Wavelet-based reversible watermarking for authentication

Jun Tian

Digimarc Corporation, 19801 SW 72nd Avenue, Tualatin, OR 97062, USA

ABSTRACT

In the digital information age, digital content (audio, image, and video) can be easily copied, manipulated, and distributed. Copyright protection and content authentication of digital content has become an urgent problem to content owners and distributors. Digital watermarking has provided a valuable solution to this problem. Based on its application scenario, most digital watermarking methods can be divided into two categories: robust watermarking and fragile watermarking. As a special subset of fragile watermark, reversible watermark (which is also called lossless watermark, invertible watermark, erasable watermark) enables the recovery of the original, unwatermarked content after the watermarked content has been detected to be authentic. Such reversibility to get back unwatermarked content is highly desired in sensitive imagery, such as military data and medical data. In this paper we present a reversible watermarking method based on an integer wavelet transform. We look into the binary representation of each wavelet coefficient and embed an extra bit to "expandable" wavelet coefficient. The location map of all "expanded" coefficients will be coded by JBIG2 compression and these coefficient values will be losslessly compressed by arithmetic coding. Besides these two compressed bit streams, an SHA-256 hash of the original image will also be embedded for authentication purpose.

Keywords: Reversible Watermark, Integer Wavelet Transform, JBIG2, SHA-256, Digital Watermarking, Content Authentication, Digimarc

1. INTRODUCTION

Digital watermarking has grown explosively in the last few years. It embeds an invisible (in some cases, visible) mark (payload) into digital content for the purpose of copyright communication and protection, content authentication, counterfeit deterrence, forensic tracking, connected content, or broadcast monitoring, etc. For a detailed review of digital watermarking, we refer to[1–6] etc.

From the application point of review, most digital watermarking methods can be divided into two categories: robust watermarking and fragile watermarking. Robust watermarking is mainly aimed at copyright protection. Here "robust" means the embedded watermark should be very resistant to various signal processing operations. On the other hand, fragile watermarking is aimed at content authentication. A fragile watermark will be altered or destroyed when the digital content is modified. As a special subset of fragile watermarking, reversible watermarking[4, 7–12] has drawn lots of attention recently. Reversible watermark, (which is also called lossless watermark, invertible watermark, erasable watermark), has an additional advantage such that when watermarked content has been detected to be authentic, one can remove the watermark to retrieve the original, unwatermarked content. Such reversibility to get back unwatermarked content is highly desired in sensitive imagery, such as military data and medical data.

In this paper, we present a reversible watermarking method of digital images. Our method can be applied to digital audio and video as well. Compared with other reversible watermarking methods, our method employs an integer wavelet transform to losslessly remove redundancy in a digital image to allocate space for watermark embedding. The embedding algorithm starts with a reversible color conversion transform. Then, we apply the integer wavelet transform to one (or more) decorrelated component. The purpose of both the reversible color conversion transform and the integer wavelet transform is to remove irregular redundancy in the digital image, such that we can embed regular redundancy into the digital image, for the purpose of content authentication and original content recovery. The regular redundancy could be a hash of the image, a compressed bit stream of

Further author information:
E-mail: jtian@digimarc.com, Phone: 1-503-495-4691, Fax: 1-503-495-4606

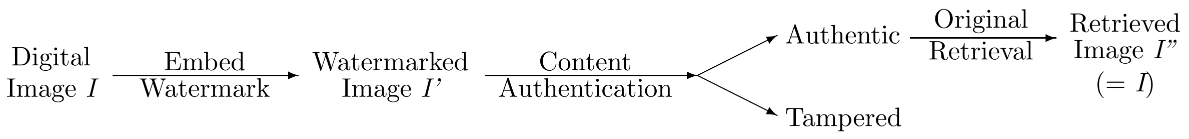
the image, or some other image content dependent watermark. In the integer wavelet domain, we look into the binary representation of each wavelet coefficient and embed an extra bit to "expandable" wavelet coefficient. Besides original content retrieval bit streams, an SHA-256 hash of the original image will also be embedded for authentication purpose.

The paper is organized as follows. We give a brief description of reversible watermarking in Sect. 2. Before presenting our reversible watermarking method, we start with a simple example in Sect. 3. Then we move to our method in Sect. 4, with a detailed explanation of the algorithm and its implementation. The paper is concluded in Sect. 5.

2. THE WHAT, WHY, AND HOW OF REVERSIBLE WATERMARKING

Reversible watermark is a special subset of fragile watermark. Like all fragile watermarks, it can be used for digital content authentication. But reversible watermark is much more than content authentication. It has an additional advantage that when watermarked content has been detected to be authentic, one can remove the watermark to retrieve the original, unwatermarked content.

As illustrated in Fig. 1, we embed a reversible watermark in a digital image *I*, and obtain the watermarked image *I'*. Before sending it to the content authenticator, the image *I'* might have been tampered by some intentional attacker or might not. If the authenticator finds that no tampering happened in *I'*, i.e., *I'* is authentic, then the authenticator will remove the reversible watermark from *I'* and retrieve the original image, which results in a new image *I"*. By definition of reversible watermark, the retrieved image *I"* will be **exactly** the same as the original image *I*, pixel by pixel.



$$\begin{tikzcd}
 & & & \text{Authentic} \arrow[r, "\text{Original}", "\text{Retrieval}"'] & \text{Retrieved Image } I'' \; (= I) \\
\text{Digital Image } I \arrow[r, "\text{Embed}", "\text{Watermark}"'] & \text{Watermarked Image } I' \arrow[r, "\text{Content}", "\text{Authentication}"'] & {} \arrow[ur] \arrow[dr] & & \\
 & & & \text{Tampered} &
\end{tikzcd}$$

Figure 1: Reversible watermark diagram.

The motivation of reversible watermark is distortion-free data embedding.[10] Though one basic requirement of digital watermarking is its imperceptibility, embedding a watermark will inevitably change the original content. Even a very slight change in pixel values may not be desirable, especially in sensitive imagery, such as military data and medical data. In such scenario, every bit of information is important. Any change will effect the intelligence of the digital content, and the access to the original, raw data is always required. Reversible watermarks will provide the original, raw data when the digital content is authentic.

On reversible watermark, it might seem impossible to achieve content authentication, original retrieval, and a very low false positive rate, all at the same time. If we take a digital grayscale image as a two dimensional matrix, with integer entries ranging from 0 to 255, Cox, et al., have shown[4] that "the only way to achieve 100%" content authentication and original retrieval "is to allow for 100% false positives". However, the same authors also noted[4] "in natural images, neighboring pixels are strongly correlated", which implies that the whole set of digital grayscale images is only a small subset of two dimensional integer matrices whose entries ranging from 0 to 255. It is the pixel value correlation exhibited in digital images that enables us to develop a reversible watermark which provides content authentication, original retrieval, and a very low false positive rate, all at the same time.

Let's take a look at Fig. 2, which is the 512 × 512, 8 bit per pixel (bpp) grayscale "Lena" image. If we take out a 3 × 3 region out of it, the pixel values may look like

$$\begin{array}{ccc}
196 & 205 & 207 \\
200 & 206 & 211 \\
201 & 208 & 213
\end{array}$$

Figure 2: 512 × 512, 8 bpp grayscale "Lena".

where pixel values are very close to their neighbors, showing a high redundance or a strong correlation.

The earliest reference to reversible watermark we could find is the Barton patent,[7] filed in 1994. In his invention, the bits to be overlayed will be compressed and added to the bitstring, which will be embedded into the data block. For retrieval, original bits will be decompressed and used to restore modified bits to derive original data block. Honsinger, et al.,[11] reconstruct the embedded watermark after a payload has been decoded from a watermarked image, then subtract the watermark from the watermarked image to losslessly recover the original image. Fridrich, et al.,[9] use a JBIG lossless compression scheme for compressing some least significant bit (LSB) planes to make space to insert an image hash. Later[10] they utilize an order-2 function (whose inverse function is itself) for high-capacity data embedding that is distortion-free. For more details of these methods and other reversible watermark methods, we refer to,[4, 7–12] etc.

Our reversible watermark method is based on an integer wavelet transform, JBIG2 compression, and arithmetic coding. In a wide-sense, Barton's method, Fridrich's methods, and our method could all be categorized as "compression-based reversible watermark". Let's start with a simple example which will illustrate the basic idea behind our reversible watermark method.

3. A SIMPLE EXAMPLE

Assume we have two grayscale values (x, y), where $x, y \in \mathbf{Z}$, $0 \le x, y \le 255$, and we would like to embed one bit b, with $b \in \{0, 1\}$ into (x, y) in a reversible way. More specifically, let's assume

$$x = 205\,,\ y = 200\,,\ \text{and } b = 0\,.$$

First we compute the average and difference of x and y,

$$l = \left\lfloor \frac{x+y}{2} \right\rfloor = \left\lfloor \frac{205+200}{2} \right\rfloor = \left\lfloor \frac{405}{2} \right\rfloor = 202\,,\ h = x - y = 205 - 200 = 5\,,$$

where the symbol $\lfloor \cdot \rfloor$ denotes the integer part of a number. For example, $\lfloor 2.7 \rfloor = 2$, $\lfloor -1.2 \rfloor = -2$. Next we expand the difference number h into its binary representation

$$h = 5 = 101_2 \, .$$

Then we add b into the binary representation of h at the location right after the most significant bit (MSB) (note that the MSB is always 1),

$$h' = 1b01_2 = 1001_2 = 9 \, .$$

Finally we compute the new grayscale values, based on the new difference number h' and the original average number l,

$$x' = l + \left\lfloor \frac{h'+1}{2} \right\rfloor = 202 + \left\lfloor \frac{9+1}{2} \right\rfloor = 207 \, , \; y' = x' - h' = 207 - 9 = 198 \, .$$

The above embedding process is also illustrated in Fig. 3.

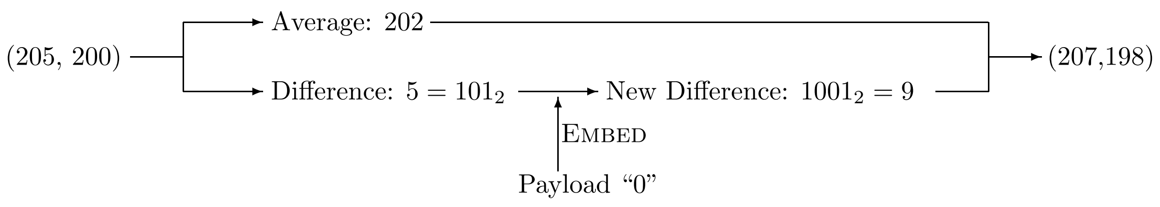


Figure 3: A simple example.

From the embedded pair (x', y'), the watermark detector (or authenticator) can extract the embedded bit b and get back the original pair (x, y) by a similar process as the embedding. Again we compute the average and difference

$$l' = \left\lfloor \frac{x'+y'}{2} \right\rfloor = \left\lfloor \frac{207+198}{2} \right\rfloor = 202 \, , \; h' = x' - y' = 207 - 198 = 9 \, .$$

Look into the binary representation of h'

$$h' = 9 = 1001_2 \, ,$$

extract the second most significant bit, which is "0" in this case, as the embedded bit b, which leaves

$$h'' = 101_2 = 5 \, .$$

Now with the average number l' and difference number h'', we can retrieve exactly the original grayscale-valued pair (x, y).

In the above example, although the embedded pair $(207, 198)$ is still 8 bpp, we have embedded an extra bit by increasing the bit length of the difference number h from 3 bits (which is number 5) to 4 bits (which is number 9). And such embedding process is totally reversible. If we have a sequence of pairs of grayscale values $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$, where $x_i, y_i \in \mathbf{Z}$, $0 \le x_i, y_i \le 255$, $1 \le i \le n$, we can embed the payload $b = \{b_1, b_2, \cdots, b_n\}$, where $b_i \in \{0, 1\}, 1 \le i \le n$, by repeating the above process,

$$l_i = \left\lfloor \frac{x_i + y_i}{2} \right\rfloor \, , \; h_i = x_i - y_i \, , \; 1 \le i \le n \, .$$

For each difference number h_i, expand it into its binary representation,

$$h_i = r_{i,0} r_{i,1} \cdots r_{i,j(i)} \, ,$$

where $r_{i,0} = 1$ is the MSB, $r_{i,m} \in \{0, 1\}$, for $1 \le m \le j(i)$, with $j(i) + 1$ as the bit length of h_i in its binary representation. Then we could embed b_i into h_i by

$${h_i}' = r_{i,0} b_i r_{i,1} \cdots r_{i,j(i)} \, .$$

Alternatively, we can combine all the bits $r_{i,m} \in \{0,1\}$, with $1 \le m \le j(i)$, $1 \le i \le n$ (note that we do not select the MSBs), and $b = \{b_i\}$ into a single bit stream

$$\mathcal{B} = r_{1,1}r_{1,2}\cdots r_{1,j(1)}r_{2,1}r_{2,2}\cdots r_{2,j(2)}\cdots r_{n,1}r_{n,2}\cdots r_{n,j(n)}b_1b_2\cdots b_n\,,$$

and use a reversible mapping f (which could be encryption, lossless compression, or other invertible operation, or combination of some of them) to form a new bit stream $\mathcal{C}$

$$\mathcal{C} = f(\mathcal{B}) = c_1c_2\cdots c_k\,,$$

where $c_i \in \{0,1\}$, for $1 \le i \le k$, with k as the bit length of $\mathcal{C}$. Then we could embed $\mathcal{C}$ into the difference numbers $h_i, 1 \le i \le n$, by

$$h_i{}' = r_{i,0}c_{s(i-1)+1}c_{s(i-1)+2}\cdots c_{s(i)}\,,$$

where $c_{s(i-1)+1}c_{s(i-1)+2}\cdots c_{s(i)}$ is a truncated subsequence of $\mathcal{C}$ with

$$s(0) = 0\,, \text{ and } s(i) = s(i-1) + j(i) + 1\,.$$

The bit length of $h_i{}'$ is still one more than that of h_i. For detection, as f is reversible, we can get back $\mathcal{B}$ by

$$\mathcal{B} = f^{-1}(\mathcal{C})\,,$$

and consequently we can get back the original pairs $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$.

The reason that we could increase the bit length of the difference number h is because of the high redundance in pixel values of natural images. Thus in most cases h will be very small and have a short bit length in its binary representation. In an edge area or an area containing lots of activity, the difference number h from a pair of grayscale values could be large. For example, if $x = 105$, $y = 22$, then $h = x - y = 83 = 1010011_2$. If we embed a bit "0" into h, $h' = 10010011_2 = 147$. With $l = 63$ unchanged, the embedded pair will be $x' = 137$, $y' = -10$. This will cause an underflow problem as grayscale values are restricted to the range [0, 255]. In the next section, we will present the definition of "expandable" pairs, which will prevent overflow and underflow problems. Working with these "expandable" pairs, we are assured to achieve the goal of content authentication, original retrieval, and a very low false positive rate, all at the same time.

4. ALGORITHM AND IMPLEMENTATION

As illustrated in Fig. 4, our reversible watermark embedding method consists of six steps: reversible color conversion transform, integer wavelet transform, JBIG2 compression of the location map, arithmetic coding of wavelet coefficient values, SHA-256 hash, and embedding. We will discuss each of them in the follow subsections.

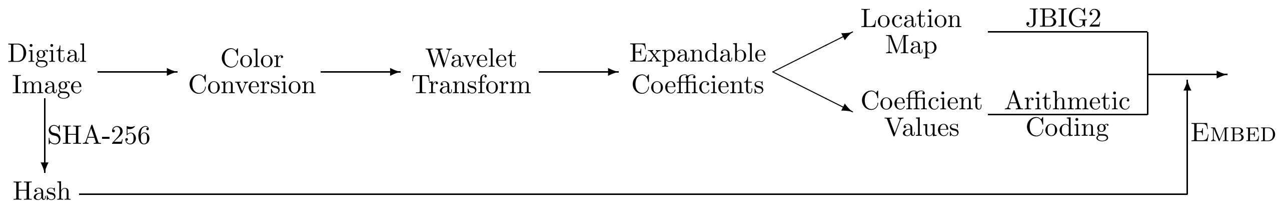


Figure 4: Reversible watermark embedding algorithm.

4.1. Reversible Color Conversion

Reversible color conversion transform[13] decorrelates the dependence among different color components to a large extent. It is a lossless color transform and the transform output is still integer-valued. For a *RGB* color image, the reversible color conversion transform is

$$\begin{aligned} Yr &= \left\lfloor \frac{R+2G+B}{4} \right\rfloor, \\ Ur &= R-G, \\ Vr &= B-G. \end{aligned}$$

Its inverse transform will be

$$\begin{aligned} G &= Yr - \left\lfloor \frac{Ur+Vr}{4} \right\rfloor, \\ R &= Ur+G, \\ B &= Vr+G. \end{aligned}$$

The reversible color conversion transform maps a grayscale-valued triplet to an integer-valued triplet. It can be thought of as an integer approximation of the CCIR 601 standard which provides a conversion to $YCrCb$ space defined by the following matrix

$$\begin{pmatrix} Y \\ Cr \\ Cb \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.500 & -0.419 & -0.081 \\ -0.169 & -0.331 & 0.500 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

The RGB to $YCrCb$ transform matrix is not integer-valued. It requires floating point computing. Such transform will introduce small roundoff errors, and will be not a reversible transform. As reversible watermarking requires original retrieval with 100% accuracy, we choose the reversible color conversion transform instead of the RGB to $YCrCb$ transform.

For a grayscale image there will be no reversible color conversion transform as we apply the next step, the integer wavelet transform, directly.

4.2. Integer Wavelet Transform

The integer wavelet transform[14] maps integers to integers and allows for perfect invertibility with finite precision arithmetic (i.e. reversible). The wavelet filters for integer wavelet transforms are dyadic rational,[15] i.e., integers or rational numbers whose denominators are powers of 2, like 13/4, -837/32. Thus the integer wavelet transform can be implemented with only three operations, addition, subtraction, and shift, on a digital computer. The fast multiplication-free implementation is another advantage of the integer wavelet transform over standard discrete wavelet transform.

For example, for the Haar wavelet filter, the integer wavelet transform will be the average and difference calculation we used in Sect. 3,

$$l_i = \left\lfloor \frac{x_{2i}+x_{2i+1}}{2} \right\rfloor, \; h_i = x_{2i} - x_{2i+1}.$$

And for the BCW-3 filters,[15] a biorthogonal filter pair with four vanishing moments for all four filters, the integer wavelet transform will be

$$h_i = x_{2i+1} - \left\lfloor \frac{9}{16}(x_{2i}+x_{2i+2}) - \frac{1}{16}(x_{2i-2}+x_{2i+4}) + \frac{1}{2} \right\rfloor, \; l_i = x_{2i} + \left\lfloor \frac{9}{32}(h_{i-1}+h_i) - \frac{1}{32}(h_{i-2}+h_{i+1}) + \frac{1}{2} \right\rfloor.$$

In this paper, we use will the Haar integer wavelet transform as an example to illustrate our reversible watermark method. The generalization to other integer wavelet transforms will be straightforward and is omitted here.

After the reversible color conversion transform, we apply the integer wavelet transform to one (or more) decorrelated component. In our implementation , we choose the Yr component, which is the luminance component. For a grayscale image, as mentioned in Sect. 4.1, we apply the integer wavelet transform directly to the whole image.

4.3. Expandable Wavelet Coefficient

As seen in Sect. 3, for the grayscale-valued pair $(105, 22)$ and a payload bit "0" (or "1"), a brute-force embedding will cause an underflow problem. Now we will study how to prevent the overflow and underflow problems.

For a grayscale-valued pair (x, y), where $x, y \in \mathbf{Z}$, $0 \le x, y \le 255$, as in Sect. 3, define the average and difference as

$$l := \left\lfloor \frac{x+y}{2} \right\rfloor , \; h := x - y \, .$$

Then the inverse transform to get back (x, y) from the average number l and difference number h is

$$x = l + \left\lfloor \frac{h+1}{2} \right\rfloor , \; y = l - \left\lfloor \frac{h}{2} \right\rfloor . \tag{1}$$

Thus to prevent the overflow and underflow problems, i.e., to restrict x, y in the range of [0, 255] is equivalent to have

$$0 \le l + \left\lfloor \frac{h+1}{2} \right\rfloor \le 255 \, , \; 0 \le l - \left\lfloor \frac{h}{2} \right\rfloor \le 255 \, .$$

Since both l and h are integers, one can derive that the above inequalities are equivalent to

$$|h| \le 2(255 - l) \, , \text{ and } |h| \le 2l + 1 \, . \tag{2}$$

Condition (2) sets a limit on the absolute value of the difference number h. As long as h is in such range, it is guaranteed that (x, y) computed from Eqn. (1) will be grayscale values. Furthermore, Condition (2) is equivalent to

$$\begin{cases} |h| \le 2(255 - l) \, , & \text{if} \quad 128 \le l \le 255 \\ |h| \le 2l + 1 \, , & \text{if} \quad 0 \le l \le 127 \end{cases}$$

With the above condition, we now define an expandable grayscale-valued pair.

Definition 4.1. *For a grayscale-valued pair (x, y), where $x, y \in \mathbf{Z}$, $0 \le x, y \le 255$, define*

$$l = \left\lfloor \frac{x+y}{2} \right\rfloor , \; h = x - y \, .$$

Then (x, y) is an expandable *pair if and only if*

$$h \ne 0 \, , \; \text{and } 2^{\lfloor \log_2 |h| \rfloor + 2} - 1 \le \min\left(2(255 - l), 2l + 1\right) \, .$$

Note that if $h \ne 0$, the bit length of the binary representation of $|h|$ is $\lfloor \log_2 |h| \rfloor + 1$. Thus $2^{\lfloor \log_2 |h| \rfloor + 2} - 1$ is the largest number whose bit length is one more than that of $|h|$. Thus for an expandable pair (x, y), if we embed an extra bit ("0" or "1") into the binary representation of the difference number h at the location right after the MSB, the new difference number h' still satisfies Condition (2), that is, the new pair computed from Eqn. (1) is guaranteed to be grayscale values. For simplicity, we will also call h expandable if (x, y) is an expandable pair.

Thus from the average number l, one can tell whether or not a difference number h is expandable, i.e., whether or not the bit length of h could be increased by 1 without causing any overflow or underflow problem. Further we define the changeable bits of h as

Definition 4.2. *For a grayscale-valued pair (x, y), assume $h \ne 0$, and the binary representation of $|h|$ is*

$$|h| = r_0 r_1 \cdots r_j \, ,$$

where $r_0 = 1$, $r_m \in \{0, 1\}$, for $1 \le m \le j$, with $j \ge 0$, and $j + 1$ is the bit length. If $g \le j$ is the largest number such that

$$\left(\sum_{i=0}^{j-g} r_i 2^{j-i} \right) + 2^g - 1 \le \min(2(255 - l), 2l + 1) \, ,$$

then we say (x, y), or equivalently h, has g changeable *bits, and they are* $r_{j-g+1}, r_{j-g+2}, \cdots, r_j$.

Since

$$|h| = r_0 r_1 \cdots r_j = \sum_{i=0}^{j} r_i 2^{j-i} ,$$

by definition, h has g changeable bits if the last g bits in the binary representation are all changed to "1", it still satisfies Condition (2), or the new pair computed from Eqn. (1) is still grayscale values. Let's look at two extreme cases:

- If $g = 0$, then h has no changeable bits.
- If $g = j$, then all bits (excluding the MSB) in its binary representation are changeable. It is clear that if h is expandable, then $g = j$. However the inverse is not true, i.e., $g = j$ does not imply h is expandable.

The number "0" does not have a proper binary representation. We can increase it (along with all positive numbers) by 1 to fit it into the definition of expandable and changeable. With such preparation, we extract bits from wavelet coefficients as follows:

1. For the Yr component of a color image or a grayscale image, apply the integer wavelet transform.
2. If $h_i \geq 0$ and $l_i < 255$, we increase h_i by 1, $h_i = h_i + 1$.
3. Construct a bit stream $\mathcal{R}$, which consists of changeable bits from all h_i. The scanning order of h_i is determined by a fixed pattern (for example, zigzag).

4.4. JBIG2 Compression

For a grayscale-valued pair (x, y), by Definition 4.1, we can tell whether or not it is expandable. When (x, y) has been modified by the embedder, it will not be clear to the watermark detector whether or not the original pair has been expanded, i.e., whether the bit length of the binary representation of the difference number has been increased by 1 (thus larger than the original one), or it is the same as the original one. In order to remove the watermark and retrieve the original, unwatermarked image, the detector needs to know the location of expanded difference numbers h in the original image.

We can define a location map of expanded difference numbers by setting its value to "1" at each location when it is expanded or "0" otherwise. The location map can be viewed as a bi-level image. To store the location map, we can losslessly compress the bi-level image and store the compressed bit stream instead. We will employ JBIG2, the new international standard for lossless compression of bi-level images, to compress the location map of expanded difference numbers h. For convenience, we will denote the JBIG2 compressed bit stream of the location map of expanded h as $\mathcal{J}$. Alternatively, the location map could be compressed by run-length coding.

4.5. Arithmetic Coding

To make more room for embedding the payload, we could further losslessly compressed the collected bit stream $\mathcal{R}$, which are all the changeable bits from difference numbers h. Either arithmetic coding[16] or Huffman coding could be used for this purpose. In our implementation, we use arithmetic coding

$$\mathcal{C} = \text{ArithmeticCoding}(\mathcal{R}) ,$$

where $\mathcal{C}$ is the compressed bit stream from the arithmetic coding.

4.6. SHA-256 Hash

To authenticate a watermarked image and detect tampering, we will embed a hash of the image into itself. The new hash algorithm SHA-256 is a 256-bit hash function that is intended to provide 128 bits of security against collision attacks. SHA-256 is more consistent with the new encryption standard, the Advanced Encryption Standard (AES) algorithm, than SHA-1, which provides no more than 80 bits of security against collision attacks. We calculate the SHA-256 hash of the digital image (before the reversible color conversion transform) and denote the hash as $\mathcal{H}$.

4.7. Embedding

With the compressed bit stream $\mathcal{J}$ of the location map, the compressed bit stream $\mathcal{C}$ of changeable bits, and the SHA-256 hash $\mathcal{H}$ (a 256 bit stream), we are ready to embed all three of them into changeable bits of difference numbers h in the integer wavelet domain. First we combine them into one big bit stream

$$\mathcal{S} = \mathcal{J} \cup \mathcal{C} \cup \mathcal{H} = s_1 s_2 \cdots s_k \,,$$

where $s_i \in \{0,1\}$, $1 \le i \le k$, k is the bit length of $\mathcal{S}$. Here we append $\mathcal{C}$ to the end of $\mathcal{J}$, and append $\mathcal{H}$ to the end of $\mathcal{C}$. The order of $\mathcal{J}$, $\mathcal{C}$, and $\mathcal{H}$ could be changed, as long as the embedder and the detector use the same order. Next we design a pseudo random scanning order for all the difference numbers h. This pseudo random order will be different from the scanning order used to construct the changeable bit stream $\mathcal{R}$. With the pseudo random order of h, we embed the bit stream $\mathcal{S}$ into h by replacing (part of) changeable bits. For expandable h, we increase the bit length of h by 1, thus increase the number of changeable bits by 1. Here is a description of the embedding:

```
1. Assume all difference numbers $h$ are ordered by the pseudo random order as $h_1, h_2, \cdots, h_n$.

2. Set $i = 1$.

3. If $i \le n$ and $k > 0$,
   ·  If $h_i$ is expandable, $|h_i| = r_0 r_1 \cdots r_j$, and $g = j$,
   ·      Set $|h_i| = r_0 0 r_1 \cdots r_j$, now $|h_i|$ has $j+1$ changeable bits.
   ·  Replace changeable bits in $h_i$ with $s_{k-g+1}, s_{k-g+2}, \cdots, s_k$.
   ·      For $m = 1 : g$
   ·          $r_{j-g+m} = s_{k-g+m}$.
   ·  If $h_i > 0$,
   ·      Set $h_i = h_i - 1$.
   ·  Set $i = i + 1$, $k = k - g$.

4. Go to Step 3.
```

We modify only the absolute value of h, and keep the sign (and its MSB) unchanged. If h is non-negative, since it has been increased by 1 in Sect. 4.3, after bit replacement, positive h will have its value decreased by 1.

We embed the bit stream $\mathcal{S}$ by replacing changeable bits in difference numbers h. The capacity of all changeable bits will be much larger than the bit length of $\mathcal{S}$. For example, the capacity of all changeable bits (including expanded bits) of the "Lena" image in Fig. 2 is about 330,000 bits, while $\mathcal{S}$ is about 210,000 bits. There are about 120,000 bits surplus, which is 0.45 bpp for the image size 512×512. This is a huge extra space which could embed additional information (such as a compressed bit stream of the image for locating tampering and recovery). So after embedding all bits in $\mathcal{S}$, a large portion of changeable bits will not be changed. We can select changeable bits based on how much difference it will introduce (how much it degrades the image quality) if it is changed during the embedding. We treat two difference cases here, non-expandable h and expandable h.

As we notice, modifying changeable bits in non-expandable h brings imperceptible changes to images. For example, for the "Lena" image in Fig. 2, if we restrict ourselves by not increasing the bit length of expandable h, and modify changeable bits only, then the worst possible distorted image is when we set changeable bits in h to be all equal to 1 or all equal to 0, depending on each h's value. Figure 5 shows the modified "Lena" image in such a worst possible case. Although the pixel value difference between the original image in Fig. 2 and the distorted one in Fig. 5 is as large as 32, the visual difference between them is almost imperceptible.

For expandable h, if we increase its bit length by 1 and embed one more bit into it, the visual quality degradation could be very noticeable when $|h|$ is large, like in an edge area or an area containing lots of activity. To achieve best image quality, the extra changeable bits which are not used for embedding should be allocated to those expandable h with large absolute values. If $|h|$ is large, even if h is expandable, we will treat it as non-expandable by turning it off to "0" in the location map. The study of selecting optimal changeable bits and expandable bits is beyond the scope of this paper and will be reported in a forthcoming paper.

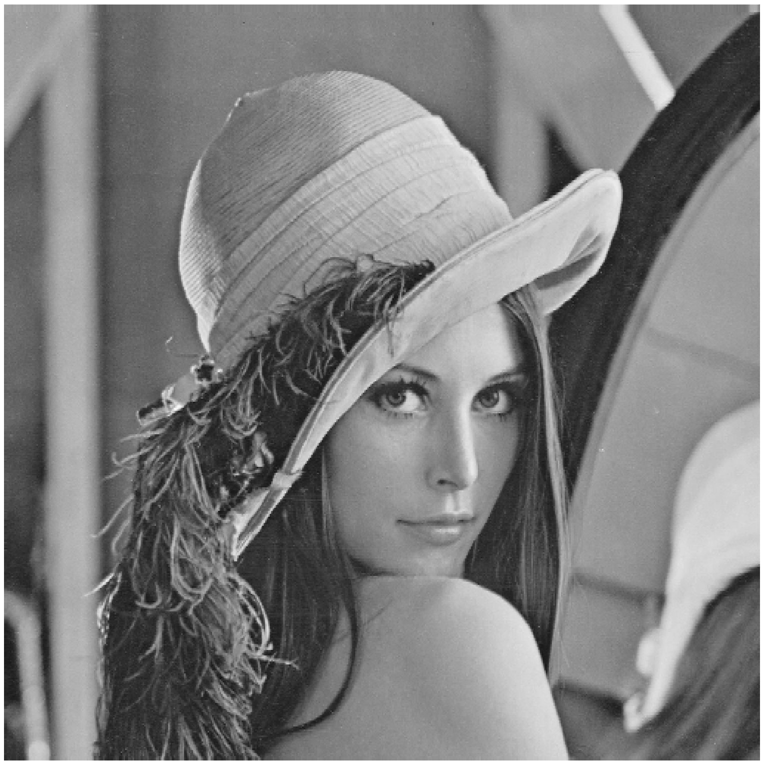


Figure 5: Worst possible image quality when changing all changeable bits, but no expanded bits.

As an example, we embed the reversible watermark into the "Lena" image of Fig. 2. Changeable bits of all difference numbers h are modified, and only small expandable h have their bit lengths increased by 1 (one extra expanded bit), to embed the bit stream $\mathcal{S}$. The watermarked image is shown in Fig. 6. From Fig. 6, the authenticator can remove the reversible watermark and retrieve the original "Lena" image, which will be exactly the same as in Fig. 2, pixel by pixel. The original retrieval algorithm is explained in the next subsection.

For security reasons, the compressed bit streams $\mathcal{J}$ and $\mathcal{C}$ from JBIG2 and arithmetic coding can be encrypted by the AES algorithm, before they are embedded into changeable bits of difference numbers h.

4.8. Authentication

Before we describe the content authentication and original retrieval algorithm, we state a lemma of changeable bits. The proof of the lemma is elementary and will be omitted here.

Lemma 4.3. *As in* Definition 4.2*, assume h has g changeable bits, and its binary representation is*

$$|h| = r_0 r_1 \cdots r_j \, .$$

If we arbitrarily change its changeable bits,

$$|h'| = r_0 r_1 \cdots r_{j-g} r'_{j-g+1} r'_{j-g+2} \cdots r'_g \, ,$$

where $r'_{j-g+i} \in \{0,1\}$, $1 \le i \le g$, then the new pair defined by Eqn. (1) *is still grayscale-valued, and the changeable bits of h' is exactly g.*

Since the embedder does not change the average numbers l, the authenticator will derive exactly the same number of changeable bits in the difference number as the embedder. For expanded h whose bit length of its binary representation has been increased by 1 during the embedding, the authenticator will know such

Figure 6: Reversibly watermarked image.

information from the location map. Thus the authenticator knows exactly which bits have been replaced and which difference numbers are expanded (by one bit) during the embedding process. All these are crucial to retrieve back the original, unwatermarked image with 100% accuracy.

The authentication algorithm is illustrated in Fig. 7. Similar to the embedding algorithm, we go through a reversible color conversion transform and the integer wavelet transform. From wavelet coefficients, we extract all changeable bits, ordered by the same pseudo random order of the embedding, as in Sect. 4.7. From the first segment of extracted bits, we decompress the location map of expanded difference numbers h. From the second segment, we decompress the original changeable bits values. The third segment will give the embedded hash. With Lemma 4.3, we know which bits are modified and which bits are extra expanded bits during the embedding. Thus we can reconstruct an image by replacing changeable bits with decompressed changeable bits. Then we compare the extracted hash and the SHA-256 hash of the reconstructed image. If they match bit by bit, then the watermarked image is authentic, and the reconstructed image is exactly the original, unwatermarked image.

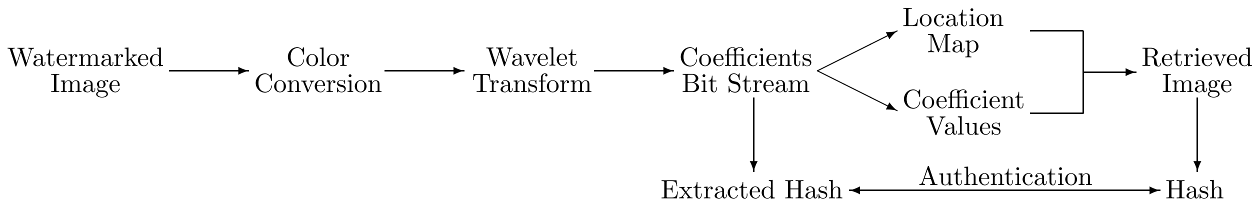


Figure 7: Reversible watermark authentication algorithm.

5. CONCLUSIONS

In this paper we present a reversible watermarking method based on the integer wavelet transform. The location map of expanded wavelet coefficients, changeable bits of all coefficients, and an SHA-256 hash are embedded. With a carefully designed location map, the watermarked image quality is superb. The authenticator can remove the reversible watermark and retrieve an image, which is exactly the same as the original, unwatermarked image, pixel by pixel.

ACKNOWLEDGMENTS

We would like to thank Hugh Brunk, Sean Calhoon, Steve Decker, Neil Lofgren, Joel Meyer, Phil Patterson, Burt Perry, Geoff Rhoads, Tony Rodriguez, Ravi Sharma, and John Stach of Digimarc Corporation for helpful discussion and valuable assistance.

REFERENCES

1. A. M. Alattar, "Smart images using Digimarc's watermarking technology," in *Security and Watermarking of Multimedia Contents II*, P. W. Wong and E. J. Delp III, eds., *Proceedings of SPIE* **3791**, pp. 264–273, 2000.
2. M. Barni, F. Bartolini, I. J. Cox, J. Hernandez, and F. Pérez-Gonzalez, eds., *IEEE Communications Magazine, special issue on digital watermarking for copyright protection: a communications perspective*, **39**, Aug. 2001.
3. V. Cappellini, M. Barni, and F. Bartolini, eds., *Signal Processing, special issue on information theoretic aspects of digital watermarking*, **81**, June 2001.
4. I. J. Cox, M. L. Miller, and J. A. Bloom, *Digital Watermarking*, Morgan Kaufmann Publishers, San Francisco, CA, 2001.
5. S. Katzenbeisser and F. A. P. Petitcolas, eds., *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House, Norwood, MA, 2000.
6. P. W. Wong and E. J. Delp III, eds., *Security and Watermarking of Multimedia Contents III, Proceedings of SPIE*, **4314**, 2001.
7. J. M. Barton, "Method and apparatus for embedding authentication information within digital data," *United States Patent* **5,646,997**, 1997.
8. G. Coatrieux, H. Maître, and B. Sankur, "Strict integrity control of biomedical images," in *Security and Watermarking of Multimedia Contents III*, P. W. Wong and E. J. Delp III, eds., *Proceedings of SPIE* **4314**, pp. 229–240, 2001.
9. J. Fridrich, M. Goljan, and R. Du, "Invertible authentication," in *Security and Watermarking of Multimedia Contents III*, P. W. Wong and E. J. Delp III, eds., *Proceedings of SPIE* **4314**, pp. 197–208, 2001.
10. M. Goljan, J. Fridrich, and R. Du, "Distortion-free data embedding for images," in *4th Information Hiding Workshop*, Apr. 2001.
11. C. W. Honsinger, P. W. Jones, M. Rabbani, and J. C. Stoffel, "Lossless recovery of an original image containing embedded data," *United States Patent* **6,278,791**, 2001.
12. C. D. Vleeschouwer, J. F. Delaigle, and B. Marq, "Circular interpretation of histogram for reversible watermarking," in *IEEE 4th Workshop on Multimedia Signal Processing*, Oct. 2001.
13. M. J. Gormish, E. L. Schwartz, A. Keith, M. Boliek, and A. Zandi, "Lossless and nearly lossless compression for high quality images," in *Very High Resolution and Quality Imaging II*, V. R. Algazi, S. Ono, and A. G. Tescher, eds., *Proceedings of SPIE* **3025**, pp. 62–70, 1997.
14. A. R. Calderbank, I. Daubechies, W. Sweldens, and B. L. Yeo, "Wavelet transforms that map integers to integers," *Applied and Computational Harmonic Analysis* **5**(3), pp. 332–369, 1998.
15. J. Tian and R. O. Wells, Jr., "Dyadic rational biorthogonal Coifman wavelet systems," Rice CML Technical Report CML9607, Computational Mathematics Laboratory, Rice University, Houston, TX, 1996. http://math.rice.edu/~juntian/publications/CML9607.ps.zip.
16. I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM* **30**, pp. 520–540, June 1987.