2017 2nd International Conference on Software, Multimedia and Communication Engineering (SMCE 2017)
ISBN: 978-1-60595-458-5

Robust Image Hashing Based on Local Features for Image Authentication

Yan ZHAO* and Qian ZHAO

School of Electric and Information Engineering, Shanghai University of Electric Power, Shanghai, China

*Corresponding author

Keywords: Image hashing, Local feature, Image authentication.

Abstract. In this paper, an image hashing method based on local features is proposed. At first the input image is pre-processed and divided into un-overlapped blocks. We choose several blocks as the effective blocks using SIFT features. Then color, texture and shape features of the selected blocks are extracted, connected and permuted to form the final hash. Experimental results show that this method is robust against most content-preserving attacks. Collision probability of this method is smaller than other methods. This method can also be used to detect tampering image.

Introduction

Recently, more and more images videos or audios are processed by different tools for different purposes. So it is difficult for users to differentiate the authentic media from the counterfeits. Image hashing is one of the important methods for image authentication. The concept of image hashing is derived from cryptographic hashing. A cryptographic hash is extremely sensitive to the input data: even one bit change in the input will change the output hash dramatically. However, for an image, even after many manipulations such as JPEG compression, it is the same in terms of human vision. Therefore original image and the content-preserved processed image should have similar hash. On the other hand, hashes of two different images should be totally different. Many researchers have proposed image hashing methods in recent years. Monga et al. [1] develop a two-step framework that includes feature extraction (intermediate hash) and coding of the intermediate result to form the final hash. That has become a routine practice in many image hashing methods. Many previous schemes are either based on global [2-5] or local [6-11] features.

In this paper, we propose a method to construct robust image hashing method using features of selected effective blocks. Then color, texture and shape features of selected blocks are extracted and connected to form the intermediate hash. Lastly, the final hash sequence is obtained by pseudo-randomly permuting the intermediate hash sequence. Similarity between hashes is measured by Euclidean distance. Simulation results show that the scheme is robust against most content-preserving attacks. This method has lower collision probability than other methods. In addition, the proposed hash scheme is capable of detecting forged images containing inserted foreign objects or deleting some objects.

Image Hashing Generation

In this section, the image hashing method is proposed. The block diagram of the image hashing method is shown in Figure 1, and the steps are as follows.

(1) The input image is pre-processed firstly. The image is resized to a standard size L×L using bi-linear interpolation, namely I, as showed in Figure 2(a). Then the luminance component of I is extracted, namely I_l, as shown in Figure 2(b).

(2) Sift points in image I_l are extracted. Then I_l is partitioned into non-overlapped blocks, each of dimension Q×Q pixels as shown in Figure 2(c). Each block is marked with a label. The number of SIFT points in the blocks are calculated. The numbers are sorted in descending order. If a block has

more SIFT points, it has more effective information in the image. So we only choose the first several blocks to extract features, as shown in Figure 2(d). The labels of the selected blocks are connected to form Flag vector, namely F.

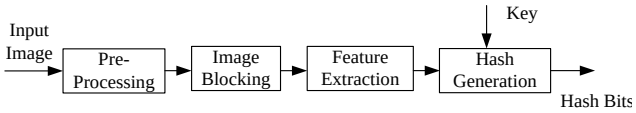

Figure 1. Hash generation module.

(3) The color image I is also partitioned into non-overlapped blocks. Then color, texture and shape features of the selected blocks are extracted. Each block is converted from RGB to HSV, and then turned to H2SV. We can get four components H1, H2, S and V. Mean of the four components constitute the color vector C. Texture features T include Tamura coarseness and contrast features, skewness, and kurtosis. Zernike moments of the blocks in I_l are calculated. Because shape features can be obtained from a small number of low frequency coefficients, the order *n* does not need to be large. And with the increase of *n*, the calculation complexity and numbers of Zernike moments are increased greatly. Further, only $Z_{n,m}$ ($m \geq 0$) is used as the image feature since $Z_{n,-m} = Z_{n,m}$. $Z_{0,0}$ represent the mean of intensity, so we do not use this feature. Table 1 lists the Zernike moments features from orders 0 through 5. The total number of Zernike moments is named by Tn (n is the order). All the Zernike moments is connected to constitute shape vector S.

(4) Then the feature vectors of all selected blocks are connected to form the intermediate hash [F, C, T, S].

(5) The final hash sequence is obtained by pseudo-randomly permuting the binary sequence obtained in the previous step.

Table 1. List of Zernike Moments for Different Orders.

Order *n*	Zernike moments	Number of the moments	Cumulative number
0	$Z_{0,0}$	1	1
1	$Z_{1,1}$	1	2
2	$Z_{2,0}$, $Z_{2,2}$	2	4
3	$Z_{3,1}$, $Z_{3,3}$	2	6
4	$Z_{4,0}$, $Z_{4,2}$, $Z_{4,4}$	3	9
5	$Z_{5,1}$, $Z_{5,3}$, $Z_{5,5}$	3	12

(a) Original image (b) Luminance image (c) Labels of the blocks (d) Ten effective blocks

Figure 2. Image Blocking and Marking.

Image Authentication

The block diagram of image authentication is shown in Figure 3. The steps are as follows.

(1) Hash decomposition. With the secret key, restore the intermediate hash from the reference hash to obtain $H_1=[F_1, C_1, T_1, S_1]$, which is a concatenated feature sequence of the trusted image.

(2) Feature extraction. Pass the test image through the steps as described in previous section to obtain the intermediate hash without encryption, namely $H_2= [F_2, C_2, T_2, S_2]$.

(3) Region matching. If labels in F_1 and F_2 are same, then the blocks marked by the labels are matched. Reshuffle the vectors by moving the matched components in each of the vector pair to the leftmost. For example, if there are three selected blocks in the reference image and four in the test

image, the vectors are shown as: $F_1=[312350000000]$, $C_1=[c_1^{(1)}\ c_1^{(2)}\ c_1^{(3)}\ 0000000]$, $T_1=[t_1^{(1)}\ t_1^{(2)}\ t_1^{(3)}\ 0000000]$, $S_1=[s_1^{(1)}\ s_1^{(2)}\ s_1^{(3)}\ 0000000]$; $F_2=[532435000000]$, $C_2=[c_2^{(1)}\ c_2^{(2)}\ c_2^{(3)}\ c_2^{(4)}\ 000000]$,

$T_2=[t_2^{(1)}\ t_2^{(2)}\ t_2^{(3)}\ t_2^{(4)}\ 000000]$, $S_2=[s_2^{(1)}\ s_2^{(2)}\ s_2^{(3)}\ s_2^{(4)}\ 000000]$.

We can see they have two matched blocks and the labels are 3 and 35. After matching, the feature vectors are changed as: $F_1=[335120000000]$, $C_1=[c_1^{(1)}\ c_1^{(3)}\ c_1^{(2)}\ 0000000]$, $T_1=[t_1^{(1)}\ t_1^{(3)}\ t_1^{(2)}\ 0000000]$, $S_1=[s_1^{(1)}\ s_1^{(3)}\ s_1^{(2)}\ 0000000]$; $F_2=[335524000000]$, $C_2=[c_2^{(2)}\ c_2^{(4)}\ c_2^{(1)}\ c_2^{(3)}\ 000000]$,

$T_2=[t_2^{(2)}\ t_2^{(4)}\ t_2^{(1)}\ t_2^{(3)}\ 000000]$, $S_2=[s_2^{(2)}\ s_2^{(4)}\ s_2^{(1)}\ s_2^{(3)}\ 000000]$.

(4) Distance calculation and judgment. We use a distance between hashes of an image pair as a metric to judge similarity/dissimilarity of the two images. To define the hash distance, a feature vector **V** is formed by concatenating the feature vectors after matching, namely **V** = [C T S]. The hash distance between the test image and the reference is defined as the Euclidean distance between $\mathbf{V}_1$ and $\mathbf{V}_2$:

$$D=\|\mathbf{V}_1-\mathbf{V}_2\| \tag{1}$$

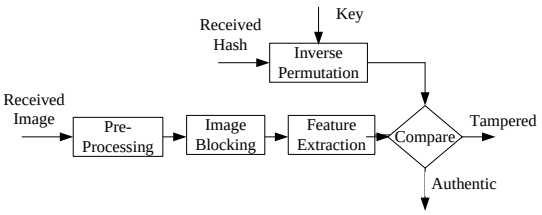

Figure 3. Image authentication module.

Experiments Results

In the experiment, L=512, Q=64, X=10, n=4, Tn=8. So the hash has 10+40+40+80=170 numbers.

Robustness Test

Three standard images sized 512×512 were used in the experiment: Airplane, House and Lena. To test robustness of the hash, StirMark 4.0[12] was used to perform attacks including JPEG coding, additive noise contamination, watermark embedding and image scaling. In addition, brightness adjustment, contrast adjustment and rotation using Adobe Photoshop, and gamma correction and 3×3 Gaussian filtering using MATLAB are also tested. The types of attacks and the parameters used are listed in Table 2. The indices in the left-most column correspond to the abscissa of Figure 4.

Distances between hashes of the original and attacked images are calculated. The results are presented in Figure 4. We observe that most distances are no more than 100, with a few exceptions corresponding to manipulations that cause significant changes in the image intensity. Thus, we can safely set a threshold at 100 to judge whether or not two images can be considered visually identical. If the distance is greater than the threshold, the two images are considered different.

Table 2. Parameters used in the robustness experiment.

Indices	Attack	Description	Parameter value
1-9	JPEG compression	Quality factor	20, 30, … , 100
10-11	Additive noise	Level	1, 2
12-21	Watermark embedding	Strength	10, 20, … , 100
22-27	Scaling	Ratio	0.5, 0.75, 0.9, 1.1, 1.5, 2.0
28-31	Brightness adjustment	Photoshop's brightness scale	10, 20, -10, -20
32-36	Contrast adjustment	Photoshop's contrast scale	10, 20, -10, -20
37-39	Gamma correction	γ	0.75, 0.9, 1.1, 1.25
40-49	3×3 Gaussian filtering	Standard deviation	0.1, 0.2, ... , 1.0

Uniqueness Test

Uniqueness means that two hash sequences from two different images should be sufficiently different. Figure 5 shows the distribution of the Hash distance calculated from C^2_{1000} =4995000 hash pairs with 1000 different images. We can see most of the hash distances are bigger than 100.

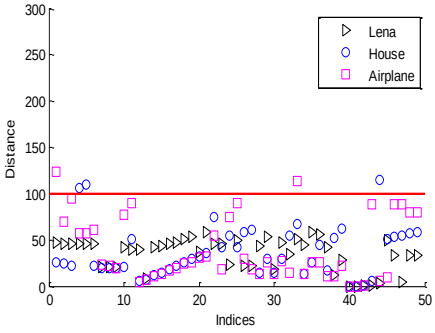

Figure 4. Robust test results.

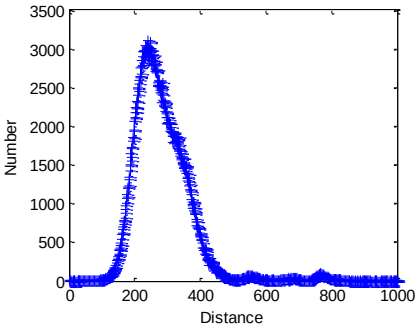

Figure 5. Distribution of Hash distance.

To evaluate performance of the proposed hashing scheme, we calculate the collision probability P_C. It means that two different images have similar hash values with the distance less than a threshold, as shown in formula (2).

$$P_C = \frac{\text{Number of different images judged as similar images}}{\text{Total number of different images}} \quad (2)$$

Table 3 shows the collision probabilities of different methods [2], [3], [6], [7] and our method. Hamming distance is used in [2] and [3]. Euclidean distance is used in [6] and [7]. So the thresholds used in these articles are different. We can see that the proposed method of collision probability is lower than other methods.

Table 3. Collision probability of different methods.

Hash methods	Threshold	*PC*
Method in [2]	35	8.67×10^{-4}
Method in [3]	60	1.14×10^{-4}
Method in [6]	3200	1.08×10^{-4}
Method in [7]	40	1.1×10^{-4}
The proposed method	100	1.02×10^{-5}

Tampering Detection Test

Table 4. Hash distance between original and forged images.

Original Image	Tampering Image	Image Size	Distance	Original Image	Tampering Image	Image Size	Distance
		640×480	141.97			500×375	214.2
		600×399	251.2			1024×700	202.7
		485×337	183.3			289×432	333.5

The proposed method can be used to detect whether an image is tampered or not. Table 4 gives some examples, with original and forged images, image size, and the distances. The forged versions of these six images were produced with Photoshop. We can see that the distances of these image pairs are bigger than the threshold 100.

Summary

In this paper, a new robust and secure image hashing method is proposed. This method uses color, texture and shape features of the selected effective blocks. Simulation results show that this scheme is robust against JPEG compression, additive noise, watermark embedding, scaling, brightness, gamma correction, and Gaussian filtering. Hashes between a pair of different images have lower collision probability compared with other methods. Image forgery involving structural modifications can be detected.

Acknowledgment

This paper is supported by the Natural Science Foundation of Shanghai (15ZR1418500, 15ZR1418400).

References

[1] V. Monga, A. Banerjee, and B. L. Evans, "A clustering based approach to perceptual image hashing," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 1, pp. 68-79, Mar. 2006.

[2] Y Zhao, S Z Wang, G R Feng, and Z J Tang, "A Robust Image Hashing Method Based on Zernike Moments". Journal of Computational Information Systems, 2010, 6(3), pp: 717-725.

[3] Z. Tang, S. Wang, X. Zhang, W. Wei, and S. Su, "Robust image hashing for tamper detection using non-negative matrix factorization," *Journal of Ubiquitous Convergence and Technology*, vol. 2, no. 1, pp. 18-26, May 2008.

[4] Z. Tang, X. Zhang, X. Li, and S. Zhang, Robust Image Hashing With Ring Partition and Invariant Vector Distance, *IEEE Transactions on Information Forensics and Security*, 2016, 11(1):200-214.

[5] C. Qin, C. Chang and P. Tsou, Robust image hashing using non-uniform sampling in discrete Fourier domain, *Digital Signal Processing*, 2013, 23(2):578-585.

[6] V. Monga and M. K. Mihcak, "Robust and secure image hashing via non-negative matrix factorizations," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 3, pp. 376-390, Sep. 2007.

[7] Y Zhao, S Z Wang, H Yao, and W Wu, "Perpetual Image Hashing Based on Conformal Mapping and Zernike Moments", Jounal of Applied Sciences, 2012, 30(1), pp: 75-81.

[8] Z. Tang, S. Wang, X. Zhang, W. Wei, and Y. Zhao, "Lexicographical framework for image hashing with implementation based on DCT and NMF," *Multimedia Tools and Applications*, vol. 52, Issue 2-3, pp. 325-345, 2011.

[9] C. Yan, C. Pun and X. Yuan, Multi-scale image hashing using adaptive local feature extraction for robust tampering detection, *Signal Processing*, 2016, 121:1-16.

[10] Y. Zhao, M. Tong, Q. Zhao and H. Bian, Robust Image Hashing Based on Salient Region, *Journal of Computational Information Systems*, 2014, 10(15):6777-6784.

[11] Y. Zhao, S. Wang, X. Zhang and H. Yao, Robust Hashing for Image Authentication Using Zernike Moments and Local Features, *IEEE Transactions on Information Forensics and Security*, 2013, 8(1): 55-63.

[12] F. A. P, Petitcolas (2000) Watermarking schemes evaluation, IEEE Signal Processing Magazine,17(5): 58-64.