Reversible Watermarking Using Gaussian Weight Prediction and Genetic Algorithm

Chaiyaporn Panyindee and Chuchart Pintavirooj

***Abstract*—This paper represents a high performance reversible watermarking technique which involve adaptable predictor and sorting parameter to suit each image and each payload in order get lowest image distortion. Our proposed method relies on a well-known prediction error (PE) expansion technique. Having small PE values and a harmonious PE sorting parameter will greatly decrease distortion. In order to get adaptable tools, Gaussian weight predictor and expanded variance mean were used as parameters in this work. A genetic algorithm has also been introduced to optimize all parameters and produce the best results possible. Implementation showed a significantly improved result compared to previous work.**

***Index Terms*—Prediction error (PE), Gaussian weight predictor, expanded variance mean, genetic algorithm**

I. Introduction

Reversible watermarking techniques have been developed to obtain the original data with better quality and shorter working process. In earlier works, modulo-arithmetic-based additive spread-spectrum techniques were used by B.Marq [1]. However, using arithmetic modulation caused salt-and-pepper artifacts. Vleeschouver et al. [2] represented a method to decrease artifact severity by using histogram transformation of the circulation interpretation of bijective transformation. Xuan et al. [3] presented an integer wavelet tranform technique for data embedment. In 2003, Jun Tian [4] introduced an effective technique called Difference Expansion (DE) which was a modified High Pass and Low Pass filter and bit-shifting process. This technique can embed data into an image and preserve the original images information. This technique relies on a lossless compression tool to reduce the size of hidden data in order to decrease image distortion like in the work of Calik et al. [7], Fridrich et al. [8] and Ni et al. [9].

In the following year, Thodi and Rodriguez [5] presented a technique called Prediction Error Expansion which improved the DE expansion in Tian's work, but used Prediction Error (PE) instead of the difference of two connected pixels because the error is smaller than the difference of pixels' value. This helps Prediction Error Expansion decrease image degradation significantly. Other researcher had tried to increase the prediction funtion's efficiency like in Shaowei et al. work [10]. However, increasing the prediction function's efficiency did not improve the overall result given by the algorithm as a whole because other components of the algorithm also needed modification. In 2005, Kamstra and Heijiman [11] used a sorting technique to decrease location map size. In 2007, Thodi and Rodrigue [6] introduced another technique called Histogram shifting. This technique was designed to decrease image distortion by shifting the unused pixels instead of expanding them. Moreover, changing a proportion between the numbers "0" and "1" in a location map would produce a more effective compression tool and also decreases the size of the location map. Sachnev et al. [12],[13] introduced the algorithm based on PE in 2009. This work utilizes a variety of techniques in the algorithm. For example, shifting and sorting with double modification testing help decrease location map size, so that it is no longer necessary to use a compression tool. There are other different novel techniques such as using infinity norm rotation [14] to transform data into another domain for data embedding or using fast transformation techniques such as in Caltoc et al. work [15],[16].

There was an attempt to use statistical tools in an algorithm as in work by Li et al. [17] and Kotvicha et.al. [18]. Li et al.'s work in 2011 presented adaptive embedding and pixels' selection technique and they also exploit "Forward, backward variance" and gap between forward and backward pixels to select proper pixel for data embedding. Kotvicha et al. also achieved better results by using EVM to sort data. Currently, Sachnev et al. work has the superior performance [12]. The principle method that this research proposed based on Sachnev et al.'s algorithm. The method includes using Gaussian distribution function to weight local data values because it has more capability in predicting pixels. Then uses Genetic algorithm to optimize parameters in the algorithm in two data dimensions and find an optimized set of parameters for sorting pixel sequence.

This process will diminish Prediction Error value. Together these processes can improve the method's efficient and also decrease image distortion. The rest of this paper will discuss in details as follows: section 2 prediction error expansion; section 3 Gaussian weighted prediction error expansion; section 4 data sorting; section 5 histogram shifting; section 6 double modification testing; section 7 optimization using Genetic Algorithm; section 8 encoding and decoding algorithm; section 9 experimental results and section 10 the conclusion.

Manuscript received October 14, 2012.

C. Panyindee and C. Pintavirooj are with the Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang (KMITL), Chalongkrung road, Ladkrabang, Bangkok 10520, Thailand (phone: +662-329-8346; fax: +662-329-8346; e-mail: chaiyaporn.pan@gmail.com)

II. Prediction Error Expansion

Reversible watermarking using a prediction-error expansion is a technique to conceal data within prediction error. PE uses data from neighboring pixels to predict nearby values. Let $d_{i,j}$ be PE at some position (*i, j*). Let $u_{i,j}$ be an original pixel's value and $u'_{i,j}$ a predicting value, PE can be calculated by $d_{i,j} = u_{i,j} - u'_{i,j}$. Bit shifting (also known as PE expansion), embeds watermark data into the last bit of expanded PE values in order to preserve original image information. This allows encoder to recover watermark extracted image Expanded PE value, *Di,j* is defined by $D_{i,j} = 2d_{i,j} + b$, where *b* is the data to be hidden. The pixel's modified value is $U_{i,j} = u'_{i,j} + D_{i,j}$. Extracting the watermark and recovering the image can be done by $b = U_{i,j} \bmod 2$ and $u_{i,j} = u'_{i,j} + \lfloor D_{i,j}/2 \rfloor$, where $\lfloor \bullet \rfloor$ is the floor function. Sachnev et al. [12] selected local mean to be their prediction value. Their process starts by separating pixels in an image into two sets called "Cross set" and "Dot set". All Cross set pixels are predicted using data from the Dot set. Each pixel is predicted using 4 neighbor pixels from the Dot set using the equation $u'_{i,j} = \lfloor (v_{i,j-1} + v_{i,j+1} + v_{i-1,j} + v_{i+1,j})/4 \rfloor$.

III. Gaussian Weighted Prediction Error Expansion

A Gaussian function is an ideal type of function for determining the weight of a predictor because its shape produced extreme value at the middle and decline when farther away from the center. The shape of Gaussian function depends on only two parameters, the x-direction and the y-direction standard deviation (SD), denote as σ_x and σ_y respectively, so sending only two parameters for an adaptive predictor will help save bits in the header. The other advantage of having only two parameters is that it is not difficult to find a proper weight for a predictor. The Gaussian weight function can be represented by

$$w(x,y) = e^{-\left(x^2/2\sigma_x^2 + y^2/2\sigma_y^2\right)} \tag{1}$$

An adaptive predicting function is

$$u'_{i,j} = \left\lfloor \frac{\sum_{k=1}^{\varphi_p} \sum_{l=-\varphi_p}^{\varphi_p - (2k-1)} \begin{pmatrix} w(l, l+2k-1) v_{i+l, j+l+2k-1} + \\ w(l+2k-1, l) v_{i+l+2k-1, j+l} \end{pmatrix}}{\sum_{k=1}^{\varphi_p} \sum_{l=-\varphi_p}^{\varphi_p - (2k-1)} \begin{pmatrix} w(l, l+2k-1) + \\ w(l+2k-1, l) \end{pmatrix}} \right\rfloor \tag{2}$$

where $(i,j) \in A_1 \times A_2 \cup A_3 \times A_4$ for a Cross set

and $(i,j) \in A_1 \times A_4 \cup A_3 \times A_2$ for a Dot set.

$\lceil \bullet \rceil$ is a ceil function and A_1, A_2, A_3, A_4 are defined by

$A_1 = \{\varphi_p + 2n - 1 \mid n = 1, 2, ..., (N - (N+1) \bmod 2 + 1)/2 - \varphi_p\}$

$A_2 = \{\varphi_p + 2n - 1 \mid n = 1, 2, ..., (M - (M+1) \bmod 2 + 1)/2 - \varphi_p\}$

$A_3 = \{\varphi_p + 2n \mid n = 1, 2, ..., (N - N \bmod 2)/2 - \varphi_p\}$

$A_4 = \{\varphi_p + 2n \mid n = 1, 2, ..., (M - M \bmod 2)/2 - \varphi_p\}$

N is number of row, and *M* is number of column in an image. φ_p is a level of expansion for a predictor.

$\varphi_p \in \{1, 2, ..., \min(\lfloor (N-1)/2 \rfloor, \lfloor (M-1)/2 \rfloor)\}$ From equation (1), the weight function is defined by using σ_x and σ_y in $w(x,y)$ and the level of expansion for a predictor (φ_p). Higher levels of expansion will require more pixels to calculate a predicting value. The number of pixels corresponding to the level of expansion φ_p is equal to $2\varphi_p^2 + 2\varphi_p$.

IV. Data Sorting

Sorting [11] is another tool which helps decreasing an image distortion. The concept is to select the lower PE values before the high PE values. In Sachnev et al. work, they used local variance as a parameter for sorting. The parameter is calculated by using 4 neighbor pixels which probably won't be good enough to get a good sequence of PEs. We should get more pixels to calculate a sorting parameter in order to obtain a better sequence. Kotvicha et al. [19] showed in his work that using local variance mean (EVM), calculated from more numbers of neighbor pixels, could get a better sorting parameter, whose parameter's order is more close to the PEs sequence. EVM can be calculated as follows (3).

$$\bar{\mu}_{i,j}^{\varphi_s} = \frac{\sum_{l=-\lfloor \varphi_s/2 \rfloor}^{\lfloor \varphi_s/2 \rfloor} \mu_{i+l, j+l} + \sum_{k=1}^{\lfloor \varphi_s/2 \rfloor} \sum_{l=-\lceil \varphi_s/2 \rceil}^{\lceil \varphi_s/2 \rceil - 2k} \left(\mu_{i+l, j+l+2k} + \mu_{i+l+2k, j+l} \right)}{2\lceil \varphi_s/2 \rceil^2 + 2\lceil \varphi_s/2 \rceil + 1 - 4(\varphi_s \bmod 2)} \tag{3}$$

where $(i,j) \in B_1 \times B_2 \cup B_3 \times B_4$ for a Cross set,

and $(i,j) \in B_1 \times B_4 \cup B_3 \times B_2$ for a Dot set.

The set B_1, B_2, B_3, B_4 are defined by

$B_1 = \{\lceil \varphi_s/2 \rceil + 2n \mid n = 1, 2, ..., (N - (N+1) \bmod 2 - 1)/2 - \lceil \varphi_s/2 \rceil\}$

$B_2 = \{\lceil \varphi_s/2 \rceil + 2n \mid n = 1, 2, ..., (M - (M+1) \bmod 2 - 1)/2 - \lceil \varphi_s/2 \rceil\}$

$B_3 = \{\lceil \varphi_s/2 \rceil + 2n + 1 \mid n = 1, 2, ..., (N - N \bmod 2 - 2)/2 - \lceil \varphi_s/2 \rceil\}$

$B_4 = \{\lceil \varphi_s/2 \rceil + 2n + 1 \mid n = 1, 2, ..., (M - M \bmod 2 - 2)/2 - \lceil \varphi_s/2 \rceil\}$.

Let $\mu_{i,j}$ be a local variance of the pixel (i,j). It can be calculated by $\mu_{i,j} = \sum_{k=1}^{4} (\Delta v_k - \Delta \bar{v}_k)^2 / 4$, where $\Delta v_1 = | v_{i,j-1} - v_{i-1,j} |$, $\Delta v_2 = | v_{i-1,j} - v_{i,j+1} |$, $\Delta v_3 = | v_{i,j+1} - v_{i+1,j} |$, $\Delta v_4 = | v_{i+1,j} - v_{i,j-1} |$, and $\Delta \bar{v}_k = (\Delta v_1 + \Delta v_2 + \Delta v_3 + \Delta v_4)/4$. φ_s is a level of variance expansion (VE).

$\varphi_s \in \{1, 2..., \min(2\lfloor (N+1)/2 \rfloor - 2, 2\lfloor (M+1)/2 \rfloor - 2)\}$

If this level increases, numbers of pixels used to calculate in (3) will also increase. A number of pixels involved in (3) equals to $2\lceil \varphi_s/2 \rceil^2 + 2\lceil \varphi_s/2 \rceil + 1 - 4(\varphi_s \bmod 2)$.

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. HISTOGRAM SHIFTING

Data embedding using prediction-error expansion may change an original image's pixels depending on PE values. Thodi and Rodriquez [6] developed the histogram shifting method to decrease image distortion by using a threshold value associated with PE. If PE values of some pixels are within the threshold t_p and t_n, PE expansion will be use. In contrast, if the values are not in the threshold, shifting will be used instead. Shifting produces an adjusted value which lies outside the range of expanded values. An adjusted value can be evaluated using:

$$D_{i,j} = \begin{cases} 2d_{i,j} + b & if\ d_{i,j} \in [t_n, t_p] \\ d_{i,j} + t_p + 1 & if\ d_{i,j} > t_p \geq 0 \\ d_{i,j} + t_n & if\ d_{i,j} < t_n < 0 \end{cases}$$

Expanded PE values range between $2t_n$ and $2t_p + 1$, while the shifted PE are greater than $2t_p + 1$ or less than $2t_n$. The original PE may be recovered with the equation:

$$d_{i,j} = \begin{cases} \lfloor D_{i,j}/2 \rfloor & if\ D_{i,j} \in [2t_n, 2t_p + 1] \\ D_{i,j} - t_p - 1 & if\ D_{i,j} > 2t_p + 1 \\ D_{i,j} - t_n & if\ D_{i,j} < 2t_n \end{cases}.$$

VI. DOUBLE MODIFICATION TESTING

Double Modification testing is the method presented by Sachnev et al. [12]. This technique is used to check if the modified pixel's value will cause underflow or overflow problem by double expanding or shifting. When a decoder calculate $D_{i,j}$, if $D_{i,j}$ is expanded or shifted and the result still remains in the range $[0, 2^B - 1]$ (for B bits images), then the pixel doesn't have an underflow-overflow problem and the location map is not needed. If the result causes an underflow-overflow problem, then a location map will be developed. An encoder can check these points by expanding or shifting the pixels which have already been expended or shifted once.

Double Modification testing separates pixels into 7 sets.

- EE is a set of pixels which values are expandable twice without an underflow – overflow problem.
- ES is a set of pixels which values are expandable in the first time and shiftable for the second time without an underflow – overflow problem.
- SS is a set of pixels which values are shiftable twice without an underflow – overflow problem.
- E is a set of pixels which values are expandable once and the second modification causes an underflow – overflow problem.
- NE is a set of pixels that their values are always unexpandable.
- S is a set of pixels that their values are shiftable at the first shift but caused underflow or overflow at the second shift.
- NS is a set of pixels that their values cannot be shifted at always.

We can see that each set can be identified both by an encoder or a decoder. Pixels in EE, ES and SS can be checked by a decoder because their values don't have underflow – overflow problem that the set E, NE, S and NS do. To separate these pixels into their correct sets, we need to use a location map .Pixels that have a code "0" in a location map indicates "modified", (expanded or shifted) and "1" means "unmodified".

VII. OPTIMIZATIONS USING THE GENETIC ALGORITHM

Genetic algorithm (GA) is a well-known search heuristic optimization technique. GA is a method that mimics natural evolution. The genetic algorithm takes parameters called genes and chromosomes. In our application, "genes" are the parameters σ_x, σ_y and φ_s in binary form that contains bits information, the longer genes the better exchange potential. Our "chromosome" is the binary string composed of the three genes. Binary representation can change to be decimal by using the equation:

$$decimal = lb + (ub - lb)(\sum_{i=1}^{m} 2^{i-1} b_{i-1}) / (2^m - 1) \qquad (4)$$

$u_b, l_b \in \Re$ when u_b is a parameter's upper bound. l_b is a parameter's lower bound. m is a number of bits using for each gene and b_{i-1} is a binary value of i^{th} bits. After generate n different binary string of chromosomes, they will be put into an objective function. An objective function (sometime called fitness function) for GA is defined to be the mean square error (MSE) as follows:

$$MSE = \sum_{i=1}^{N} \sum_{j=1}^{M} \left| u_{i,j} - U_{i,j} \right|^2 / (N \cdot M) \qquad (5)$$

MSE is an average value for the square of the difference between an original image and an embedded image. We need to find values for σ_x, σ_y and φ_s which give lowest MSE value.

$b_{m_{\sigma_x}-1} b_{m_{\sigma_x}-2} \ldots b_{0_{\sigma_x}}$	$b_{m_{\sigma_y}-1} b_{m_{\sigma_y}-2} \ldots b_{0_{\sigma_y}}$	$b_{m_{\varphi_s}-1} b_{m_{\varphi_s}-2} \ldots b_{0_{\varphi_s}}$
gene1= σ_x	gene2= σ_y	gene3= φ_s

We would have to control the boundary of each parameter. Consider σ_x and σ_y, defined as m bits, they can represent up to 2^m different values. In order to find an upper bound and a lower bound of the parameters, let w_R^x, w_R^y be the Gaussian weight, which located R unit away from the center along x-direction and y-direction respectively, so $w_R^x = e^{-(R^2/\sigma_x^2)}$ and $w_R^y = e^{-(R^2/2\sigma_y^2)}$. Suppose $w_{R=1}^x = \alpha \cdot w_{R=2}^x$, the Gaussian weight of a point which is 1 unit from the center is α times bigger than one that is 2 units from the center along x-direction. Suppose further that $w_{R=1}^x = \beta \cdot w_{R=1}^y$, the weight located 1 unit away from the center along x-direction, is β

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

times weight along the y-direction. Thus $\sigma_x = \sqrt{3/(2\ln\alpha)}$ and $\sigma_y = \sqrt{3/(2\ln\alpha + 6\ln\beta)}$. If α is very big, the curve will decay quickly. If $\beta >> 1$, the weight along y-direction is relatively small compared to the other direction, so the curve expands mainly in x-direction. Suppose defining $\alpha = 10^4$ and $\beta = 10^4$, we will get $\sigma_x = 0.4$ and $\sigma_y = 0.2$. The weight locates at the radius 1 on x-direction is 10^4 times of the weight at the radius 2 on the same direction, and it is 10^4 times of the weight at the same radius on y-direction. This is an example of the Gaussian curve expands only in the x-direction. Its shape decays very quickly far from the center because the weight at radius 2 is 10^4 times smaller than the weight at radius 1. In this case we will have the smallest possible neighborhood region used in the predicting function. For this reason, we set 0.2 to be a lower bound of SD values. If the SD is 5, the weight at radius 1 will be 2 times of the one at radius 6 which give a wide enough region used for the predicting function, so we set 5 as a parameter's upper bound. Since SD is bounded, the domain of the parameter is limited. This helps the GA tool perform faster. For $\alpha = 10^4$ and $\beta = 1$, $\sigma_x = \sigma_y = 0.4$, meaning that all weight located at radius 1 have the same value and the weight at radius 2 is small enough so that it can be ignored. This case can be considered to be the same as Sachnev et al. predicting function. The next step is updating all chromosomes using 3 basic steps of genetic operation, selection, crossover and mutation [19].

VIII. Encoding and Decoding

Encoding algorithm

1. Set $\tau = 0$ and set φ_p
2. Start GA by defining initial binary values for *n* chromosomes σ_x, σ_y and φ_s.
3. Transform all binary from parameters to decimal numbers using (4).
4. Define positions into a Cross set and a Dot set.
5. For each set of parameters from each chromosome, apply it to:
5.1 Calculate Gaussian weight using (1)
5.2 For each position in the cross set compute:
5.2.1 Prediction value using (2) and its PE.
5.2.2 The sorting parameter, EVM, using (3).
5.3. Preserve the first *h* pixels for header and sort the rest of pixels in the Cross set in ascending order using EVM.
5.4. Check a space for embedding algorithm:
5.4.1 Set up *k* as an index represents order in the sorted sequence and start from 1.
5.4.2 Increase *k* one by one and count number of set *EB* and *LC*.
5.4.3 Find the order, *k*, that according to [13] satisfied.
If according to [13] is satisfied and $k \le n$ *(cross set)* - *h* for some set of parameters, go to the next step, or if GA is terminated, increase τ and go to step 2.
5.5. Collect the first *h* preserved pixels' LSB and replace them by the header.
5.6. Collect location map and put both LSB from the previous step and location map into payload.
5.7. Modify the first *k* pixels and embed payload respectively and the modified pixel's value $U_{i,j} = u'_{i,j} + D_{i,j}$.
5.8. Apply the steps 5.2-5.7 to the dot set.
5.9. Evaluate MSE using (5).
6. The lowest MSE obtained in that generation is the best fitness value. If the lower MSE cannot be found or a number of iterations reach the maximum generation, GA will be terminated. Then go to next step or update chromosomes using GA and go back to step 3.
7. Compare MSE obtained from the 5 first embeddable thresholds. The one which give the lowest MSE will be selected for data embedding and exit.

Decoding algorithm

1. Separate all positions in an image into a Cross set and a Dot set.
2. Extract the header from the LSB of the first *h* pixels of Dot set and convert all binary parameters to decimal using (7). The parameters $\sigma_x, \sigma_y, \varphi_s, t_p, t_n$ and payload size are obtained.
3. Calculate the EVM of the pixels starts from the $(h+1)^{th}$ position in dot set and sort them in ascending order.
4. Calculate the prediction value and modified PE values of all pixels in the sequence using (2).
5. Define the set corresponding to each pixels in the sorted sequence using the definition 2 and the thresholds from the header.
6. Extract the payload from the pixels which belong to the sets *EE* and *ES* sequentially.
7. Identify the watermark, header's LSB and location map from payload.
8. Replace the header's LSBs back to the first *h* pixels.
9. Recover all modified pixels in sorting sequence using according to [13] and the location map extracted from payload.
10. Repeat step 2-9 to Cross set.

IX. Experimental Results

In our experiment, we tested the proposed method using several 512x512 standard test images. The method effectiveness is measured by using PSNR. It is evaluated $PSNR = 10\log_{10}((2^B - 1)^2 / MSE)$ where MSE is the mean square error defined by equation (5). The higher quality image will have a higher PSNR value. φ_p is defined to be 6 in our work. The implementing results show that our proposed method gives better results than the previous work. Figure1 shows the implementing results of the proposed method compared to Sachnev et al. and Li et al. method. The tested payloads start from 10,000 bits and increasing by 10,000 bits. The PSNR of the proposed method is higher than both comparing method in every test image and in almost every size of payload. This success is largely due to the fact that the coefficients which are used in our predicting function can be adapted to best suit each image while the predictor of Sachnev et al. method is static. Sachnev et al. method doesn't consider how far a local relation expands. The graph of Li et al. method is lower than the proposed one except at the tail of the graph in some images.

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

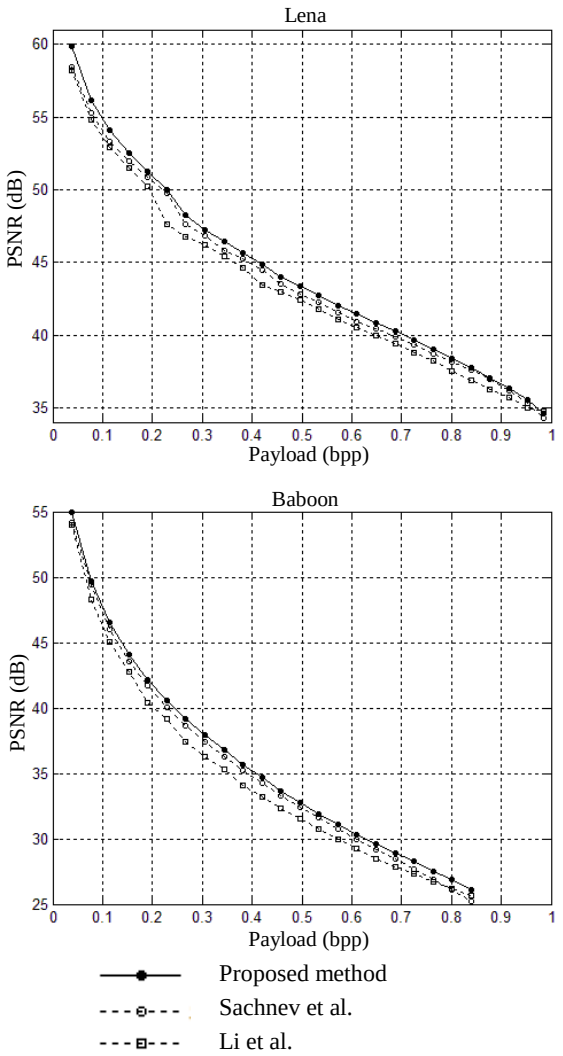

Figure 1: Payload (bpp) vs Distortion (PSNR) graphs of the purposed method compare to Sachnev et al. and Li et al. apply to two test images, Lena and Baboon respectively.

X. Conclusion

Our proposed reversible watermarking method produces considerably less distorted images than Sachnev et al. method. The technique we use combines adaptive models for both the predictor and sorting parameter as well as optimization techniques to increase predictor efficiency and decreased prediction error. We used a Gaussian weight function for the predictor because it can be modified for specific parameter values by changing only two variables. The prediction error value cannot be used to sort data because hiding data causes sorting errors when the decoder attempts to reinterpret the data. Thus, instead of relying on the prediction error value, we use an optimization tool to obtain a different sorting parameter. This adaptive sorting tool works well with the predictor value which improves efficiency. Experimental data shows that our genetic algorithm produces significant improvement in image quality over previous methods.

Acknowledgment

This work was supported by the Office of the Higher Education Commission, Ministry of Education, Thailand, under the research project title of "Higher Education Research Promotion".

References

[1] B. Macq, "Lossless multiresolution transform for image authenticating watermarking," *EUSIPCO Proceeding. Tampere:EUSIPCO.*, pp.533-536, 2000.

[2] C. De Vleeschouver, J. E. Delaigle, and B. Marq, "Circular interpretation of bijective transformation in lossless watermarking for media asset management," *IEEE Trans Multimedia*, vol.5, no.1, pp.97-105, Mar. 2003.

[3] G. Xuan, J. Zhu, J. Chen, Y. Q. Shi, Z. Ni and W. Su, "Distortionless data hiding based on integer wavelet transform," *IEE Electron. Lett.*, vol.38, no.25, pp.1646-1648, Dec. 2002.

[4] J. Tian, "Reversible watermarking using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol.13, no.8, pp.890-896, Aug. 2003.

[5] D. M. Thodi and J. J. Rodriquez, "Prediction-error based reversible watermarking," *Proceeding of IEEE Conf. Image Processing*, pp.1549-1552, Oct. 2004.

[6] D. M. Thodi and J. J. Rodriquez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans Journal Image Process.*, vol.16, no.3, pp.721-730, Mar. 2007.

[7] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber,"Reversible data hiding," *IEEE International Conference on Image Processing*, vol. 2, pp. II/157–II/160, 2002

[8] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding-New paradigm in digital watermarking," *EURASIP Journal Appl.Signal Process.*, vol.2, pp.185-196, 2003.

[9] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, Q. Sun, and X. Lin,"Robust lossless image data hiding," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '04)*,vol. 3, pp. 2199–2202, Taipei, Taiwan, 2004.

[10] W. Shaowei, Z. Yao, N. RongRong, and P. Jeng-Shyang, "Lossless data hiding based on prediction-error adjustment," *Sci China Ser F-Inf Sci*, vol.52, no.2, pp.269-275, Feb. 2009.

[11] L. H. J. Kamstra and A. M. Heijimans, "Reversible data embedding into images using wavelet and sorting," *IEEE Tran. Image Process.*, vol.14, no.12, pp.2082-2090, Dec. 2005

[12] V. Sachnev, H. J. Kim, J.Nam, S.Suresh and Y.Q. Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Transaction on Circuit System and Video Technology*, vol.19, no.7, pp.989-999, July 2009.

[13] V. Sachnev, H. Kim, S. Suresh, and Y.Q. Shi, "Reversible watermarking algorithm with distortion compensation,"*EURASIP Journal on Advances in Signal Processing.*, volume 2010.

[14] L.Yang and P. Hao, "Infinity Norm Rotation transforms," *IEEE Transaction on Signal Processing.*, vol.57,no.7, pp.2594-2603, July 2009.

[15] D. Coltuc and J. M. Chassery, "Very fast watermarking by reversible contrast mapping," *IEEE Signal Process. Lett.*, vol. 14, no. 4, pp.255–258, Apr. 2007.

[16] D. Coltuc, "Improved embedding for prediction based reversible watermarking,"*IEEE Transaction on Information Forensics and Security.*, vol.6,no.3, pp.873-882, Sept 2011.

[17] X. Li, B. Yang and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Transaction on Image Processing.*, vol.20,no.12, pp.3524-3533, Dec. 2011.

[18] A. Kotvicha, P. Sanguansat and M.L.K. Kasemsa, "Expand variance mean sorting for reversible watermarking," *Proceeding of Signal and Information Processing(IPCSIT)*, July 2012.

[19] A. Chipperfield, P. Fleming, H.Pohlheim and C. Fonseca, *Genetic Algorithm Toolbox for Use with MATLAB User's Guide Version1.2*, Department of Automatic Control and System Engineering, University of Sheffield, England.

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)