Journal of Information Hiding and Multimedia Signal Processing ©2012 ISSN 2073-4212
Ubiquitous International Volume **3**, Number **4**, October **2012**

Reversible Watermark Combining Pre-processing Operation and Histogram Shifting

Shaowei Weng

Information Engineering College
Guangdong University of Technology, P. R. China

Jeng-Shyang Pan*

Innovative Information Industrial Research Center
Shenzhen Graduate School, Harbin Institute of Technology
jspan@cc.kuas.edu.tw; *Corresponding Author

Xia Gao

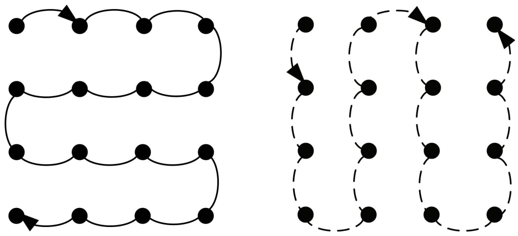
Heibei North University

Received June 2012; revised August 2012

Abstract. *A reversible watermark scheme employing a pre-processing operation before integer Haar wavelet transform (IHWT) is presented in this paper. After two consecutive pixels are pre-processed, IHWT is applied to the resulting pixels to obtain their difference. Histogram shifting is adopted to embed watermark into selected differences. Due to the application of pre-processing, after watermark embedding, for the right one of two watermarked pixels, no change occurs, i.e., it is just the same with its original value. Hence, this pixel can be used again and grouped into a new pair with its right-neighboring pixel in raster scan order. As a result, the number of differences used for embedding are greatly increased, and correspondingly, the embedding capacity is also considerably enhanced. Experimental results reveal the proposed scheme is effective.*
Keywords: Reversible Watermarking, Pre-processing operation

1. **Introduction.** Reversible watermarking based on difference expansion is proposed by Tian [1]. The differences between two pixels are expanded to carry watermark information if neither overflow nor underflow occurs. Coltuc *et al.* propose a threshold-controlled embedding scheme based on an integer transform for pairs of pixels [2]. In Thodi's recent work [3], histogram shifting is incorporated into Tian's method to produce a new algorithm called Alg. D2 with a overflow map. Although Alg. D2 has achieved high performance relative to Tian's method, its embedding rates still can not exceed 0.5 bpp (bit per pixel) for a single embedding processing. According to the performance comparison figure in paper [3], a inflexion point will appear at around 0.5 when multiple embedding is applied to achieve rates close to 1 bpp. Though Weng's method [4] can achieve embedding rates of about 1 bpp for a single embedding process, it is not capable of embedding small payloads at low distortions for some medical images.

The number of differences which occupies only half the size of images is the main reason leading to the rates less than 0.5 bpp. Therefore, the main aim of the proposed method is to greatly improve the number of differences. Therefore, a pre-processing operation is presented in the proposed method. After one pair composed of two neighboring pixels are pre-processed, IHWT is applied to the resulting pair to obtain their difference. Histogram shifting is adopted, which adds watermark to differences or shifts difference histogram according to the occurrences of differences. Due to the application of pre-processing, after watermark embedding, the right pixel of the watermarked pair is kept unaltered. Hence, this pixel can be used again and is paired with its right-neighboring pixel in raster scan order. Since every two neighboring pairs have 1-pixel overlap, the number of differences is increased twice compared with Tian's method and Alg. D2, and meanwhile, the embedding rate of around 1 bpp can be achieved.



(a) Embedding (b) Extraction

FIGURE 1. Grouping way for a 4 × 4 images block

Experimental results reveal that the proposed scheme outperforms Alg. D2 at almost all embedding rates and especially at low rates.

2. **The proposed method.** In Tian's method, the forward integer transform $T(\cdot)$ given by Eq. (1) is applied to one pixel pair denoted by (x, y) to calculate its average value l and difference value d.

$$\begin{aligned} l &= \lfloor \tfrac{x+y}{2} \rfloor \\ d &= x - y \end{aligned} \quad (1)$$

The inverse integer transform $(x, y) = T^{-1}(l, d)$ is defined as

$$\begin{aligned} x &= l + \lfloor \tfrac{d+1}{2} \rfloor \\ y &= l - \lfloor \tfrac{d}{2} \rfloor \end{aligned} \quad (2)$$

In Tian's method, some original difference values with small amplitude need to be multiplied by 2 so that the watermark is inserted into their vacant least significant bits (LSBs). In recent several years, due to the demand for the embedding distortion increased considerably, the distortion introduced by enlarging one time for the difference values can not be ignored. Hence, in the proposed method, the distortion is largely reduced by adopting histogram shifting which directly adds 1-bit watermark to some selected difference values.

Otherwise, since every two neighboring pixels is grouped into one pixel pair, the number of differences is just half the size of the original images. Therefore, increasing the number of differences will help to the enhancement of the potential embedding capacity. In the proposed method, a pre-processing operation is employed before the transform $T(\cdot)$. After one pixel pair is pre-processed, $T(\cdot)$ is applied to the resulting pair to obtain their difference value and average value. Note that differences are preserved unchanged before and after pre-processing. Watermark is embedded into differences by shifting difference histogram. After watermark embedding, for the right pixel, i.e., y, no change occurs due to the application of pre-processing. Hence, this pixel can be used again and grouped into a new pair with its right-neighboring pixel in raster order. As a result, the number of available differences are largely increased to the same size of the original images, and correspondingly, the embedding capacity is considerably enhanced.

Histogram shifting and pre-processing are respectively described as follow.

2.1. **Histogram shifting.** Histogram shifting is formulated as

$$d = \begin{cases} sign(d) \times (|d| + b), \ \texttt{if} \ |d| = M_d \\ sign(d) \times (|d| + 1), \ \texttt{if} \ |d| > M_d \\ d, \texttt{if} \ |d| \leq M_d - 1 \ \texttt{and} \ M_d \geq 1 \end{cases} \quad (3)$$

where the symbol $|d|$ denotes the absolute value of d, M_d is the absolute value of differences having the largest occurrence, $sign(d)$ is 1 if $d > 0$, 0 if $d = 0$, and -1 if $d < 0$, b denotes 1-bit watermark, $b \in \{0, 1\}$.

After adding one bit watermark b to d $(d = M_d)$, $x^{'}$ and $y^{'}$ obtained via Eq. (4) is given below.

$$\begin{aligned} x^{'} &= l + \lfloor \tfrac{M_d + sign(M_d) \times b + 1}{2} \rfloor \\ y^{'} &= l - \lfloor \tfrac{M_d + sign(M_d) \times b}{2} \rfloor \end{aligned} \quad (4)$$

When $d = M_d$, we analyze the difference between $(x^{'}, y^{'})$ and its original pixel pair (x, y) respectively from the odd and even property of M_d. First, if M_d is an even integer, then $x^{'} = l + \frac{M_d}{2} + \lfloor \frac{b+1}{2} \rfloor$, $y^{'} = l - \frac{M_d}{2} - \lfloor \frac{b}{2} \rfloor$. For all values of $b \in \{0, 1\}$, $\lfloor \frac{b+1}{2} \rfloor$ is equivalent to b, and $\lfloor \frac{b}{2} \rfloor$ equals 0. $x^{'}$ is simplified as $l + \frac{M_d}{2} + b$. Similarly, $y^{'} = l + \frac{M_d}{2}$. Since $x = l + \frac{M_d}{2}$, $y = l - \frac{M_d}{2}$ referring to Eq. (2), $x^{'} = x + b$,

Table 1. Relations between $x^{'}$ and x or $y^{'}$ and y

<table>
<tr><td rowspan="2"></td><td colspan="3">$d \geq 0$</td><td colspan="3">$d < 0$</td></tr>
<tr><td colspan="2">odd</td><td>even</td><td>odd</td><td colspan="2">even</td></tr>
<tr><td rowspan="2">$|d| = M_d$</td><td>$b = 0$</td><td>$x^{'} = x$
$y^{'} = y$</td><td rowspan="2">$x^{'} = x + b$
$y^{'} = y$</td><td rowspan="2">$x^{'} = x - b$
$y^{'} = y$</td><td>$b = 0$</td><td>$x^{'} = x$
$y^{'} = y$</td></tr>
<tr><td>$b = 1$</td><td>$x^{'} = x$
$y^{'} = y + 1$</td><td>$b = 1$</td><td>$x^{'} = x$
$y^{'} = y - 1$</td></tr>
<tr><td>$|d| > M_d$</td><td colspan="2">$x^{'} = x$
$y^{'} = y + 1$</td><td>$x^{'} = x + 1$
$y^{'} = y$</td><td>$x^{'} = x - 1$
$y^{'} = y$</td><td colspan="2">$x^{'} = x$
$y^{'} = y - 1$</td></tr>
</table>

$y^{'} = y$. That is, after embedding, a small change for x occurs, while y is kept unaltered. Otherwise, if M_d is odd, then $\lfloor \frac{M_d+b+1}{2} \rfloor = \frac{M_d+1}{2}$, substitute it into Eq. (4), we will find $x^{'} = x, y^{'} = y + b$.

If $d = -M_d$ and M_d is even, then $x^{'} = l + \frac{M_d}{2} + \lfloor \frac{-b+1}{2} \rfloor$. For all values of $b \in \{0, 1\}$, $\lfloor \frac{-b+1}{2} \rfloor = 0$, $\lfloor \frac{-b}{2} \rfloor = -b$. Hence, $x^{'} = x$, $y^{'} = y - b$. Similarly, if M_d is odd, $x^{'} = x - b, y^{'} = y$.

If $|d| > M_d$, then $d^{'} = sign(d) \times (|d| + 1)$. Substitute it into Eq. (3), $x^{'} = l + \lfloor \frac{sign(d)\times(|d|+1)+1}{2} \rfloor, y^{'} = l - \lfloor \frac{sign(d)\times(|d|+1)}{2} \rfloor$. When $d \geq 0$, if d is odd, $x^{'} = x$, $y^{'} = y + 1$ after simplification. If d is even, $x^{'} = x + 1$, $y^{'} = y$. When $d < 0$, if d is odd, then $x^{'} = x - 1$, $y^{'} = y$. Otherwise, $x^{'} = x$, $y^{'} = y - 1$.

We summarize relations between $x^{'}$ and x or $y^{'}$ and y under ten conditions mentioned above into a table illustrated in Table 1.

2.2. **Pre-processing with invariant differences.** Referring to Table 1, you will easily find it is certain that $y^{'}$ is unequal to y under four out of ten conditions. C_1, C_2, C_3 and C_4 are used to respectively stand for these four conditions. C_1 consists of three sub-conditions, i.e., $d = M_d$, M_d is odd and $b = 1$. C_2 is composed of $d = -M_d$, M_d is even and $b = 1$. C_3 consists of $d > M_d$ and d is odd. C_4 is composed of $d < -M_d$ and d is even.

To solve this problem of $y^{'} \neq y$, a pre-processing operation is proposed to ensure $y^{'} = y$ under conditions C_1, C_2, C_3 and C_4. The pre-processing operation is defined as the following equation.

$$\begin{cases} x_p = x + 1 \\ y_p = y + 1 \end{cases} \quad \texttt{if } d \texttt{ satisfies } C_1 \texttt{ or } C_3 \tag{5a}$$

$$\begin{cases} x_p = x - 1 \\ y_p = y - 1 \end{cases} \quad \texttt{if } d \texttt{ satisfies } C_2 \texttt{ or } C_4 \tag{5b}$$

For the other six conditions, $x_p = x$ and $y_p = y$.

After the pre-processing operation, the transform $T(\cdot)$ is applied to the resulting pair (x_p, y_p) to calculate their difference value d_p and average value l_p. Note that there is no change taking place for the difference d_p, i.e., $d_p = d$, after the pre-processing operation. After Eq. (3), a new pair $(x_p^{'}, y_p^{'})$ is obtained via Eq. (2). Table 2 lists relations between $x_p^{'}$ and x or $y_p^{'}$ and y. In comparison with Table 1, you will see $y_p^{'}$ is equivalent to y.

Let z be the right-neighboring pixel of y in a scanning order. After preprocessing, since $y_p^{'}$ is equal to y, $y_p^{'}$ can be used again to be paired with z. Consequently, when every two neighboring pairs have 1-pixel overlap, the embedding rate of around 1 bpp can be achieved.

2.3. **Over/underflow Prevention and Overhead Information.** In order to reversibly retrieve the original image, a problem must be considered, i.e., overflow and underflow problem (for simplicity, overflow is used to represent either overflow or underflow in the rest of this paper). Therefore, a location map is created to record the locations of pixels which will exceed the permitted value range, e.g., $[0, 255]$ for 8-bit grayscale image, after watermark embedding.

2.4. **Watermark embedding.** Let W and H be respectively the width and height of the original image I. I is converted into a one-dimension pixel list respectively according to the orders shown in Fig. 1(a) and Fig. 1(b). For each list, we find the number of absolute differences having the maximum occurrence, and respectively use $M_d^{'}$ and $M_d^{''}$ to denote them. Comparing $M_d^{'}$ with $M_d^{''}$, if $M_d^{'} \geq M_d^{''}$, the order

TABLE 2. Relations between $x_p^{'}$ and x or $y_p^{'}$ and y

<table>
<tr><th rowspan="2"></th><th colspan="3">$d \geq 0$</th><th colspan="3">$d < 0$</th></tr>
<tr><th colspan="2">odd</th><th>even</th><th>odd</th><th colspan="2">even</th></tr>
<tr><td rowspan="2">$|d| = M_d$</td><td>$b = 0$</td><td>$x_p^{'} = x$ $y_p^{'} = y$</td><td rowspan="2">$x_p^{'} = x + b$ $y_p^{'} = y$</td><td rowspan="2">$x_p^{'} = x - b$ $y_p^{'} = y$</td><td>$b = 0$</td><td>$x_p^{'} = x$ $y_p^{'} = y$</td></tr>
<tr><td>$b = 1$</td><td>$x_p^{'} = x + 1$ $y_p^{'} = y$</td><td>$b = 1$</td><td>$x_p^{'} = x - 1$ $y_p^{'} = y$</td></tr>
<tr><td>$|d| > M_d$</td><td colspan="2">$x_p^{'} = x + 1$ $y_p^{'} = y$</td><td>$x_p^{'} = x + 1$ $y_p^{'} = y$</td><td>$x_p^{'} = x - 1$ $y_p^{'} = y$</td><td colspan="2">$x_p^{'} = x - 1$ $y_p^{'} = y$</td></tr>
</table>

referring to Fig. 1(a) is selected, and M_d is set to $M_d^{'}$. Otherwise, the order shown in Fig. 1(b) is selected, and $M_d = M_d^{''}$. I is converted into a one-dimension pixel list I_{D1} according to the selected order.

Two consecutive pixels x, y of I_{D1} is grouped into a pair (x, y), where $0 \leq x \leq 255$, $0 \leq y \leq 255$. A location map L_M is generated and denoted by a bit sequence of size $W \times H$. Referring to Table 2, since some changes are done to x (i.e., $x_p^{'}$ is not equivalent to x), to ensure reversibility, we need to consider whether $x_p^{'}$ falls into the range $[0, 255]$ or not. Take $d = M_d$ and d is an odd integer for example. Two values of b, i.e., $b = 0$ and $b = 1$, respectively correspond to different values of $x_p^{'}$. Under the condition of $b = 0$, $x_p{'}$ equals x, and undoubtedly, $x_p^{'}$ is in the range of $[0, 255]$. Therefore, only when $0 \leq x_p^{'} = x + 1 \leq 255$, pair (x, y) is marked by '1' in L_M, otherwise by '0'. Similarly, for d under the remaining conditions, as long as $0 \leq x_p^{'} \leq 255$, then (x, y) is marked by '1' in L_M, otherwise by '0'. The location map is compressed losslessly by an arithmetic encoder and the resulting bitstream is denoted by $\mathcal{L}$. L_S is the bit length of $\mathcal{L}$.

The proposed method is implemented and tested using MATLAB. Embedding procedure of the whole list I_{D1} is given below if (x, y) is marked by '1' in L_M. Otherwise, (x, y) is kept unchanged.

```
if |d| == M_d
{
    { if d satisfies C_1
        x_p = x + 1; y_p = y + 1; l_p = ⌊(x_p+y_p)/2⌋;
    elseif d satisfies C_2
        x_p = x − 1; y_p = y − 1; l_p = ⌊(x_p+y_p)/2⌋;
    else
        l_p = l;
    }
d' = sign(d) × (|d| + b);
}
elseif (|d| == M_d − 1)&(M_d ≥ 1)
    l_p = l; d' = d;
else
{
    { if d satisfies C_3
        x_p = x + 1; y_p = y + 1; l_p = ⌊(x_p+y_p)/2⌋;
    elseif d satisfies C_4
        x_p = x − 1; y_p = y − 1; l_p = ⌊(x_p+y_p)/2⌋;
    else
        l_p = l;
    }
d' = sign(d) × (|d| + 1);
}
```

After the first L_S pixels of I_{D1} is processed, bitstream L is embedded into the LSBs of $x_p^{'}$ by simple LSB replacement. The L_S LSBs to be replaced are appended to payload. Embedding procedure is applied to the remaining pixels until the whole list I_{D1} is processed. The resulting list holding watermark sequence is converted into a two-dimension watermarked image I_W.

2.5. **Data extraction and image restoration.** The watermarked image I_W is converted into a one-dimension list I_{W1} according to the same order as was done in embedding. For I_{W1}, LSBs of the first L_S watermarked pixels are collected into a bitstream $\mathcal{B}$. $\mathcal{B}$ are decompressed by an arithmetic decoder to retrieve the location map.

Data extraction and pixel restoration is carried out in inverse order as in embedding. M_d is transmitted to the receiving side. With help of M_d, the watermark sequence can be correctly extracted. For each pair $(x_p^{'}, y_p^{'})$, if $x_p^{'}$'s location is associated with '0' in the location map, then it is ignored. Otherwise, d is retrieved according to the following procedure. $(l_p, d^{'})$ is obtained by applying Eq. (1) to $(x_p^{'}, y_p^{'})$.

```
if |d'| == M_d + 1
    { b = 1;  d = sign(d) × (|d'| − 1);
    (x_p, y_p) = T^{-1}(l_p, d);
    if (mod(M_d, 2) == 1) and d ≥ 0
        x = x_p − 1; y = y_p − 1;
    elseif (mod(M_d, 2) == 0) and d < 0
        x = x_p + 1; y = y_p + 1;
    else
        x = x_p; y = y_p;
    }
elseif |d'| == M_d
    b = 0; (x, y) = T^{-1}(l_p, d');
elseif (|d'| == M_d − 1)and(M_d ≥ 1)
    (x, y) = T^{-1}(l_p, d');
else
    { d = sign(d) × (|d'| − 1);
    (x_p, y_p) = T^{-1}(l_p, d);
    { if (mod(M_d, 2) == 1) and d ≥ 0
        x = x_p − 1; y = y_p − 1;
    elseif (mod(M_d, 2) == 0) and d < 0
        x = x_p + 1; y = y_p + 1;
    else
        x = x_p; y = y_p;
    }
}
```

where $mod(\cdot)$ denotes modulo operation.

2.6. **Embedding distortion.** After embedding, let d_x be the difference between $x_p^{'}$ and x, i.e., $d_x = x_p^{'} - x$. If difference d satisfies C_1, then $x_p^{'} = l + 1 + \lfloor \frac{d+1}{2} \rfloor$. Since $d = M_d$, M_d is odd and $b = 1$, compared with Eq. (2), we have $d_x = 1$. Similarly, d_x is also equal to 1 if C_3 is satisfied. In addition, if difference d satisfies C_2 or C_4, $d_x = -1$. For the remaining conditions, $d_x = 0$.

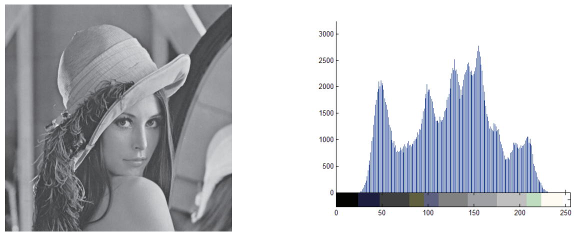
In this paper, the PSNR (Peak Signal to Noise Ratio) is used to evaluate the quality of the watermarked images. PSNR is calculated as the following formula.

$$\mathrm{PSNR} = 10 \cdot \log_{10} \frac{255^2 \times W \times H}{\sum\limits_{i=0}^{W \times H - 1} \left(x_{pi}^{'} - x_i\right)^2} \tag{6}$$

where x_i represents the ith pixel of the pixel list I_{D1}. Correspondingly, x_{pi} is its watermarked value. If the worst happens, i.e., all the pixels in values are increased or decreased by 1 for one single embedding process, the PSNR value obtained is 48.1dB.

3. **Experimental results.** The capacity vs. distortion comparisons among the proposed scheme, Tian's, Coltuc's, Weng's and Alg. D2 are shown in Figs. 2 to 3. For Tian's, Coltuc's and Alg. D2, multiple embedding has been done in order to achieve rates above 0.5 bpp.

As shown in Fig. 2(b), all the pixels are concentrated in the center part, i.e., the range $[26, 247]$, of the histogram. According to the description in Section 2.6, adding 1 to the grayscale value of pixel x means



(a) ‘Lena’ (b) Histogram of ‘Lena’ image

FIGURE 2. Grouping way for a 4 × 4 images block

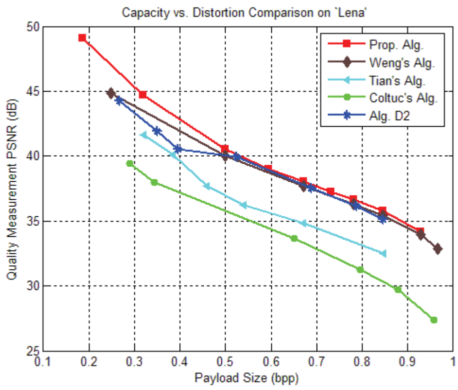


FIGURE 3. Capacity vs. Distortion Comparison on ‘Lena’

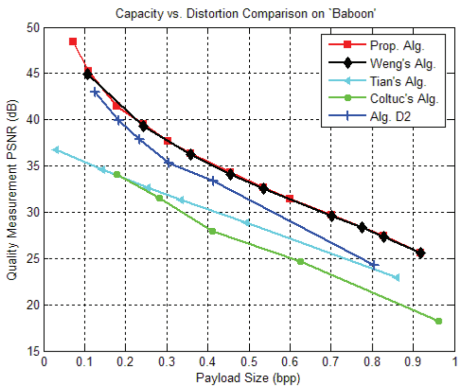
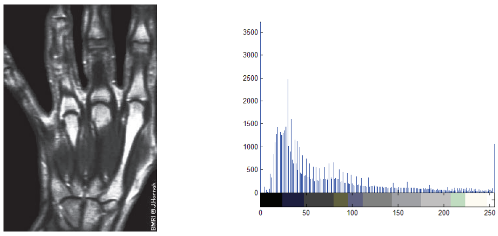


FIGURE 4. Capacity vs. Distortion Comparison on ‘Baboon’

shifting x by 1 towards the right-hand side, while subtracting 1 to the grayscale value of pixel x means shifting x by 1 towards the left-hand side. Since there are no pixels distributed on both high and low end of the histogram, no pixels exceeds the range of [0,255] after embedding. As a result, the location map is compressed into a 40-bit sequence. In the proposed method, the capacity of embedding small payloads at low distortions is largely enhanced. For example, if the order shown in Fig. 1(b) is adopted, the embedding rate can reach 0.1859 bpp when PSNR is 49.1008 dB. As shown in Fig. 3, the proposed method outperforms Tian's, Weng's, Coltuc's and Alg. D2.

‘Baboon’ is a typical image with large areas of complex texture, so the obtained bit-rate is slightly lower at the same PSNR. Fig. 3 shows that the proposed method achieves higher embedding capacity with lower embedding distortion than the others.

‘Hand’ medical image contains a large number of pixels whose values are zero. After Tian's method is applied to ‘Hand’ image, we find that pair $(0, 0)$ is changed to pair $(1, 0)$ or $(0, 0)$, respectively when the watermark bit is 1 or 0. In order to achieve the embedding rate of over 0.5 bpp, multiple embedding is



(a) ‘Hand’ (b) Histogram of ‘Hand’ image

FIGURE 5. Grouping way for a 4 × 4 images block

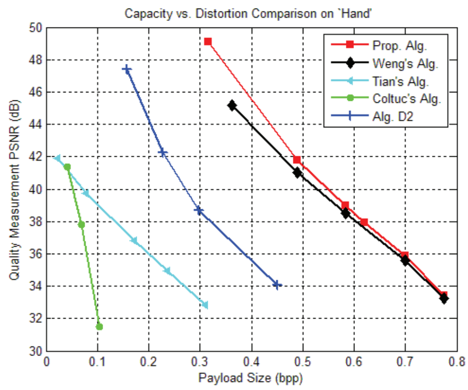


FIGURE 6. Capacity vs. Distortion Comparison on ‘Hand’

needed. Pairs such as $(1, 0)$ can not be used for embedding. Hence, when Tian’s method is used for ‘Hand’ image, the performance is not good. However, such situation does not exists in the proposed method. The proposed method can achieve the rate of close to 1 bpp. In the proposed method, 49.0722dB can be obtained when the embedding rate is 0.3170bpp.

Conclusion: A reversible watermark scheme employing a pre-processing operation before integer Haar wavelet transform (IHWT) is presented in this paper. The application of pre-processing operation helps to make the embedding rate of close to 1 bpp. The experimental results reveal that the proposed method outperforms Tian’s, Coltuc’s and Alg. D2 and Weng’s both in hiding capacity and PSNR value.

Acknowledgment. This work was supported in part by National NSF of China (No. 61201393, No. 61001179).

REFERENCES

[1] J. Tian, Reversible data embedding using a difference expansion, *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890-896, 2003.
[2] D. Coltuc and J. M. Chassery, Very fast watermarking by reversible contrast mapping, *IEEE Signal Processing Letters*, vol. 14, no. 4, pp. 255-258, 2007.
[3] D. M. Thodi and J. J. Rodriguez, and S. Member, Expansion embedding techniques for reversible watermarking, *IEEE Trans. Image Processing*, vol. 16, no. 3, pp. 721-730 ,2007.
[4] W. Shaowei, Z. Yao, N. Rongrong, and J. S. Pan ,Parity-Invariability-Based Reversible Watermarking, *IET Eletronics Letters*, vol. 45, no. 20, pp. 1022-1023, 2009.
[5] C. Ming, C. Zhenyong, Z. Xiao, and X. Zhang, Reversible data hiding using additive prediction-error expansion, *Proc. of the 11th ACM workshop on Multimedia and security*, pp. 19-24 , 2009.