Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 350753, 9 pages
http://dx.doi.org/10.1155/2014/350753

Research Article

Color Image Authentication and Recovery via Adaptive Encoding

Chun-Hung Chen,[1] Yuan-Liang Tang,[2] and Wen-Shyong Hsieh[1,3]

[1] *Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 80424, Taiwan*
[2] *Department of Information Management, Chaoyang University of Technology, Taichung City 41349, Taiwan*
[3] *Department of Computer and Communication, Shu-Te University, Kaohsiung 82445, Taiwan*

Correspondence should be addressed to Yuan-Liang Tang; yltang@cyut.edu.tw

Received 18 March 2014; Accepted 13 August 2014; Published 28 August 2014

Academic Editor: Sabri Arik

We describe an authentication and recovery scheme for color image protection based on adaptive encoding. The image blocks are categorized based on their contents and different encoding schemes are applied according to their types. Such adaptive encoding results in better image quality and more robust image authentication. The approximations of the luminance and chromatic channels are carefully calculated, and for the purpose of reducing the data size, differential coding is used to encode the channels with variable size according to the characteristic of the block. The recovery data which represents the approximation and the detail of the image is embedded for data protection. The necessary data is well protected by using error correcting coding and duplication. The experimental results demonstrate that our technique is able to identify and localize image tampering, while preserving high quality for both watermarked and recovered images.

1. Introduction

The revolution of digital technologies has brought many conveniences to our daily lives. For example, it becomes very easy to create, duplicate, transmit, or modify digital products. Accompanying such advance, however, unauthorized use, illegal copying, and malicious modification of digital products become serious problems. A common approach to tackle such problems is the use of digital watermarking techniques, and one of the applications is image content authentication, in which the integrity of an image is considered very important and therefore requires protection.

Researchers tried to develop various image authentication techniques to detect if an image has experienced unauthorized modifications. Some of them can only detect if a certain part of the image has been tampered with, whereas others may have the additional capability to recover the tampered regions. Lin et al. [1] embedded an image block's average intensity and parity check into its corresponding block, which is pseudorandomly determined. The tampered regions are identified based on a three-level hierarchical structure, and the extracted data is used to recover the tampered blocks. With such a hierarchy, the inspection can be performed at different levels and the precision of tamper detection can be achieved close to 100%. Wang and Tsai [2] used fractal codes to generate the approximation for the region of interest (ROI) of the image. Such approximation acts as the recovery data and is embedded into the least significant bits of the image pixels. During the recovery process, the tampered regions in the ROI are recovered using the extracted data, and the blocks outside the ROI are recovered by the inpainting technique instead. One of the problems of their method is that the quality of the recovered blocks outside the ROI is not as good as that in the ROI. Moreover, the original image is required for tamper detection. Zhang and Wang [3] used the five most significant bits of the pixels in each block to generate the reference bits. The check bits are derived by applying a hash function to the block, and they are embedded using a reversible method. The tampered regions are then identified with the extracted check bits. In their method, a tampered

region can be perfectly recovered as long as its total size is less than 3.2% of the whole image. Lee and Lin [4] generated the recovery and verification data of each block and then embedded two copies of them into two other blocks. Such strategy provides more opportunities to recover the block in case one of the two corresponding blocks is corrupted. However, this method suffers from the collage attack. In He et al.'s method [5], an image is divided into blocks of size 8×8, followed by the discrete cosine transform (DCT), and the leading 11 DCT coefficients of each block are embedded into the corresponding block. During authentication, a statistics-based rule is used to determine the validity of a block by considering its adjacent and mapping blocks. This method provides a high detection rate for tampered regions, but they are recovered with only ordinary quality. Qin et al. [6] applied the non-sub-sampled contourlet transform (NSCT) on each image block and selected the low-frequency coefficients to generate the recovery bits. The size of the recovery data of each block is determined by the block's characteristics and the smooth blocks are encoded with less data than the textured ones. Their method maintains high quality for watermarked images by effectively controlling the size of the embedding data. Qian et al. [7] also divided an image into blocks of size 8×8, followed by the DCT, and each block is classified into one of the several types according to the index of the last nonzero DCT coefficients. Several DCT coefficients are encoded with variable-length coding and embedded into the three least significant bits (LSBs), and the length of the code is determined by the type of the block. During the recovery process, the data is extracted using dequantization and the inverse DCT. Their method recovers the tamper areas with high image quality.

While most of the existing techniques deal with grayscale images, others are designed specifically for color images. Wang and Chen [8] computed the means of YCbCr as the recovery data, and the authentication data is derived from the global and local features. During authentication, all the extracted authentication data are taken into consideration and a majority-voting strategy is used to determine the validity of the image block. Their method recovers the tampered regions with acceptable quality. Liu [9] utilized the block-edge pattern to describe the details of the luminance channel and to preserve more complete luminance information. After pattern matching, the best index is recorded. The positive gradient is obtained by calculating the difference between the mean and the bright pixels, whereas the negative gradient is obtained by the mean and the dark pixels. The method can achieve good quality for both watermarked and recovered images.

In this paper, we describe an authentication and recovery scheme for color image protection based on adaptive encoding. The image blocks are categorized according to their contents and different encoding schemes are applied according to their types. Such adaptive encoding results in better image quality and more robust image authentication. The approximations of the luminance and chromatic channels are carefully calculated, and for the purpose of reducing the data size, differential coding is used to encode the channels with variable size according to the characteristic of the block. The recovery data which represents the approximation and detail of the image is embedded for data protection. The necessary data is well protected by using error correcting coding (ECC) and duplication. This paper is organized as follows. The embedding and authentication processes are described in Section 2. Several experimental results are presented in Section 3 and Section 4 gives some concluding remarks. The experimental results demonstrate that our technique is able to identify and localize image tampering, while preserving high quality for both watermarked and recovered images.

2. The Proposed Method

2.1. Watermark Embedding. Let $X = \{(R, G, B)\}$ be the host color image of size $M \times M$, and its red, green, and blue components are denoted as $R = \{r_{ij} \mid 1 \le i, j \le M\}$, $G = \{g_{ij} \mid 1 \le i, j \le M\}$, and $B = \{b_{ij} \mid 1 \le i, j \le M\}$, respectively. Figure 1 delineates the watermark embedding process, which consists of the following steps.

(1) The original image X is divided into nonoverlapping blocks of size 2×2 pixels, which produces $(M/2)^2$ blocks in total. A block in image X is denoted as X_{mn}, $0 \le m, n < M/2$, and the red channel of X_{mn} is denoted as $R_{mn} = \{r_{ij} \mid 2m \le i < 2(m+1), 2n \le j < 2(n+1), 0 \le m, n < M/2\}$. The green and blue channels of block X_{mn} are denoted as G_{mn} and B_{mn}, respectively.

(2) Because the human visual systems are more sensitive to luminance than chromatic changes, transforming the RGB color model into another in which luminance and chromatic channels are separated would result in better image analysis. One of such models is the YCbCr model, in which Y presents the luminance and Cb and Cr represent the chromatic information, respectively. Denoting the transformed version of X as $T = \{(Y, \mathrm{Cb}, \mathrm{Cr})\}$, the three channels of block T_{mn} can thus be denoted as Y_{mn}, Cb_{mn}, and Cr_{mn}, respectively.

(3) Next, we analyze the intensity and texture (i.e., contrast and edge) features of the image. The four pixels in block Y_{mn} are denoted by y_a, y_b, y_c, and y_d, as illustrated in Figure 2. The top-left pixel, y_a, of each block is recorded and the collection of all y_a's forms a set of $(M/2)^2$ vectors, with the length of each vector being 8 bits. The vector set is further processed using the LBG vector quantization technique [10, 11] to generate a codebook b_I consisting of 64 codewords, $b_I = \{b_i \mid 0 \le i < 64\}$. An example of b_I is illustrated in Figure 3.

The intensity feature of block Y_{mn} is obtained by computing the similarities between pixel y_a and the codewords in the codebook:

$$\left(b_j, j_I\right) = \min_i \left(\left|y_a - b_i\right|\right). \tag{1}$$

Function min( ) returns two parameters, b_j and j_I, where b_j is the codeword most similar to y_a and j_I is its index. The index, whose size is 6 bits ($2^6 = 64$), will be recorded, and the block is modified by replacing y_a with b_j.

(4) Next, the remaining pixels, $y_{b\sim d}$, will be encoded. Because adjacent pixels in a natural image are usually highly correlated, recording their relationship is an effective way of reducing the encoding size. Such an idea is realized by

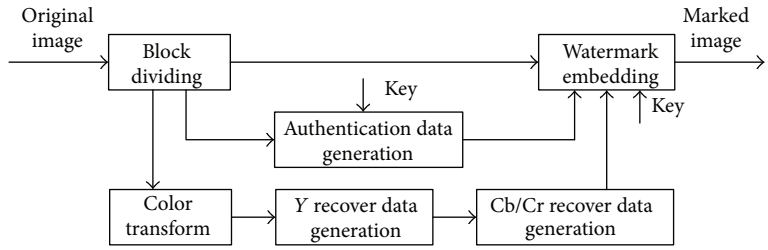


Figure 1: Watermark embedding process.

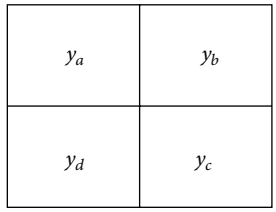


Figure 2: The four pixels of block Y_{mn}.

b_1

Index	Codeword
0	35
1	37
⋯	⋯
⋯	⋯
⋯	⋯
63	210

Figure 3: An example of codebook b_1.

the differential coding technique, in which the differences between pairs of the pixels are calculated first:

$$\delta_1 = y_b - y_a, \qquad \delta_2 = y_c - y_b, \qquad \delta_3 = y_d - y_c. \tag{2}$$

And then the maximum of the absolute differences, $\delta_M = \max(|\delta_1|, |\delta_2|, |\delta_3|)$, is used to classify the block into four types according to Table 1. The blocks of different types will be encoded with different codebooks consisting of different number of codewords. A uniform block will be encoded with a small codebook, whereas a block with large differences will be encoded with a large codebook in order to reduce the error.

The average difference, $\delta_A = \text{avg}(|\delta_1|, |\delta_2|, |\delta_3|)$, is obtained for each block. Those δ_A's of type-3 blocks are collected to form a dataset, and the K-means clustering technique [12, 13] is used to separate the dataset into five clusters (subtypes). That is, type-3 blocks are further divided into five subtypes, that is, types 3.1~3.5, and for each subtype, we collect δ_1, δ_2, and δ_3 to form a set which have $3 \times (M/2)^2$ elements.

Figure 4: Two examples of differential encoding.

Table 1: Block types defined by the pixel differences.

δ_M	Type
0, 1	0
2	1
3, 4	2
≥5	3

The vector quantization technique is then used to generate a codebook consisting of 16 representative codewords (i.e., differences) of each type (for different average magnitudes). An example of the block types and codebooks, b_D's, of different coding is shown in Table 2, in which the codebooks of types 1 and 2 are predefined without training and the codebooks of types 3.1 to 3.5 are generated using the vector quantization technique. For each Y block, we encode the three differences with its corresponding codebook, and three bits are used to encode the block type. Furthermore, two, three, and four bits are used to encode the differences of the blocks of types one, two, and three, respectively. To better explain the encoding process, Figure 4 illustrates two examples of Y_{mn}. In the first example, $\delta_1 = y_b - y_a = 51 - 50 = 1$, $\delta_2 = y_c - y_b = 50 - 51 = -1$, $\delta_3 = y_d - y_c = 51 - 50 = 1$ and $\delta_M = \max(|1|, |-1|, |1|) = 1$. According to Table 1, this block is of type 0. Because the neighbors are almost the same, their differences are not recorded, and the only thing to encode is the block type (3 bits), t_D. In the second example, $\delta_1 = 23 - 20 = 3$, $\delta_2 = 23 - 23 = 0$, $\delta_3 = 21 - 23 = -2$, and $\delta_M = \max(|3|, |0|, |-2|) = 3$. Hence, this block is of type 2, and the most similar code to δ_1 (= 3) is also 3, under index $j_{\delta 1} = 6$. Likewise, the most similar codeword to δ_2 (= 0) is 1 and −1. The smaller codeword, −1, under index $j_{\delta 2} = 3$ is chosen because it is closer to the codeword of δ_3 (= −2). To compensate the error, δ_3 is further updated by $\delta_3' = \delta_3 + (\delta_2 - (-1)) = (-2) + (0 - (-1)) = -1$. The appropriate code for δ_3' is then −1 under index $j_{\delta 3} = 3$. Therefore, the data to be recorded is the block type and the indices of the codewords

Table 2: Block types and codebooks b_D of differential coding.

Type	δ_M	Codeword															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	**0, 1**	**N/A**															
1	**2**	**−2**	**−1**	**1**	**2**												
2	**3, 4**	**−4**	**−3**	**−2**	**−1**	**1**	**2**	**3**	**4**								
3.1	≥5	−11	−10	−9	−8	−7	−5	−2	0	1	2	4	5	7	8	10	11
3.2	≥5	−20	−17	−15	−12	−10	−7	−5	−2	0	3	6	8	12	13	16	19
3.3	≥5	⋯															
3.4	≥5	⋯															
3.5	≥5	⋯															

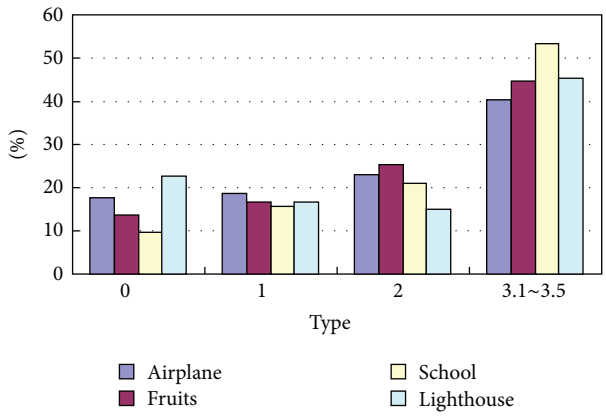


Figure 5: The percentages of each block.

b_{Cb}

Index	Codeword
0	114
1	118
⋯	⋯
15	163

b_{Cr}

Index	Codeword
0	107
1	111
⋯	⋯
15	182

Figure 6: Examples of codebooks b_{Cb} and b_{Cr}.

of δ_1, δ_1, and δ_3. The size of each index of types 1, 2, and 3 are 2, 3, and 4 bits, respectively. Therefore, the total sizes of the data to be recorded of types 0, 1, 2, and 3 is 3, 9 (= 3 + 2 × 3), 12 (= 3 + 3 × 3), and 15 (= 3 + 4 × 3), respectively. Codebooks of type 3, denoted as b_Y, will be recorded. We have conducted several experiments to understand the distributions of each block type in a few images as shown in Figure 5. About 45% to 60% blocks belong to types 0 to 2, which indicates that neighboring pixels are very similar as expected, and they can be encoded with smaller sizes.

(5) For each Cb block, the mean of four Cb pixels is calculated. And then, the vector quantization technique is used to generate a codebook consisting of 16 representative codewords for the means. The index of the best codeword for a Cb mean, j_{Cb}, will be recorded with size of 4 ($2^4 = 16$) bits. The processing of Cr blocks is identical to that of Cb blocks. The codebooks of Cb and Cr are denoted as b_{Cb} and b_{Cr}, respectively, and Figure 6 gives examples of them.

(6) For an RGB block, three LSBs of channel R, two LSBs of channel G, and three LSBs of channel B are cleared before hashing, and a random value is generated as the seed of the hash function. The LSB-cleared block ($2 \times 2 \times 3 = 12$ bytes) is hashed using the SHA-1 algorithm [14]. The XOR of the even bits of the 180-bit hash values are obtained to generate the first bit of the authentication data, and the XOR of the odd bits of the 180-bit hash values are obtained to generate the second bit of the authentication data. The 2-bit authentication data is denoted as a.

(7) Now, the global data consists of four codebooks: b_I, b_D, b_{Cb}, and b_{Cr}. To protect the data, two techniques are used: error correcting coding (ECC) and duplicating (majority voting). As the Reed-Solomon (7, 3) coding [15] has 28.6% (2/7) correcting capability, it is used to encode the global data and the encoded data are further duplicated into N copies. The duplication dramatically improves the robustness, and even if some copies (less than $N/2$) are corrupted, the bit can still be restored correctly. Let C denote the length of ECC-encoded data; N is determined by $\lfloor (M/2)^2/C \rfloor$. The duplicated string is zero padded to make the total size be $(M/2)^2$. The final data is separated into $(M/2)^2$ bits and each bit for block (i, j) is denoted as g_{ij}. Thus, the payload of each block is $j_I \| j_{\text{Cb}} \| j_{\text{Cr}} \| a \| g_{ij} \| t_D \| j_{\delta 1 \sim 3}$ (if any).

(8) The corresponding block of block (i, j) is determined by using the Torus Automorphism [16]. A prime number is chosen by the user and treated as a private key. The LSBs of the corresponding block are replaced with the variable-sized payload, with four different sizes: 20, 26, 29, and 32 bits. The bit allocation is shown in Figure 7, and to maintain imperceptibly, the embedding order is B, R, and then G channel.

2.2. Authentication and Recovery. During authentication, the same process is applied on the received image X', as shown in Figure 8. X' is divided into blocks, and for each block, the position (index) of its corresponding block is computed using the Torus Automorphism with the same key. Because there are four different sizes of the payload, we first extract the 20-bit data from LSBs of the corresponding block. According to the last three bits of the extracted data, t'_D, we will know the size of the rest unextracted data. We then extract the following 0, 6, 9, or 12 bits. After obtaining the payload of each block, we have the complete data: j'_I, j'_{Cb}, j'_{Cr}, a', g'_{ij}, t'_D, and $j'_{\delta 1 \sim 3}$ (if any).

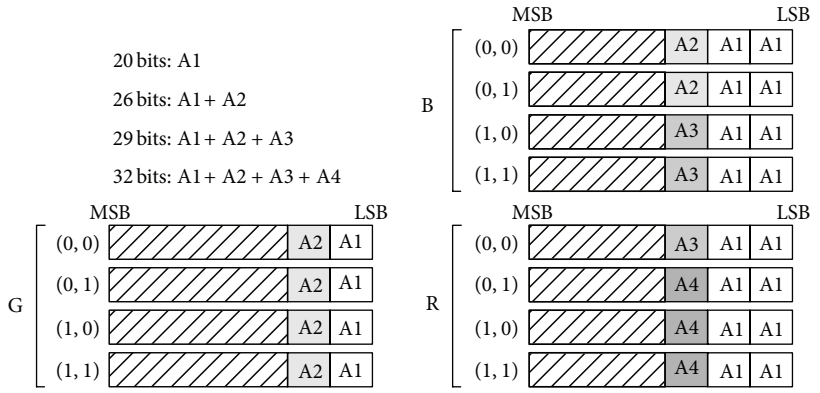


Figure 7: Bit allocation.

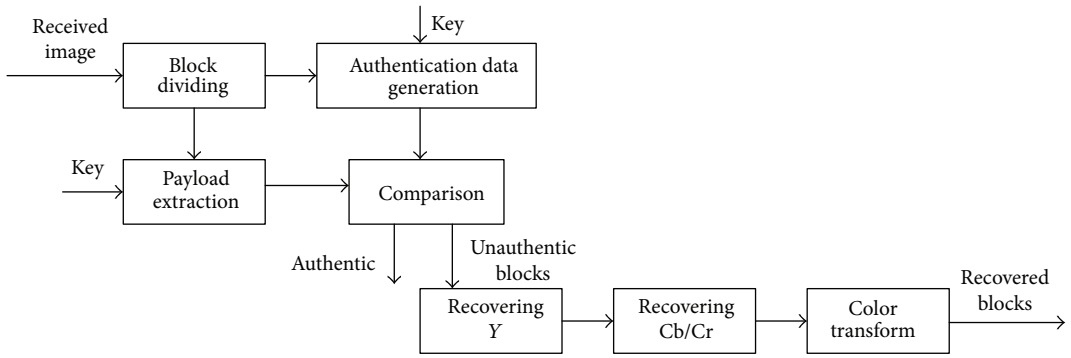


Figure 8: Authentication process.

The encoded string of the global data is obtained by combining g_{ij} which are collected from each block. The sting is separated into N pieces with each piece decoded by ECC decoding. Then, the majority voting is applied on each bit of the N decoded results. If the count for a bit exceeds $N/2$, the bit is 1; otherwise, the bit is 0. The results of the processing are thus the four codebooks: b'_I, b'_D, b'_{Cb}, and b'_{Cr}.

A binary authentication map with size of $(M/2) \times (M/2)$ is set as zero. For each block, the hash data, a'', is produced with the correct seed and then compared with a' (the original one). If a mismatch occurs, the corresponding pixel on the map is set as one for an unauthentic block. If there are no mismatches after checking all blocks, the image is authentic and the authentication process terminates. Otherwise, the binary map will be further processed to generate the final map which shows the position of the tampered blocks. Because an attacker usually tries to alter the semantics of the image, tampered pixels tend to cluster together (may be in several locations). Randomly altering the pixels is meaningless and hence not likely to happen. Morphological operations can be used to remove the (noise like) false positives and concentrate the shapes of meaningful tampered regions. Based on this analysis, dilation and erosion operations are applied on the map, and according to the processed map, the tampered regions (which may consist of many blocks) are localized.

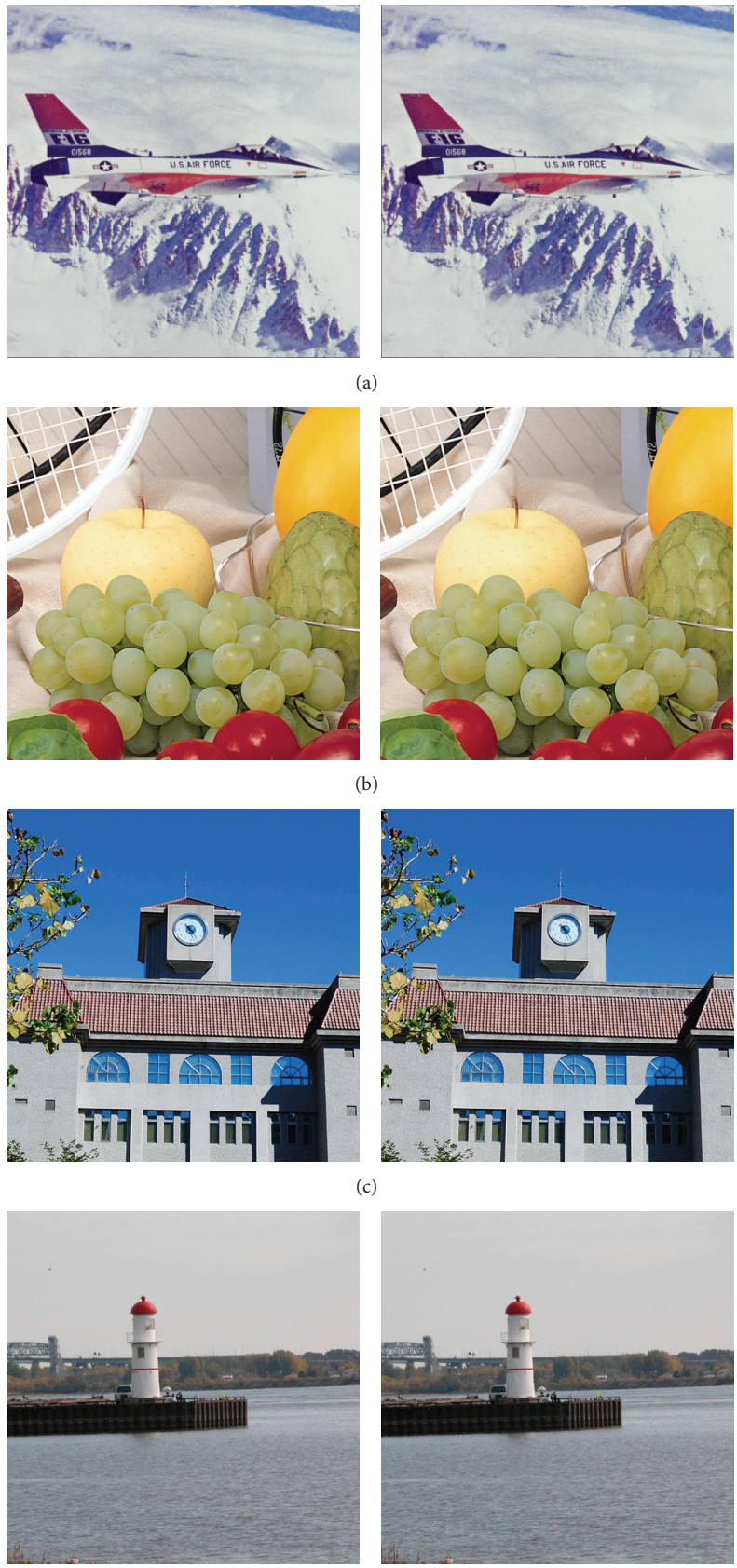
After localizing the tampered blocks, their three channels, R''_{mn}, G''_{mn}, and B''_{mn}, will be recovered by the following process.

(1) By using the table-lookup process, the top-left pixel of Y block Y'_{mn}, y'_a, is obtained by finding j'_I from b'_I. The codebooks of types 0, 1, and 2 are predefined and the codebooks of type 3 are b'_D. According to t'_D, we select the corresponding codebook and the three differences, δ'_1, δ'_2, and δ'_3, are obtained from indices $j'_{\delta 1}$, $j'_{\delta 12}$, and $j'_{\delta 3}$, respectively. The other three pixels of Y'_{mn} can be decoded by using the following equations: $y'_b = y'_a + \delta'_1$, $y'_c = y'_b + \delta'_2$, and $y'_d = y'_c + \delta'_3$. Thus block Y'_{mn} has been recovered.

(2) Through table-lookup, a corresponding codeword for Cb is obtained by finding j'_{Cb} from b'_{Cb}. A 2×2 block, Cb'_{mn}, is constructed by duplicating the codeword four times. The Cr block Cr_{mn} is obtained exactly the same as Cb blocks. Thus blocks Cb'_{mn} and Cr'_{mn} are recovered.

(3) R'_{mn}, G'_{mn}, and B'_{mn} are obtained by transforming Y'_{mn}, Cb'_{mn}, and Cr'_{mn} from YCC to RGB color model. The tampered block is recovered by replacing R''_{mn}, G''_{mn}, and B''_{mn} with R'_{mn}, G'_{mn}, and B'_{mn}, respectively.

The chromatic channels of the block are recovered with the approximation and the luminance channel is recovered with fine details. Such mechanism preserves the image quality very well.



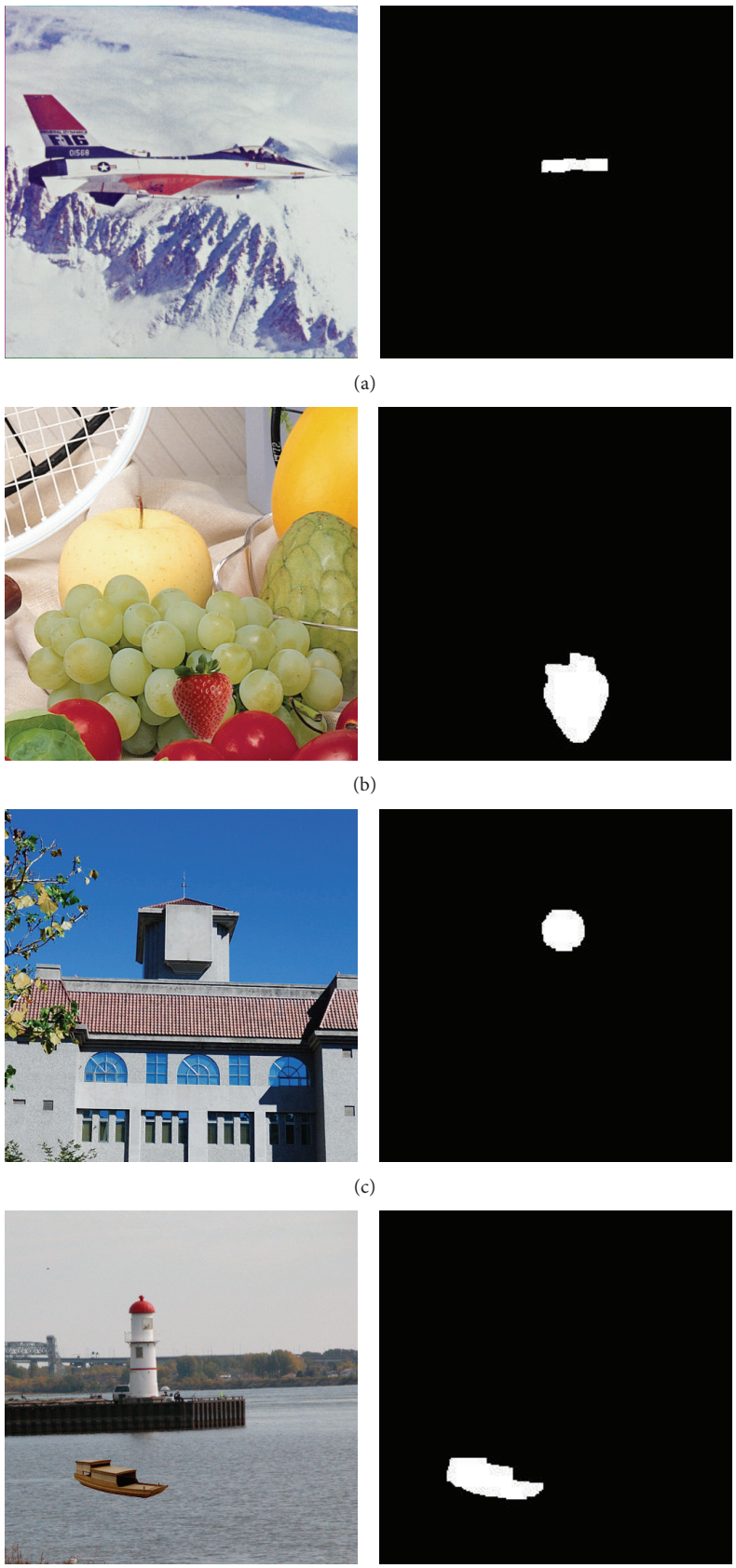
(a)

(b)

(c)

(d)

Figure 9: Original (left column) and watermarked (right column) images: (a) airplane, (b) fruits, (c) school, and (d) lighthouse.



(a)

(b)

(c)

(d)

Figure 10: Tampered images (left column) and results of tamper detection (right column): (a) airplane, (b) fruits, (c) school, and (d) lighthouse.

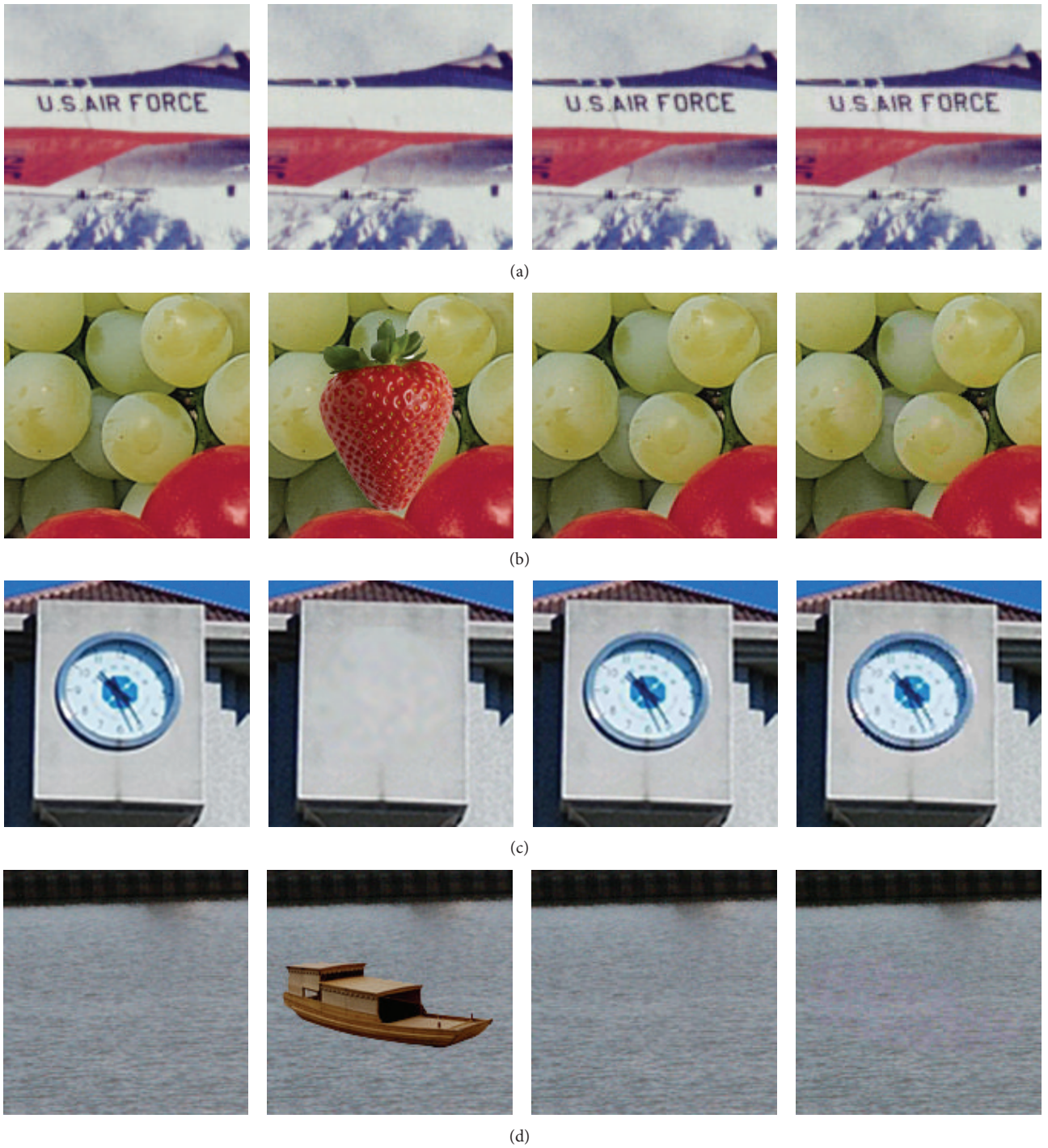


Figure 11: Local areas of original images (left column), tampered images, images recovered by the proposed method, and images recovered by Liu's method (right column): (a) airplane, (b) fruits, (c) school, and (d) lighthouse.

Table 3: PSNR values of watermarked images.

Method	Airplane	Fruits	School	Lighthouse	Average
The proposed method	40.60	40.33	40.25	40.73	**40.48**
Liu's method	39.51	38.75	39.22	38.95	**39.11**

Table 4: PSNR values between recovered images and original images.

Method	Airplane	Fruits	School	Lighthouse	Average
The proposed method	40.50	40.09	40.03	40.68	**40.33**
Liu's method	38.55	37.59	38.46	38.11	**38.18**

3. Experimental Results

Four 256 × 256 color images were tested in our experiments: *airplane*, *fruits*, *school*, and *lighthouse*. There are thus $(256/2)\times(256/2) = 16,384$ blocks in total, and the number of duplications, N, is 18. The original and watermarked images produced by the proposed method are shown in Figure 9. As can be seen, these images all have high visual quality. The performance of our method is compared against that of Liu's method [9]. Table 3 lists the comparisons of the PSNR values and it shows that our PSNR levels are quite acceptable and our method outperforms Liu's.

To demonstrate the effectiveness of our technique, we performed various attacks on the watermarked image including removing the words from airplane, placing a strawberry on the bottom, removing the clock from the building, and placing a boat on water. The tampered images and their detection results are shown in Figure 10. As is obvious in the figure, the tampered regions (blocks) are all correctly detected and localized. The tampered regions (blocks) are further recovered. Figure 11 shows the local areas of the results recovered by the proposed and Liu's methods. In addition, Table 4 lists the PSNR values of the recovered images produced by the proposed and Liu's methods. Compared with the original images, we can see that the images recovered by the proposed method have high fidelity both for colors and textures. The PSNR values also verify such results. Comparing the PSNR values between the watermarked and recovered images, the proposed method only produces tiny degradations. However, we can see apparent chromatic degradation in the images produced by Liu's method. From both Figure 11 and Table 4, it is obvious that our method outperforms Liu's.

4. Conclusion

An authentication and recovery scheme for color images is proposed in this paper. The approximations of the luminance and chromatic channels are properly calculated, and to reduce the data size, differential coding is used to encode the details of the luminance channel with variable size according to the characteristic of the block. The recovery data which represents the approximation and the details of the image is embedded in the image for data protection. The important data is well protected by using the ECC technique and duplication. The experimental results demonstrate that our technique is able to identify and localize image tampering, while preserving high quality for both watermarked and recovered images.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] P. L. Lin, C. Hsieh, and P. Huang, "A hierarchical digital watermarking method for image tamper detection and recovery," *Pattern Recognition*, vol. 38, no. 12, pp. 2519–2529, 2005.

[2] S. Wang and S. Tsai, "Automatic image authentication and recovery using fractal code embedding and image inpainting," *Pattern Recognition*, vol. 41, no. 2, pp. 701–712, 2008.

[3] X. Zhang and S. Wang, "Fragile watermarking with error-free restoration capability," *IEEE Transactions on Multimedia*, vol. 10, no. 8, pp. 1490–1499, 2008.

[4] T. Lee and S. D. Lin, "Dual watermark for image tamper detection and recovery," *Pattern Recognition*, vol. 41, no. 11, pp. 3497–3506, 2008.

[5] H. J. He, J. S. Zhang, and F. Chen, "Adjacent-block based statistical detection method for self-embedding watermarking techniques," *Signal Processing*, vol. 89, no. 8, pp. 1557–1566, 2009.

[6] C. Qin, C. Chang, and P. Chen, "Self-embedding fragile watermarking with restoration capability based on adaptive bit allocation mechanism," *Signal Processing*, vol. 92, no. 4, pp. 1137–1150, 2012.

[7] Z. Qian, G. Feng, X. Zhang, and S. Wang, "Image self-embedding with high-quality restoration capability," *Digital Signal Processing: A Review Journal*, vol. 21, no. 2, pp. 278–286, 2011.

[8] M. S. Wang and W. C. Chen, "A majority-voting based watermarking scheme for color image tamper detection and recovery," *Computer Standards and Interfaces*, vol. 29, no. 5, pp. 561–570, 2007.

[9] K. C. Liu, "Colour image watermarking for tamper proofing and pattern-based recovery," *IET Image Processing*, vol. 6, no. 5, pp. 445–454, 2012.

[10] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic, Boston, Mass, USA, 1992.

[11] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications Systems*, vol. 28, no. 1, pp. 84–95, 1980.

[12] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, 1967.

[13] E. Forgy, "Cluster analysis of multivariate data: efficiency versus interpretability of classifications," *Biometrics*, vol. 21, no. 3, pp. 768–769, 1965.

[14] D. Eastlake and P. Jones, "US secure hash algorithm 1 (SHA1)," Tech. Rep. 3174, IETF Request for Comments, 2001.

[15] B. Sklar, *Digital Communications: Fundamentals and Applications*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2nd edition, 2001.

[16] G. Voyatzis and I. Pitas, "Chaotic mixing of digital images and applications to watermarking," in *Proceedings of the European Conference Multimedia Application, Service, and Techniques*, vol. 2, pp. 687–695, 1996.