PROGRESSIVE REVERSIBLE DATA HIDING BY SYMMETRICAL HISTOGRAM EXPANSION WITH PIECEWISE-LINEAR HAAR TRANSFORM *

Lei Yang[a] Pengwei Hao[a,b] Chao Zhang[a]

[a] Center for Infomation Science, Peking University, Beijing 100871, China

[b] Department of Computer Science, Queen Mary University of London, London E1 4NS, UK

E-mail: {yanglei, phao, chzhang}@cis.pku.edu.cn, phao@dcs.qmul.ac.uk

ABSTRACT

In this paper, we present a progressive reversible data hiding technique by symmetrical histogram expansion in the transform domain of Piecewise-Linear Haar (PLHaar). Data are embedded into the PLHaar coefficients of images progressively from the pivotal bin of a histogram of PLHaar coefficients to both sides of the pivot symmetrically. With PLHaar, no overflow or underflow occurs to the pixel values, and our data hiding method achieves the highest embedding capacity with PSNR around 50dB compared to the previous methods in the literature. The method can also be applied to artificial images with an exactly flat histogram, which is impossible for a method to hide data in the spatial domain. The progressiveness of the proposed method also enables a rough but automatic capacity-PSNR control. The effectiveness of our method is demonstrated with a number of experiments.

Index Terms— Reversible data hiding, symmetrical histogram expansion, PLHaar, capacity-PSNR control, progressive data hiding

1. INTRODUCTION

Reversible data hiding [1-10], often referred to as reversible watermarking, was proposed as a promising technique for sensitive image (such as medical and remote sensed images) authentication, and it has drawn much attention in the recent years. Such an embedding algorithm allows extraction of intact hidden data from the watermarked digital carriers and lossless recovery of the original images, if no modification has been made to the watermarked digital carriers.

The earliest reference of reversible data hiding appeared as an authentication method in a patent proposed by Honsinger *et al* [1], which utilizes modulo arithmetic but suffers from salt-and-pepper visual artifact. Afterwards, a number of reversible data hiding techniques have been developed, and the methods can be classified into two main categories by different data embedding domains. One category applies data embedding in the spatial domain [1-4], with relatively low capacity. The other category hides data into the coefficients in the transform domains [5-9], such as the integer DCT and the integer wavelet transform domains. The second category of methods are all facing the same problem: how to choose appropriate embedding locations to avoid pixel value overflow and underflow, and all the solutions have to pay the overhead cost to record all the embedding locations, which greatly reduces the embedding capacity.

In this paper, we try data hiding in the transform domain of the Piecewise-Linear Haar transform (PLHaar) [11]. PLHaar is an integer Haar-like transform, and was originally developed by J. Senecal for lossy and lossless image compression. It maps an n-bit integer to another n-bit integer, which results in no dynamic range expansion. For a 2D point of a pair of pixel values in the range of $[0,255]\times[0,255]$, PLHaar transforms the point into a new point in the same range of $[0,255]\times[0,255]$ and just with 45^o rotation in the sense of the L_∞ norm instead of the L_2 norm. Thus, all the PLHaar coefficients of images with pixel values in [0,255] are still in [0,255]. PLHaar guarantees no overflow or underflow of pixel values, so every coefficient in the PLHaar domain is available for embedding data, as long as the coefficients remain in the same range after embedding.

We embed watermark bits into the coefficients in the PLHaar domain, i.e., we only modify bits of the coefficients, so watermarked coefficients are still in the same range as the original, [0,255]. Then with the inverse PLHaar transform, the watermarked images we obtain are surely without pixel value overflow or underflow. Consequently, there is no need to carefully select proper embedding locations for hiding data. Without such a process, the PLHaar allows higher hiding capacity and simpler hiding algorithm. At the same time, all the data in our algorithm with PLHaar can be represented with 8-bit numbers, which makes cheap and fast hardware implementation possible for 8-bit fixed-point arithmetic.

Histograms in the PLHaar domain have the good properties of decorrelated coefficients and peaked distribution. So the artificial images with exactly flat histograms, which have zero embedding capacity in the spatial domain, can be embedded into a lot of bits with our algorithm. In Section 2, we propose a symmetrical

* This work was supported by NKBRPC of China (2004CB318005).

histogram expansion algorithm in the PLHaar transform domain, which outperforms the histogram expansion methods in the spatial domain, with high embedding capacity and progressiveness. We choose a pivotal bin as the symmetrical axis in a histogram, and shift the other histogram bins to both sides in order to evacuate some bins for data embedding. Then, we insert bins of pixel values to both the left and the right sides of the pivot in the histogram, progressively according to the length of the hidden data. The only overhead of our algorithm is the location of the pivotal bin and the length of the watermark bits, which is hidden in the least significant bits (LSB) of the coefficients in the first row. Progressiveness of our algorithm enables a rough but automatic control of capacity-PSNR. By pushing pixel value bins in histograms further to both sides for more number of empty bins around the candidate pixel value bins for embedding, we can embed the watermark bits into not only the least significant bits, but also the other less significant bits.

We will summarize our data embedding and extraction algorithms in Section 3. Experimental results on seven $512 \times 512 \times 8$ gray-scale images are presented in Section 4, and conclusions are drawn in Section 5.

2. SYMMETRICAL HISTOGRAM EXPANSION

To make our description clearer, we introduce a concept of *n-Buddy* for *m*-bit numbers, with which *n* bits of a coefficient can be changed in reversible histogram modification.

Definition 1: *n-Buddy*

An n-Buddy of an m-bit number is an m-bit number, whose upper $m-n$ bits are the same, but the lower n bits have at least one bit different.

Take 8-bit numbers as examples: 2 and 3 are 1-buddies of each other, and 4, 6, 7 are all 2-buddies of 5 under Definition 1. It is obvious that if a number is modified in the lower *n* bits, it must be one of its *n*-buddies or itself. Therefore, after the lower *n* bits of the coefficients in a histogram bin are modified, the pixel value in the bin expands into the empty neighbor *n*-buddy bins. In the recovery process, these neighbor *n*-buddy bins are all mapped into the original single bin. Our lower *n*-bit modification scheme is illustrated in Fig. 1.

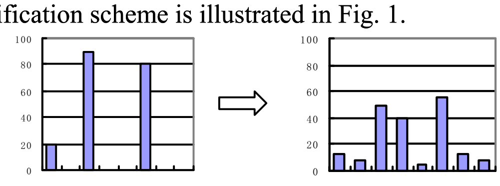

Fig.1. Histogram modification for embedding data

In Fig. 1, the LSB of pixel value 0 and 2, the LSB and the second LSB of 5 are modified for data hiding, and then the left histogram is transformed into the right. After modification (data hiding), as in the right histogram of Fig. 1, pixel values 0 and 1 as actually come from pixel value 0, pixel values 2 and 3 are from 2, and pixel values 4, 5, 6 and 7 are from 5. That means if we want to embed data into the lower *n* bits of a bin number in a histogram, we need shift its *n*-buddy bins aside to make room for the reversible histogram modification. If there are *k* coefficients, we can embed *kn* bits into the coefficients by modifying the lower *n* bits. The embedding process can be formulated as:

$$C_w = \lfloor C/2^n \rfloor \cdot 2^n + W \tag{1}$$

Fig. 2 illustrates our symmetrical histogram expansion scheme. Firstly, we choose bin 6 as the pivotal bin for the first embedding and shift bin 7 to the right as bin 8 to evacuate bin 7 for the embedding. Then we shift bin 4 to the left to embed data into bin 5. Finally, we embed data into bin 8. That is for watermark embedding in a zigzag scanning order. For hidden data extraction, we convert pixel values 6 and 7 into 6, 5 and 4 into 5, 3 into 4, 8 and 9 into 7 so as to recover the original coefficient values in the same order as in embedding.

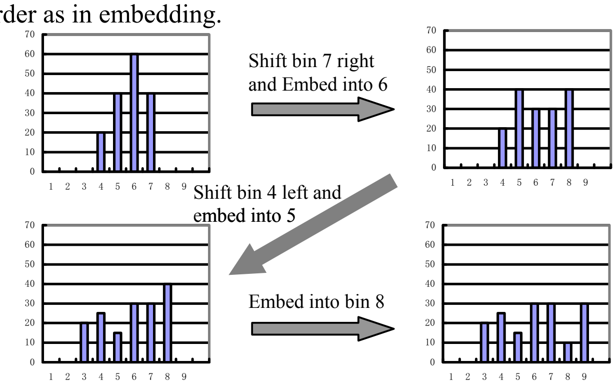

Fig.2 Symmetrical histogram expansion for data embedding

Some advantages of our symmetrical histogram expansion scheme can be summarized as follows.

(1) Due to symmetrical histogram expansion, the histograms are equalized, resulting in enhanced images with better visual quality (see Fig.3).
(2) The algorithm is simple, and only the location of the pivotal bin and the length of the hidden data need be recorded for lossless extraction.
(3) Histogram shifts from the pivotal bin to both sides for data hiding, so the data hiding process is progressive and there is no need to evacuate all the embedding bins beforehand.
(4) Progressiveness allows a rough automatic control of capacity-PSNR, since fewer watermark bits need less histogram shifting.

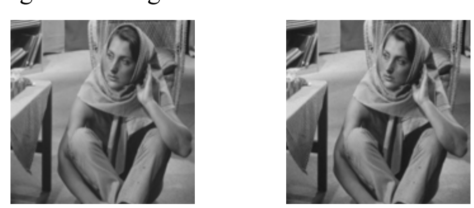

Fig.3 The original image (Left) and the watermarked image with 16298 byte hidden data and 34.04dB PSNR (Right)

3. DATA HIDING AND EXTRACTION

Based on the PLHaar transform and the symmetrical histogram expansion, our progressive algorithm is summarized as follows.

Data embedding:

1. Transform the original image using PLHaar.
2. Attach the lower bits of the coefficients in the first row to the watermark bits.
3. Choose a middle bin in the PLHaar coefficient histogram as the pivot, and record the location and the length of the data to hide into some lower bits of the coefficients in the first row.
4. ***While*** there are watermark bits left to be embedded ***do***
   Expand the histogram symmetrically;
   Modify the lower n bits of the coefficients.
   End while
5. Apply the inverse PLHaar transform and output the watermarked image.

Data extraction and image recovery:

1. Transform the watermarked image using PLHaar.
2. Extract the pivot value and the length of the hidden data from the coefficients in the first row.
3. ***While*** there are watermark bits left to be extracted ***do***
   Extract the lower n bits of the coefficients;
   Shrink the expanded histogram back.
   End while
4. Recover the coefficients in the first row.
5. Apply the inverse PLHaar transform to recover the original image.

About the above algorithm, the following three points help to make it perform obviously better:

(1) The scanning order for data extraction is the same as that of data embedding from the pivotal bin to both sides.

(2) If there are no empty bins in the coefficient histogram, use the coefficients of small bins as the shifting locations, and the overhead can be embedded into any pre-designated coefficients.

(3) The choice of the pivotal bin should be determined by the applications and the characteristics of a histogram. For high embedding capacity requirements, the pivot can be the highest bin in the histogram. For high image quality requirements, the pivot can be chosen for fewer bin shifts.

Fig. 4 Grayscale images used for experiments

4. EXPERIMENTAL RESULTS

The performance of our proposed reversible data hiding algorithm has been demonstrated by a number of experiments on six 512×512 grayscale images shown in Fig.4 and an artificial image with exactly flat histogram.

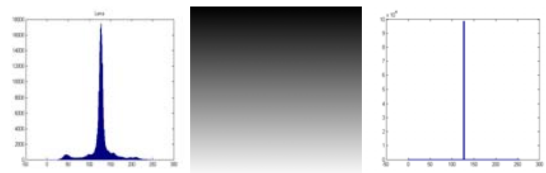

Fig. 5 (Left) Histogram of coefficients of Lena image
(Mid) Artificial image with exactly flat histogram
(Right) Histogram of the coefficients of the artificial image

Our experiments show that the proposed algorithm of symmetrical expansion of the PLHaar coefficient histograms works well for both the natural images and the artificial image. The test results of hidden data size and the corresponding PSNR are listed in Table 1.

Table 1 Capacity (bytes) and PSNR (dB) for the test images

Barbara	50.69	46.14	44.49	42.65	40.10	37.93	34.04
	1716	4961	6652	7871	10482	12662	16298
Lena	49.77	44.59	41.35	37.12	35.56	34.20	32.46
	4178	8131	11568	16662	18403	19731	21238
Boat	49.75	44.55	42.42	39.51	37.48	35.78	33.96
	4515	8576	9971	11478	16297	17604	18569
Jet	50.65	44.98	41.62	39.11	37.13	35.44	33.98
	5927	9789	13591	15998	17635	18975	19832
Baboon	51.28	45.31	41.82	39.34	37.41	35.84	33.35
	1551	3004	4435	5837	7202	8476	10884
Gold	49.94	44.73	41.44	39.07	37.20	35.64	33.13
	3105	6045	8746	11151	13243	15033	17871
Artificial	30.01	26.97	---	---	---	---	---
	1461	2922	---	---	---	---	---

In Table 1, the numbers in the upper row for each image are PSNR of the watermarked images relative to the original image, and those in the lower row are the corresponding capacity for data hiding. *Artificial* represents the man-made artificial image shown in Fig.5. Table 1 shows that our algorithm achieves very high embedding capacity at PSNR around 50dB. Actually, higher capacity can still be exploited with our algorithm.

Table 2 Capacity comparison (in bytes) between ours and other reversible watermarking methods [4] at high PSNR(dB)

Method	Lena (512*512*8)		Baboon (512*512*8)	
	Capacity	PSNR	Capacity	PSNR
Honsinger et al [1]	<128	---	<128	---
Macq et al [10]	<256	---	<256	---
Fridrich et al [2]	128	---	128	---
Goljan et al [3]	3014	39	364	39
Xuan et al [7]	3277	48.2	3277	42.04
Celik et al [9]	1537	46.9	746	42.1
Ni et al [4]	683	48.2	678	48.2
Ours	4178	49.77	3738	42.66

We have also compared our method with other methods [4], and the information is listed in Table 2. It shows that our method achieves the highest data hiding capacity.

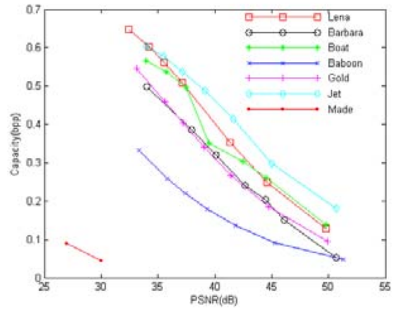

Fig.6 Capacity-PSNR performance on the test images

Capacity-PSNR performance of our algorithm is shown in Fig.6. It gives the information about the progressiveness of our method. Data are embedded into an image progressively from the lower-right end upwards to the upper-left of the curves. Fig.7-9 show the performance comparison between our algorithm and Celik's, which performs well in the literature of reversible data hiding. The figures show that our algorithm achieves higher capacity at the same PSNR for all the test images, or interpreted as higher PSNR at the same capacity.

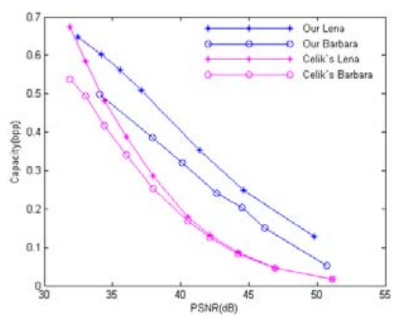

Fig.7 Comparison with Celik's algorithm on Lena and Barbara

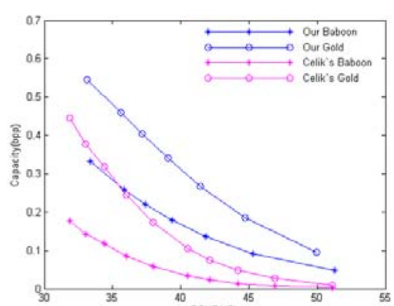

Fig.8 Comparison with Celik's algorithm on Baboon and Gold

5. CONCLUSIONS

PLHaar does not lead to overflow or underflow, so our proposed algorithm for reversible data hiding in PLHaar transform domain is simple and efficient. Our new algorithm uses symmetrical histogram expansion in PLHaar transform domain, and it performs well with all the test images and outperforms all other methods for high image quality applications. Anyway, there should be some other transforms that keep the dynamic range and perform better. To search for such transforms is an interesting subject of research in the future.

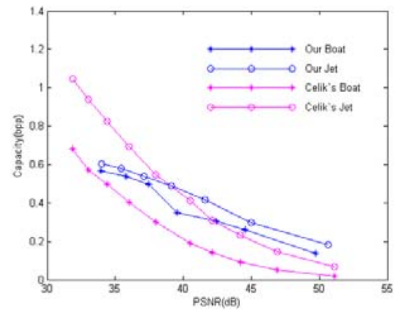

Fig.9 Comparison with Celik's algorithm on Boat and Jet

6. REFERENCES

[1] C. Honsinger, P. Jones, M. Rabbani and J.Stoffel, "Lossless recovery of an original image containing embedded data," *US Patent, No.US6278791*, 1999.

[2] J. Fridrich, M. Goljan and R. Du, "Invertible authentication," *Proc. SPIE, Security and watermarking of multimedia contents*, San Jose, California, pp. 197-208, January 23-24, 2001.

[3] M. Goljan, J. Fridrich and R. Du, "Distortion-free data embedding," *Proc. 4th Information Hiding Workshop*, Pittsburgh, pp. 27-41, Apr. 2001.

[4] Z. Ni, Y. Shi, N.Ansari and W. Su, "Reversible data hiding," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol.16, No.3, March 2006.

[5] G. Xuan, J. Zhu, J. Chen, Y. Shi, Z. Ni and W. Su, "Distortionless data hiding based on integer wavelet transform," *IEE Electronics Letters*, Vol.38, No.25, pp. 1646-1648, Dec. 2002.

[6] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. on Circuits and Systems for video Technology,* Vol.13, No.8, August 2003.

[7] G. Xuan, J. Zhu, J. Chen, Y. Shi, Z. Ni and W. Su, "Reversible data hiding on wavelet spread spectrum," *IEEE Proceedings of Multimedia Signal Processing Workshop*, Siena, Italy, Sept. 2004.

[8] B. Yang, M. Schmucker, W. Funk, C. Busch and S. Sun, "Integer DCT-based reversible watermarking for images using companding technique," *Proceeding of SPIE, Security and watermarking of Multimedia Content*, San Jose (USA), 2004.

[9] M. Celik, G. Sharma, A. Tekalp and E. Saber, "Lossless generalized-LSB data embedding," *IEEE trans. on image processing*, Vol.14, No.2, February 2005.

[10] B. Macq and F. Deweyand, "Trusted headers for medical images," presented at the *DFG VIII-D II Watermarking Workshop*, Erlangen, Germany, Oct. 1999.

[11] J.Senecal, M. Duchaineau and K. Joy, "An improved n-bit to n-bit reversible Haar-like tramsforms," *Pacific Conference on Computer Graphics and Applications*, August 16-18, 2004.