	0.37	0.17	0.67	1	1	0	0

To explain these probabilities, note that, given the above estimate for (the histogram of) I, a pixel in J with gray value 88 can only be a -1 pixel, since in the estimate for I, there are no pixels with gray value 87. A pixel in J with gray value 91, can originate from the 3020 pixels in I with gray value 90 if it is a $+1$ pixel, or it can originate from the 1354 pixels with gray value 92 if it is a -1 pixel. The probabilities of either of these being the case are

$$\frac{3020}{3020+1354} = 0.69 \quad \text{and} \quad \frac{1354}{3020+1354} = 0.31,$$

respectively. All other probabilities are computed similarly.

For gray scale images, the eventual estimate for the watermark can be based on the above probabilities. The best we can do is guess that for instance *all* pixels in J with gray value 91 are $+1$ pixels, because $p(+1) > p(-1)$ for these pixels. Note that the exact probabilities tell us how good our estimate is! Indeed, 31% of the 2187 pixels with gray value 91 in J will be estimated falsely!

For RGB images, a similar analysis as above can be applied to all color components, so for each pixel, the eventual estimate can be based on 3 computed probabilities.

4.2 Flattening

In this section we will introduce 'flattened' images. For the histogram attack it is often advantageous to first filter the image J, e.g. with the filter

$$F = \begin{bmatrix} -1/9 & -1/9 & -1/9 \\ -1/9 & 8/9 & -1/9 \\ -1/9 & -1/9 & -1/9 \end{bmatrix}.$$

For more detailed information on filtering we refer to e.g. Van Den Enden and Verhoeckx (1989). The filtered image $\bar{J} = FJ$ will be called the *flattened* image. Figure 2 illustrates the flattened image of lenna, where the gray value 127 is added to FJ to visualize the result. The flattening filter is chosen such that if

Fig. 2. The flattened image of lenna.

J contains a watermark, then $\bar{J}$ also contains the watermark 'just as hard', but the color range of the flattened image is much smaller than that of the original.

4.3 Practical results

Before we discuss the histogram attack any further, let us consider some results obtained with this attack. We applied the attack to 10 RGB test images shown in Figure **??**. Table 1 shows the estimated percentages of the watermark, where the images are all watermarked with uniform depth 1. The first column lists the results when the attack is directly applied to the images. The second column lists the results when it is applied to the flattened images. It is obvious from this

image	% for J	% for $\bar{J}$
airplane	5.4	30.4
baboon	1.0	7.1
clown	5.6	20.0
cornfield	3.8	17.4
lenna	1.0	26.2
mobcal	5.2	17.5
monarch	22.9	36.7
peppers	5.9	25.8
sail	20.5	10.3
tiffany	54.4	27.9

Table 1. The correctly estimated percentages for 10 test images watermarked with depth 1, when the histogram attack is applied to the original watermarked images, as well as to the flattened images.

table that the results of this attack very much depend on the particular image. Images with smooth histograms are not very vulnerable, while images with a peaky histogram can be extremely vulnerable to this attack. We also see that flattening increases the result in most cases. Figure 3 shows the histograms of the original and watermarked images of baboon and monarch. The histogram of

monarch contains many isolated peaks, which after watermarking are replaced by twin peaks. The very smooth histogram of baboon is almost uneffected by watermarking.

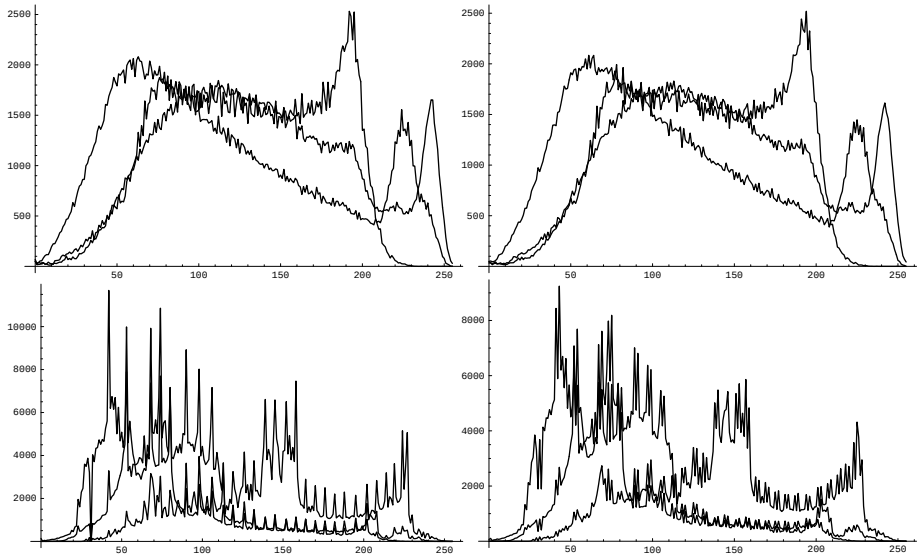

Fig. 3. The histograms of baboon (upper) and monarch (lower), of the original (left) and watermarked (right) images, respectively. All 3 color components are plotted within the same figure.

Histograms of flattened images usually consist of only one peak (ignoring small peaks at 0 and 255 caused by clipping). The width of the peak is related to the vulnerability to the histogram attack. Figure 4 shows the histograms of the flattened watermarked images of baboon and monarch images. The narrower peak of monarch leads to a better estimate.

4.4 Non-zero length sequences

As was illustrated by the influence of the width of the peaks of the flattened images, and as also illustrated by Example 2, the length of non-zero sequences in the histograms have an impact on the achieved estimate of the watermark. Short non-zero sequences yield high estimates. Note that 'non-zero' need not be taken literally: when large numbers are surrounded by relatively small numbers, our analysis applies as well. To illustrate the effect of the length of non-zero sequences, we consider images consisting of uniform noise within a given limited range. So each pixel in the image has any of colors within the range with equal probability. We say we have an image of non-zero length l. We can now compute the result of the histogram attack when it is applied to such an image.

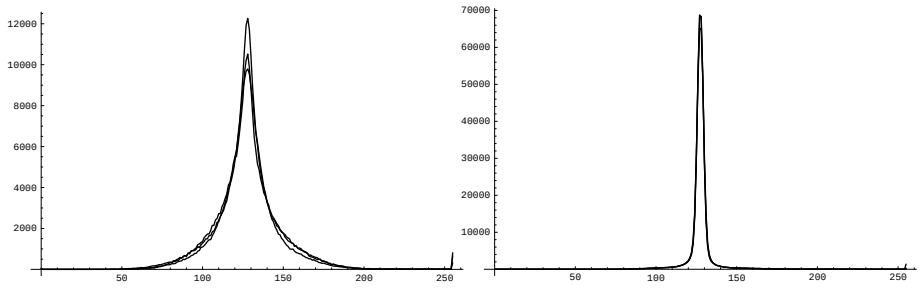

Fig. 4. The histograms of the flattened images of the watermarked baboon (left) and monarch (right). All 3 color components are plotted within the same figure.

Proposition 1. *Let l be an integer with $l \geq 3$ and let I be an image consisting of uniform noise of non-zero length l. Then the histogram attack applied to $J = I_W$ gives an expected estimate of*

$$\frac{2}{l} \cdot 100\%. \tag{11}$$

if I is a gray scale image. If I is an RGB image and the noise in all color components are independent, then the expected estimate is

$$(1 - (\frac{l-2}{l})^3) \cdot 100\%. \tag{12}$$

Proof. Let $x := N/l$. Consider the gray scale image first. Assume for convenience that the noise is in the range $\{101, 102, \ldots, 100 + l\}$. The expected numbers in the histograms of I and J are as follows. We have

	99	100	101	102	...	$100+l-1$	$100+l$	$100+l+1$	$100+l+2$
I	0	0	x	x	...	x	x	0	0
J	0	$x/2$	$x/2$	x	...	x	$x/2$	$x/2$	0

From the histogram of J, we see that $J^{100} + J^{101} + J^{100+l} + J^{100+l+1} = 2x$ pixels are estimated correctly (either $p(-1) = 1$ or $p(+1) = 1$), while for the other pixels, no reasonable estimate can be done because $p(-1) = p(1) = \frac{1}{2}$. For the latter pixels, the expected contribution to the inner product $W.\tilde{W}$ is 0. The correctly estimated pixels contribute $2x$, which corresponds to a fraction $2x/N = 2/l$ of the total inner product. This proves (11).

To prove (12), note that a pixel is correctly estimated when it is among the $2x$ correctly estimated pixels in either of the 3 colors. The probability that a pixel is among the $N - 2x$ remaining pixels (that is the probability that it can *not* be estimated correctly) equals

$$(\frac{N-2x}{N})^3 = (\frac{l-2}{l})^3.$$

So the fraction of correctly estimated pixels equals $1 - (\frac{l-2}{l})^3$.

4.5 The algorithm

We will now sketch the basic algorithm of the histogram attack. We assume we have a fixed depth watermark applied to a gray scale image. Recall that for a DC-free uniform depth $d = 1$ watermark, a randomly chosen pixel is a $+1$ or -1 pixel, each with probability $p = 1/2$. We now consider more general watermarks, with possible depths $d_1 < d_2 < \ldots < d_k$ occurring with probabilities $p_1, p_2, \ldots, p_k$, where $0 < p_i < 1$ for all i, and where $\sum_{i=1}^{k} p_i = 1$. We will say that we have a

$$\{(d_1, p_1), (d_2, p_2), \ldots, (d_k, p_k)\} \tag{13}$$

watermark. So a DC-free uniform depth $d = 1$ watermark is a $\{(-1, 1/2), (1, 1/2)\}$ watermark, and the sum of two such statistically independent watermarks gives a $\{(-2, 1/4), (0, 1/2), (2, 1/4)\}$ watermark. Now consider what happens to the histogram of an image when it is watermarked with a watermark as given in (13). If a pixel in I has gray value h, then after watermarking it has value $h + d_i$ with probability p_i (we then say that the pixel is a d_i pixel). So, if I^h denotes the number of pixels with histogram value h, we find that the expected contribution of these pixels to J^{h+d_i} equals $p_i I^h$.

Note that in all this we ignore clipping effects at the values 0 and 255. Taking care of clipping in this model is not really difficult, but cumbersome to describe in detail. Let us now have a closer look at the steps in the main algorithm.

Step 1. From the histogram of the watermarked image J, a *most likely* histogram of an original, unwatermarked image I is computed.

In principle, from the statistics of the watermark model, the most likely histograms of I (it need not be unique!) can be determined. This however leads to very involved computations, and heuristic approaches as described in Example 2 seem to work well enough. In a greedy way, we determine a best approximation of the histogram of J, by 'building it up' from basic patterns that represent the distribution of the various depths in the watermark. The basic pattern should be proportional to

$$\{p_1 d_1, p_2 d_2, \ldots p_k d_k\}.$$

This pattern is fitted (e.g. from left to right) at the positions in the histogram of J. When it fits a number of times, then at the corresponding position in I, we get the estimate for the histogram of I. The histogram of J is adapted, and we proceed at the next position.

Step 2. From the histogram of I as computed in Step 1, probabilities of being d_k pixels are assigned to the values in the histogram of J.

Given a pixel in the histogram of J with gray value h, this pixel is a d_i pixel with probability

$$P(h, d_i) = \frac{p_i I^{h-d_i}}{\sum_{j=1}^{k} p_j I^{h-d_j}}. \tag{14}$$

Step 3. For each pixel in J, the watermark is estimated based on the probabilities computed in Step 2. If all probabilities p_i are equal to some probability p, then we simply estimate a pixel with gray value h to be a d_i pixel if $P(h, d_i)$ is maximum for this i. If J is really a watermarked image, then all estimated depths are equally likely to occur.

If however the probabilities p_i are not all equal, things are a bit more subtle. If $p_{\max}$ is the maximum of all p_i with corresponding depth $d_{\max}$, we should prevent that all pixels are estimated to be $d_{\max}$ pixels. This can be done by enforcing that the correct fraction of about $p_i N$ pixels, are estimated to be d_i pixels. One way to achieve this which works in practice is to use

$$\tilde{P}(h, d_i) = \frac{I^{h-d_i}}{\sum_{j=1}^{k} p_j I^{h-d_j}} \tag{15}$$

instead of (14. We have divided the right-hand side of (14) by p_i, and the probabilities in (15) should be normalized to speak of probabilities in a correct sense.

4.6 Attacks using many images

The more watermarked images an attacker has at his disposal, the easier it will be to extract the watermark, provided that all images contain the same watermark. Assume we have k watermarked images $J_1, J_2, \ldots, J_k$, all of the same format and containing the same watermark. We then compute k watermark estimates $\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_k$, and we combine these to obtain a final estimate $\tilde{W}$. To illustrate this, consider the example of a uniform depth $d = 1$ watermark for which estimating the watermark boils down to determining the sign of the watermark in all images. So for each of the estimated watermarks $\tilde{W}_j$ with $1 \leq j \leq k$, pixels are estimated to be $+1$ or -1 pixels. A straightforward way of estimating $\tilde{W}$ is by doing a majority vote: a pixel in $\tilde{W}$ is estimated to be $+1$ if it is estimated $+1$ more often in $\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_k$ than it was estimated -1.

We can statistically analyse the effect of combining the estimates, provided that the probabilities of a pixel being estimated correctly in the k images are independent of each other. Suppose for convenience that k is odd, and that in each image, each pixel is estimated correctly with probability p. Then a pixel is estimated correctly in $\tilde{W}$ if it is correctly estimated in at least $(k+1)/2$ images. The probability $P(p, k)$ that this is the case equals

$$P(p, k) = \sum_{j=(k+1)/2}^{k} p^j (1-p)^{k-j} \binom{k}{j}. \tag{16}$$

In Figure 5, we have used (16) to compute the estimated percentages of $\tilde{W}$ as a function of k.

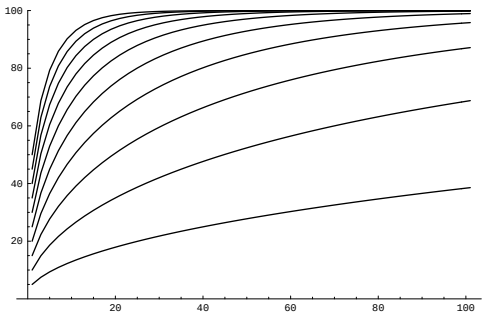

Fig. 5. The estimated percentages of $\tilde{W}$ when the watermark is estimated on the basis of k estimated watermarks with correctly estimated percentages of $5, 10, 15, \ldots, 50\%$. The horizontal axis represents the number k.

5 Conclusions

We have shown that fixed depth watermarks are easily hacked by attackers. Based on only a few images, and without the use of the detector, almost complete knowledge of the watermark can be obtained by the histogram attack. We have also shown how the detector can be misled in the case that only partial knowledge of the watermark can be obtained.

Variable, image dependent depth embedding is absolutely necessary to be more robust against hackers. Even then, some of the principles of the attack described here can be applied, and with enough watermarked images at the disposal of an attacker, good estimates can be obtained.

Histogram analysis may be required before embedding, in order to prevent the histogram attack. In the case of peaky histograms, noise should be added before or after embedding the watermark, or the histograms should be smoothed in other ways.

Some watermark methods are based on changes in the DCT coefficients of an image. Again, if these changes are not image-dependent, an analogous histogram attack in the DCT domain can be applied.

As a final remark, note that histogram analysis may also be used for watermark detection. There may exist applications in which one is not so much worried about malicious attackers, but where the watermark has to survive geometric transformations or cropping. Detection becomes a difficult task under these circumstances, and a layered approach seems to be a good option. Histogram analysis may then well be applied to give a first indication of whether a uniform depth watermark is present in such a transformed image. If the histogram indication is negative, further exhaustive searching is not needed. If the histogram indication is positive, then the analysis can be continued to search for the actual watermark.

References

1. Bender, W., Gruhl, D., Morimoto, N., Lu, A.: Techniques for data hiding, IBM Systems Journal **35** No. 3/4 (1996) 313-336
2. Nikolaidis, N., Pitas, I.: Copyright protection of images using robust digital signatures, IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP-96) **4** (1996) 2168-2171
3. Pitas, I., Kaskalis, T.H.: Applying signatures on digital images, IEEE Workshop on Nonlinear Image and Signal Processing, Neos Marmaras, Greece (1995) 460-463
4. Pitas, I:.A method for signature casting on digital images, IEEE Int. Conf. Image Processing (ICIP'96), vol. III, Lausanne (1996) 215-218
5. Van Den Enden, A.W.M., Verhoeckx, N.A.M.: Discrete Time Signal Processing, Prentice-Hall (1989)
6. Voyatzis, G., I. Pitas, I.:. Chaotic mixing of digital images and applications to watermarking, European Conf. on Multimedia Applications, Services and Techniques (ECMAST'96) **2** Louvain-la-Neuve, Belgium, (1996) 687-695

This article was processed using the LaTeX macro package with LLNCS style