SOURCE-CHANNEL CODING APPROACH TO GENERATE TAMPER-PROOF IMAGES

Saeed Sarreshtedari, Mohammad Ali Akhaee

School of Electrical and Computer Eng., College of Eng., University of Tehran, Tehran, Iran
s.sarreshtedari, akhaee@ut.ac.ir

ABSTRACT

Watemarking the original images to protect them against tampering has recently attracted an overgrowing interest. Recent works in this field offer algorithms that not only localize the tampering, but also recover the original content in the lost area. In this paper, we show that this self-restoration problem can be modeled as a source-channel coding problem. The original image is compressed using an efficient source encoder. The output is then channel coded to be capable of tolerating a certain rate of tampering. At the receiver, decoder reveals the encoder output bit stream if the tampering is below the certain limit. Decoder exploits the location of the erased blocks at the decoding, which are known thanks to the embedded check bits. The output of the source decoder is then used to replace the content of the tampered area. We show that approaching the self-restoration problem from this general viewpoint, the performance is significantly improved comparing to the state of the art schemes, in terms of the quality of watermarked image, quality of the restored content, and tolerable tampering rate.

Index Terms— Image tampering protection, Self-recovery, Watermarking

1. INTRODUCTION

Since the birth of digital data hiding, image authentication has been one of its most widespread applications. During the first years, aside from a few numbers of works, this application was limited to verifying the integrity of the image or locating its tampered regions. Nowadays, these watermarks are designed in sophisticated ways to not only localize the image tampering, but also recover the lost content in those areas. In tampering protection and self-recovery schemes, the goal is to embed a representation of the original image into itself in a way to efficiently compromise between three design parameters; the quality of the watermarked image, the quality of the content recovery in tampered areas, and the tolerable tampering rate (TTR). Embedding a larger watermark, results in worse watermarked image quality. On the other hand, larger embedded information can be used to either increase the TTR or improve the quality of the restored image.

During recent years, this trade-off has been approached using various structures [1, 2, 3, 4, 5, 6, 7, 8]. Zhang et. al. propose a method in which the image representation is generated using a random projection of the discrete cosine transform (DCT) coefficients [9]. In this method, compressive sensing concepts are applied to recover the lost content when the tampering rate is high. Therefore, an estimation of the lost content is available in the receiver, which its quality degrades with increase of the tampering rate. These methods are called "flexible quality" recovery. Korus et. al. have very recently proposed a "constant quality" recovery scheme which works based on modeling the tampering as an erasure channel, and designing the appropriate channel coding structure using fountain codes [10]. The proposed scheme in this paper is also a constant method.

Although many algorithms have been proposed in this field so far, they usually propose a specific design rather than considering the whole problem and offering a solution to compromise the design parameters. In this paper, we model the tampering-protection as a source-channel coding problem. Generating a representation of the original image in its extreme is the same problem as compressing the image using a proper source coding scheme. Applying a source coding to the whole image rather than its blocks, ensures us to achieve the best image representation and the highest performance of the image compression. The output of the source encoder is vulnerable against the tampering, and requires to be protected using a properly designed channel code. As a result, the tampered region is recovered with the quality of the compression method, as long as the tampering rate is lower that the TTR offered by the channel code. Otherwise, the channel decoder fails and the source encoder output is not recovered.

2. PROPOSED WATERMARKING SCHEME

2.1. Basics

The goal of our algorithm is to embed a watermark into original image to protect it against tampering. It means that the watermark must be capable of both finding the tampered areas of the received image, and recovering the content of the original image in those zones. In order to achieve this goal, we keep n_m most significant bits (MSB) of each pixel unchanged, and use the n_w remaining least significant bits

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

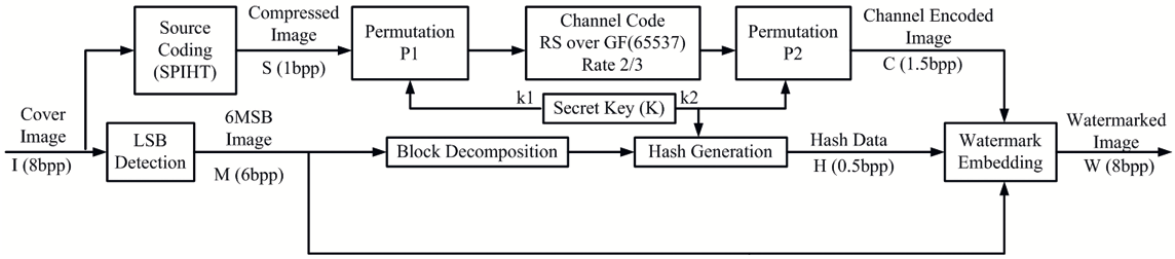

Fig. 1. The generic block diagram of the proposed watermark embedding using two LSBs. System design parameters of n_w, n_s, n_p and n_h equal two, one, half and half respectively.

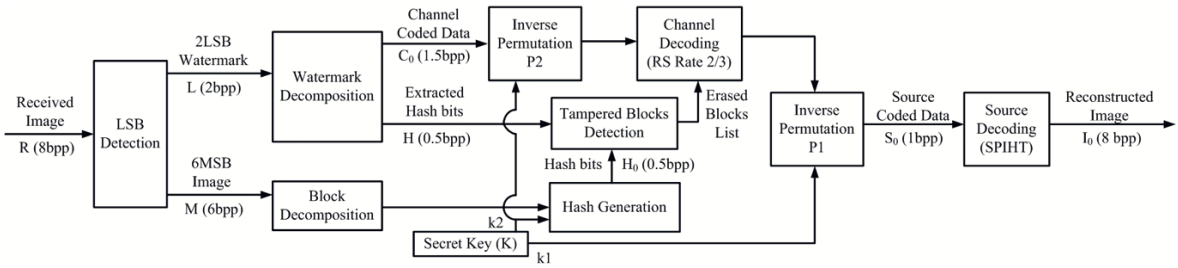

Fig. 2. The block diagram of our tampering detection and image recovery scheme using 2 LSBs of each pixel. System design parameters of n_w, n_s, n_p and n_h equal to two, one, half and half, respectively.

(LSB) for the watermark embedding. For the sake of image recovery, we compress the image using a source encoding algorithm, and embed the result as the watermark. The compressed image bit stream must be channel coded to exhibit robustness against certain amount of tampering. In order to detect tampered blocks at the receiver, some check bits are generated from those parts of image which remain unchanged during watermark embedding procedure. These check bits are inserted as a part of total watermark. As a result, the n_w least significant bits comprise both channel coded bits and check bits. Having tampered blocks known using the check bits, tampering can be modeled as an erasure. At the receiver, the check bits locate tampered blocks. The list of tampered blocks identifies erasure locations and helps the channel erasure decoder to find the compressed image bit stream. Then source encoded image would be decoded and the estimation of the original image is recovered. Following, we discuss about the embedding and recovery phases separately.

2.2. Watermark Embedding

Consider the original image I represented by 8-bit gray-scale pixel values. These eight bits are divided into four parts: The most significant bits that will not change at the watermark embedding phase, check bits, source code bits, and channel code parity bits, denoted by n_m, n_h, n_s and n_p respectively. The n_m MSB of each pixel are remained unchanged during watermark embedding and will be used later for hash generation and image reconstruction. The remaining bits are used for the purpose of watermark embedding.

Assume that N denotes the number of image pixels. We compress the original image into $N_s = N \times n_s$ bits using proper source coding algorithm. In this work, we have implemented the set partitioning in hierarchical trees (SPIHT) image compression algorithm [11], as the source encoder. SPIHT is an embedded compression algorithm, i. e., one can extract an estimation of the original image by truncating its output in every desired rate. This property which fits our design of a general framework, together with the high compression gain when applied over the whole image, have been our main motivations to employ the SPIHT.

Channel coding algorithm of rate $R = n_s/n_c$ is applied to the permuted compressed image bit stream, where $n_c = n_s + n_p$. Knowing the location of a tampered block at the receiver, all of its watermark bits are marked as erased. Therefore, we can integrate these lost bits into a few symbols by setting up the channel code over large fields. The other demand of our application is to implement a channel encoder and decoder that work on long blocks as input and output. In this case, the best performance of the channel code in terms of TTR is achieved, when the whole input bit stream is channel encoded using only a single block. Reed-Solomon (RS) codes [12] can be implemented on the large fields, and automatically can be applied to a very long block of the symbols. Therefore, RS is our choice as the channel code. In the next Section, we show that the whole image can be channel encoded by only applying a single iteration of the channel code.

Channel code yields $N_c = N \times n_c$ bits in total. These bits

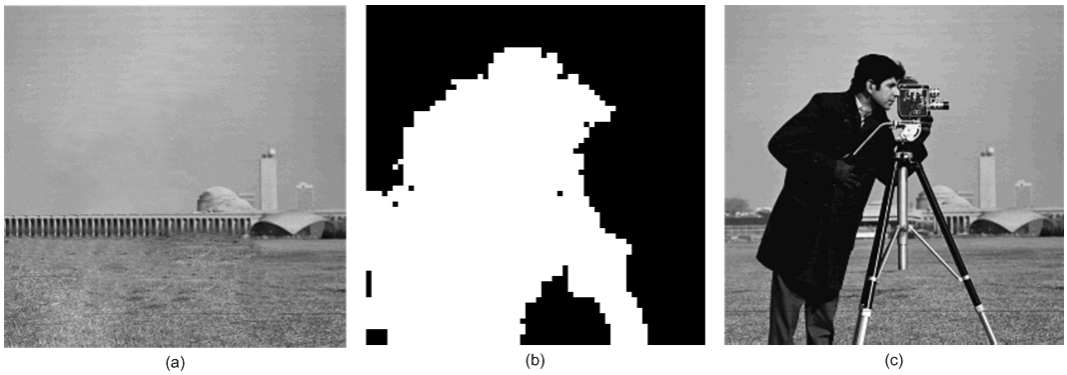

Fig. 3. (a) The produced watermarked image using our proposed 3-LSB method is tampered at the rate of 48.5%. (b) Tampered area detected by check bit examination. (c) The reconstructed image from the tampered one by our 3-LSB proposed method, PSNR=44.9 dB.

are permuted and spread over the whole image, which means every pixel will host n_s source code bits and n_p channel code parity bits. The permutations before and after channel coding are generated using keys k_1 and k_2, both derived from a secret key K, which is known to both embedding phase (transmitter end) and image reconstruction phase (receiver end), to guarantee the security of our algorithm. The original image is divided into blocks of size $B \times B$, thus each block will host $b_c = n_c \times B^2$ channel code bits. These b_c bits have originally belonged to some other blocks, whose rows and column indices are turned into a binary stream of b_{rc} bits called position bits. These b_{rc} position bits along with $b_m = n_m \times B^2$ MSB of each block are used as input to a hash generator algorithm (MD5 here), to produce $b_h = n_h \times B^2$ hash bits. A random binary key of length b_h fixed over whole image is generated at the embedding phase. This key is *XOR*ed with hash bits to generate b_h check bits. These b_h check bits along with b_c channel code bits of each block are spread over that block which results in replacing last $n_w = n_c + n_h$ least significant bits of each pixel of the original image, where n_w is the number of LSB per pixel used for watermark embedding. In the case that n_w LSB of each pixel is used for the sake of watermark insertion, our algorithm is called n_w-LSB. After having all blocks processed, watermarked image is produced. Block diagram of the watermark embedding for 2-LSB algorithm is shown in Fig. 1.

2.3. Tampering Detection and Image Recovery

The received image which is probably tampered is decomposed into blocks of size $B \times B$. For each block, position bits are found using k_2, derived from shared secret key. Block bits are decomposed to n_m MSB and n_w watermark LSB per pixel (bpp), which results in $b_m = n_m \times B^2$ MSB and $b_w = n_w \times B^2$ watermark bits. The watermark bit stream itself is decomposed into $b_h = n_h \times B^2$ check bits and $b_c = n_c \times B^2$ channel code bits. b_{rc} position bits along with b_m MSB are used to generate b_h hash bits. The *XOR* of calculated hash bits and extracted check bits is recorded for each block. For unaltered blocks, this bit stream equals the random key used in the embedding phase. Therefore, comparing these results and spotting the different ones lead to locating the tampered blocks. The probability of missing a tampered block equals 2^{-b_h}, which is almost zero for sufficiently large b_h.

After locating the tampered blocks, the N_c channel code bits are collected through the whole image. Channel code bits are undergoing proper inverse permutation. Then, they are delivered as input to RS erasure decoder along with the erasure locations calculated from the list of tampered blocks. The compressed image bit stream available at the output of the decoder is passed through the source decoder after undergoing proper inverse permutation. The output of source decoder is the reconstructed image. An example of image recovery for 2-LSB algorithm is given in Fig. 2.

3. EXPERIMENTAL RESULTS

8-bit gray scale Cameraman image of size 512×512 is watermarked using a 3-LSB version of our proposed method explained in Section 2. Fig. 3(a) shows the tampering of rate 48.5%. Tampered image had been already protected against high-rate tampering by watermark generated from the 3-LSB version of our algorithm. In this case, we set $n_s = 1$, $n_c = 2.5$ and $n_h = 0.5$. The RS code is constructed on the $GF(2^{16})$. This means that the 512×512 bits at the output of the source encoder can be denoted by 16384 symbols. The channel code rate equals $n_c/n_s = 2.5$, thus; the whole compressed image can be encoded only using a single block of $RS(40960, 16384)$ over $GF(2^{16})$. Note that every RS code with the output length of up to $2^t - 1$ is feasible on the $GF(2^t)$ [12]. Tampered blocks are recognized and their information is perfectly recovered as illustrated in Figs. 3(b) and 3(c).

Next in this section, we compare our proposed algorithm

with two of the most recent works in this field. 2-LSB and 3-LSB versions of our algorithm are both implemented, with the parameters previously mentioned in this article. Zhang's method is a flexible recovery scheme, i. e., the quality of the restored content decays with the tampering rate [9]. Korus's method is a constant recovery algorithm which is implemented for both $\lambda = 1$ and $\lambda = 2$ [10]. The performance comparison is made through all three main parameters of the systems: restored image quality, TTR, and the watermarked image quality.

Zhang's method exploits 3 LSB for the watermark embedding. The maximum quality of the restored image is achieved when the 5 MSB are recovered perfectly, and 3 LSB are replaced with 4. Therefore, the maximum restoration quality in terms of PSNR (peak signal to noise ratio) equals 40.73 dB [6]. The quality decreases with the tampering rate. Korus's method also replaces 3 LSB for watermark embedding, which offers the same maximum restoration quality. But this maximum is not achieved for $\lambda = 1$ scheme [10]. Both Korus's and proposed method are implemented on the 10000 sample images available in BOWS2 data-set [13]. The average restoration quality of Korus's $\lambda = 1$ method equals 36.37 dB, while that of the proposed method is the compression quality offered by the SPIHT at the rate of 1bpp and equals 40.34 dB, when averaged over the whole data-set. This declares an improvement of 4 dB for our proposed method comparing to Korus's $\lambda = 1$ method. Finding the average performance, we compare the results of the algorithms for a sample image (6631.pgm) of this data-set, which can be restored with PSNRs around the mean values when watermarked with Korus's $\lambda = 1$ and the proposed method. The result of this experiment is presented in Fig. 4. The superiority of the proposed method to the Zhang's one is obvious in this figure. Using the same number of LSB for the watermark embedding, the constant recovery performance of the proposed method outperforms the decaying one of the Zhang's method. This improvement exceeds even the significant value of 14 dB for high tampering rates.

The next parameter to analyze is the TTR. TTR is not defined for the Zhang's method. For Korus's method, it is shown that $TTR = \frac{1}{\lambda+1}$, resulting to 50% for $\lambda = 1$ and 33% for $\lambda = 2$ [10]. TTR of our proposed method is determined by the performance of the channel code. Since every $RS(n, k)$ code is capable of correcting up to $n - k$ erasures [14], we have $TTR = 1 - \frac{n_s}{n_c}$ for the proposed method. This yields to the TTR of 33% and 60%, for 2-LSB and 3-LSB versions of our algorithms with mentioned design parameters respectively. This result is confirmed by Fig. 4. It is also inferred from this figure that the 3-LSB version of the proposed method offers a considerably better TTR than Korus's $\lambda = 2$ scheme, while it outperforms Korus's $\lambda = 1$ scheme in terms of both PSNR and restoration quality.

The last parameter to analyze is the quality of the watermarked image, which is determined by the number of the LSB

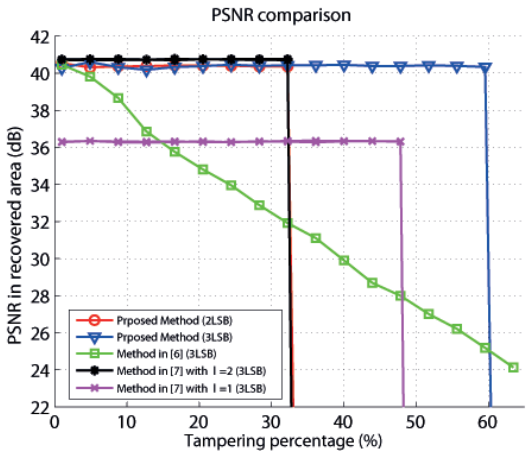

Fig. 4. Simulation results for different methods, expressed as the PSNR in recovered area in terms of tampering rate. All algorithms are tested on sample image 6631.pgm, from [13]

used for the watermark embedding. It can be shown that the PSNR of the watermarked image equals 37.9 dB and 44.15 dB for the algorithms which exploit three and two LSB respectively. This means that the watermarked image generated by a 2-LSB scheme is quite perfect, comparing to the original one. Despite of the other comparing algorithms, our proposed method offers a 2-LSB version which its performance is almost the same as Korus's $\lambda = 2$ scheme. This fact, confirms the improvement achieved by our algorithm in terms of the quality of the watermarked image. All in all, we can conclude that the 2-LSB version of our algorithm offers almost the same performance comparing to these 3-LSB methods, while increasing the watermark size to 3 LSB can be exploited to increase either the quality of the restored content or TTR.

4. CONCLUSION

In this paper, a general source-channel coding framework is proposed to solve the image tampering protection problem. The original image is compressed using an efficient source encoder (SPIHT), and the output bit stream is protected against tampering through RS channel codes. For each block, check bits are calculated and embedded. These bits are used to locate the tampered blocks. If the tampering rate is below a certain limit, the channel erasure decoder succeeds, and the compressed version of the original image is recovered. It is shown that our proposed algorithm using 2 LSB for the watermark embedding, offers almost the same performance comparing to the recent 3 LSB schemes, which are totally outperformed by our 3 LSB version, in terms of both quality of the restored content and TTR. The general viewpoint of our proposed method allows flexibly adapting the parameters to generate different schemes suitable for different applications.

5. REFERENCES

[1] Shao-Hui Liu, Hong-Xun Yao, Wen Gao, and Yong-Liang Liu, "An image fragile watermark scheme based on chaotic image pattern and pixel-pairs," *Applied Mathematics and Computation*, vol. 185, no. 2, pp. 869 – 882, 2007.

[2] Tien-You Lee and Shinfeng D. Lin, "Dual watermark for image tamper detection and recovery," *Pattern Recognition*, vol. 41, no. 11, pp. 3497 – 3506, 2008.

[3] Xinpeng Zhang and Shuozhong Wang, "Fragile watermarking scheme using a hierarchical mechanism," *Signal Processing*, vol. 89, no. 4, pp. 675 – 679, 2009.

[4] Xinpeng Zhang, Shuozhong Wang, and Guorui Feng, "Fragile watermarking scheme with extensive content restoration capability," in *Digital Watermarking*, vol. 5703 of *Lecture Notes in Computer Science*, pp. 268–278. Springer Berlin Heidelberg, 2009.

[5] Chun-Wei Yang and Jau-Ji Shen, "Recover the tampered image based on vq indexing," *Signal Processing*, vol. 90, no. 1, pp. 331 – 343, 2010.

[6] Xinpeng Zhang, Shuozhong Wang, Zhenxing Qian, and Guorui Feng, "Reference sharing mechanism for watermark self-embedding," *Image Processing, IEEE Transactions on*, vol. 20, no. 2, pp. 485–495, 2011.

[7] Zhenxing Qian, Guorui Feng, Xinpeng Zhang, and Shuozhong Wang, "Image self-embedding with high-quality restoration capability," *Digital Signal Processing*, vol. 21, no. 2, pp. 278 – 286, 2011.

[8] P. Korus and A. Dziech, "Reconfigurable self-embedding with high quality restoration under extensive tampering," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*, 2012, pp. 2193–2196.

[9] Xinpeng Zhang, Zhenxing Qian, Yanli Ren, and Guorui Feng, "Watermarking with flexible self-recovery quality based on compressive sensing and compositive reconstruction," *Information Forensics and Security, IEEE Transactions on*, vol. 6, no. 4, pp. 1223–1232, 2011.

[10] P. Korus and A. Dziech, "Efficient method for content reconstruction with self-embedding," *Image Processing, IEEE Transactions on*, vol. 22, no. 3, pp. 1134–1147, 2013.

[11] A. Said and W.A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 6, no. 3, pp. 243–250, 1996.

[12] Stephen B. Wicker, *Reed-Solomon Codes and Their Applications*, IEEE Press, Piscataway, NJ, USA, 1994.

[13] "*The Dataset from the 2nd Bows Contest.*," (2012, Mar. 26) [Online]. Available: http://bows2.ec-lille.fr/.

[14] Shu Lin and Daniel J. Costello, *Error Control Coding, Second Edition*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2004.