Robustness and Discrimination Oriented Hashing Combining Texture and Invariant Vector Distance

Ziqing Huang
School of Computer Science and Technology, Tianjin University
Tianjin, China
skyhuangzq@163.com

Shiguang Liu*
School of Computer Science and Technology, Tianjin University
Tianjin, China
lsg@tju.edu.cn

ABSTRACT

Image hashing is a novel technology of multimedia processing with wide applications. Robustness and discrimination are two of the most important objectives of image hashing. Different from existing hashing methods without a good balance with respect to robustness and discrimination, which largely restrict the application in image retrieval and copy detection, i.e., seriously reducing the retrieval accuracy of similar images, we propose a new hashing method which can preserve two kinds of complementary features (global feature via texture and local feature via DCT coefficients) to achieve a good balance between robustness and discrimination. Specifically, the statistical characteristics in gray-level co-occurrence matrix (GLCM) are extracted to well reveal the texture changes of an image, which is of great benefit to improve the perceptual robustness. Then, the normalized image is divided into image blocks, and the dominant DCT coefficients in the first row/column are selected to form a feature matrix. The Euclidean distance between vectors of the feature matrix is invariant to commonly-used digital operations, which helps make hash more compact. Various experiments show that our approach achieves a better balance between robustness and discrimination than the state-of-the-art algorithms.

CCS CONCEPTS

• **Computing methodologies** → **Image representations**; **Image compression**;

KEYWORDS

Image hashing, Robustness and discrimination, Dominant DCT coefficients, Invariant vector distance

ACM Reference Format:
Ziqing Huang and Shiguang Liu. 2018. Robustness and Discrimination Oriented Hashing Combining Texture and Invariant Vector Distance. In *2018 ACM Multimedia Conference (MM '18), October 22–26, 2018, Seoul, Republic of Korea,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3240508.3240690

*This is the corresponding author

MM '18, October 22–26, 2018, Seoul, Republic of Korea

ACM ISBN 978-1-4503-5665-7/18/10…$15.00
https://doi.org/10.1145/3240508.3240690

1 INTRODUCTION

Image hashing [7, 17, 27, 34] is a frontier research topic in the field of multimedia content security and management [15, 19]. It aims at encoding any size image to a content-based compact sequence called hash by a hash function [10, 13, 20], while preserving some notion characteristics of the original image. With the compact hash codes, it has been successfully applied to copy detection [3, 39], image retrieval [12, 18], image authentication [35, 37], and image forensic [2, 16] under less time and storage cost. Note that, in many practical applications, the images always undergo some content-preserving operations, such as JPEG compression, brightness/contrast adjustment, or geometric transformation. These operations will inevitably alter their digital representations, but would not affect their visual contents. Therefore, the content-based hash should be kept unchanged after these operations. This is a property of hash function called perceptual robustness [1, 34], i.e., hashes of an image and its processed versions are expected to be the same or very similar. The other property, discrimination [32], requires that those images with different visual contents should have different image hashes.

Unfortunately, perceptual robustness and discrimination contradict with each other, and improvement of perceptual robustness will lead to decrease of discrimination and vice versa. A common phenomenon is that most hashing algorithms can only reach a single robustness or discrimination, which greatly restricts its application. Different from previous hashing methods, we innovatively combine the global feature via texture and the local feature via dominant DCT coefficients to achieve a good balance between robustness and discrimination.

1.1 Related Work

Previous image hashing methods mainly include matrix theory based hashing [4, 11, 22], manifold learning based hashing [29, 31, 38], and frequency domain based hashing [14, 21, 36].

Matrix theory based hashing methods consider an image as a matrix and extract the eigenvalues or eigenvectors by Singular Value Decomposition (SVD) or Nonnegative Matrix Factorization (NMF) to generate hash codes. Kozat et al. [11] first introduced SVD into image hashing, which achieved

good robustness to image rotation, but the promotion space is relatively large for discrimination. Monga and Mihcak [22] applied NMF to replace SVD, used the combination coefficients in the matrix factorization to construct a secondary image, obtained its low-rank matrix approximation by using NMF again, and concatenated the matrix entries to form the NMF-NMF vector. This method outperforms the SVD hashing, but is sensitive to watermark embedding. However, limited to the constraints of NMF, Chen et al. [4] further embraced Semi-NMF [33] since it could hold broader issues with mixed signs such as CNN or zero-centered features. Overall, these methods are robust to geometric attacks (rotation, translation, and scaling), but the discrimination is still less satisfactory.

Manifold learning based methods utilize Locally Linear Embedding (LLE), Principal Component Analysis (PCA), or Multidimensional Scaling (MDS), to explore data distribution and preserve local similarity structures from the high dimensional data. Tang et al. [31] investigated the use of LLE in image hashing and found that embedding vector variances of LLE are approximately linearly changed by content-preserving operations, which made it highly robust. But it is sensitive to the geometric attacks, and the discrimination is unsatisfied. Zhu et al. [38] took PCA to learn hash function by using the probability representations of all representative data points. They first partitioned the whole data set into groups, generated the probability representations of all the data points, while also preserved the neighborhood of the data. It is not difficult to find out these methods tend to focus on the local structural similarity and neglect the effect of global features on hashing performance, which is hard to reach a good balance between robustness and discrimination.

Compared with the above two kinds of methods, the most advantage of frequency domain based hashing methods is that it can freely choose different coefficients to represent image features so as to construct hash by Discrete Wavelet Transform (DWT), Discrete Cosine Transform (DCT), and Fourier Transform. In detail, these methods attempt to separate the high-frequency coefficients from the low-frequency coefficients so that easily study hash function by analyzing frequency distribution, statistical features and the histogram shape of the different coefficients, which can achieve a better balance between robustness and discrimination than other methods. For example, Venkatesan et al. [34] explored statistics of wavelet coefficients to design hash. This method is resilient to JPEG compression, median filtering and rotation within 2°. Lin and Chang [14] designed an image authentication system with robust hashing, which is based on invariant relations between DCT coefficients at the same position in separate blocks. Yan et al. [36] used quaternion DFT to achieve good hashing performance and can also detect the tamper location. This paper also performs some research on frequency domain and study the distance characteristics of dominant DCT coefficients. In contrast, we innovatively combine coefficient features with image texture to make up for the local limitation of frequency domain coefficients so as to achieve better results.

1.2 Contribution

We propose a new hashing that observably improves the robustness and discrimination with the combination of texture and frequency domain coefficients. Specifically, we first leverage the statistical characteristics in GLCM to reveal the texture changes of an image, which is of great benefit to improve the perceptual robustness. We further introduce the local feature through dividing image into blocks, while DCT is carried out to obtain the dominant coefficients in first row/column for each block. We show that the Euclidean distance between vectors of dominant coefficients is invariant to commonly-used digital operations. Extensive experiments indicate our method can achieve superior performance over the state-of-the-art algorithms. The main contributions of our method are summarized as follows:

(1) The local and global features of an image provide complimentary information to each other so as to achieve optimal performance. This indicates that it is reasonable for us to combine these two kinds of complementary structures to realize a good balance between robustness and discrimination.

(2) We find that the DCT coefficients in other positions can be calculated by the first row/column coefficients. This allows us to reflect the changes of the whole block pixels by only choosing the coefficients in the first row/column, which greatly reduces the time complexity. In addition, we reveal that the Euclidean distance between vectors of dominant coefficients is invariant to commonly-used digital operations, which is beneficial to construct a more robust hash coding (Section 2.2.2 and Section 3.2).

(3) We conduct experiments with popular database to validate efficiency of our method. The results demonstrate that our hashing is robust against commonly-used digital operations, and has good discriminative capability. ROC curve comparisons show that our hashing exhibits superior perceptual robustness and discriminative capability over existing popular methods.

2 PROPOSED IMAGE HASHING

The purpose of our method is to seek a hash function $H(\mathbf{I})$, which maps an image to the short sequence $\mathbf{h} = H(\mathbf{I})$, so that it is able to represent the original image applied in copy detection, image retrieval, image quality assessment, etc. In general, we define that $\mathbf{I}_1$ is a visually similar content version of $\mathbf{I}$, and $\mathbf{I}_2$ is a different image. Let $P(\cdot)$ be the probability of event occurrence, and $r(\cdot,\cdot)$ is the metric function to judge the similarity of two hashes. For a given threshold T and a small positive number ε close to 0, the hash function $H(\mathbf{I})$ should satisfy the following two basic properties.

(1) Robustness. It is hoped that the hash of the normally processed image remains unchanged, i.e., for two images with visually similar content, whether their internal data are consistent or not, the same probability of hashes should be close to 1. In this paper, the correlation coefficient is chosen as the metric function. If $r(\cdot,\cdot) \geq T$, the two images are viewed as similar images, which can be derived from the

following formula.

$$P\{r[H(\mathbf{I}_1), H(\mathbf{I})] \geq T\} \geq 1 - \varepsilon \quad (1)$$

(2) Discrimination. It implies two images with different contents should have completely different hashes, i.e., the different probability of hashes should be close to 1.

$$P\{r[H(\mathbf{I}_1), H(\mathbf{I})] < T\} \geq 1 - \varepsilon \quad (2)$$

Based on the above-mentioned requirements, we adopt the new strategy that simultaneously preserve two kinds of complementary features (global feature via texture and local feature via DCT coefficients) to achieve a good balance between robustness and discrimination. Specifically, the input image is converted to a normalized image by some operations for robust feature extraction, including bilinear interpolation, Gaussian filtering and color space conversion. Next, the normalized image is processed with two operations separately. One is to extract the texture information by GLCM, and the other is to obtain the invariant Euclidean distance between vectors of dominant DCT coefficients. Finally, the features of texture and invariant vector distance are integrated into a one-dimensional vector, the elements of which are quantized for reducing the storage cost.

2.1 Preprocessing

The original image is resized to a standard size $S \times S$ by bilinear interpolation for getting the fixed-length hash. In order to reduce the influence of the slight noise on the image quality, the Gaussian low-pass filtering is considered to smooth the standard image. However, it is known that the R, G, B components in RGB color system are highly correlated, which leads to the uniformity of RGB color system is very poor, and the perceptual difference cannot be expressed as the distance between two colors. It is worth noting that the HSI color system is closer to people's experience and color perception than the RGB color system. Therefore, we take nonlinear transformation to convert an image from RGB color system to HSI color system, and extract intensity component as the input image for subsequent feature extraction.

2.2 Feature Extraction

2.2.1 Global Statistical Feature. Texture is a visual feature which reflects the comprehensive information with gray-level statistic, spatial distribution and structure of an image. It is a set of pixels with a certain shape and size, which is a global inherent characteristic of almost all image surfaces. GLCM [25] characterizes the texture by calculating how often pairs of pixels with specific values and with a specified spatial relationship occur in an image, creating a co-occurrence matrix, and then extracting the statistical features. Let $P(x, y|d, \theta)$ express the number of times that the pixel with value x occurred at d distance and θ direction to a pixel with value y. When d and θ are determined, $P(x, y|d, \theta)$ is shown by $P(x, y)$. Four statistical features, i.e., contrast, correlation, energy, and homogeneity, are considered from the co-occurrence matrix with the intent to describe the texture feature.

(1) Contrast is a measure of intensity contrast between a pixel and its neighbor over the whole image. Contrast is 0 for a constant image.

$$contrast = \sum_x \sum_y |x - y|^2 p(x, y) \quad (3)$$

(2) Correlation returns a measure of how correlated a pixel is to its neighbor over the whole image. Its range is [-1, 1].

$$correlation = \sum_x \sum_y \frac{(x - u_x)(y - u_y)p(x, y)}{\sigma_x \sigma_y} \quad (4)$$

where u_x and u_y are the GLCM mean of the first and the second components, σ_x and σ_y are the GLCM variance of the first and the second components.

(3) Energy calculates the sum of squared elements in the GLCM and reflects the distribution of image gray-scale uniformity of weight and texture.

$$energy = \sum_x \sum_y p(x, y)^2 \quad (5)$$

(4) Homogeneity evaluates the closeness of the distribution of elements in the GLCM to the GLCM diagonal, and illustrates local changes in image texture.

$$homogeneity = \sum_x \sum_y \frac{1}{1 + |x - y|^2} p(x, y) \quad (6)$$

The texture features are computed for the intensity component when d=1 and $\theta = 0^\circ, 45^\circ, 90^\circ, 135^\circ$. In each direction four texture features are calculated. Therefore, we obtain texture features $\mathbf{T}$ with the length of 16.

2.2.2 Local Invariant Feature. The intensity component is firstly divided into $s \times s$ non-overlapping blocks, where S is the integral multiple of s for simplicity. Thus, the total number of blocks is $N = (S/s)^2$. Let B_i be the $i - th$ block indexed from left to right and top to bottom ($1 \leq i \leq N$), and $B_i(j, k)$ be the value in the $(j + 1) - th$ row and $(k + 1) - th$ column of B_i. For image block B_i, a 2D-DCT is applied. Here, those dominant DCT coefficients in the first row/column are extracted to construct feature matrix. This is because these DCT coefficients can fully indicate the changes of the whole block pixels. Specifically, the DCT coefficients in the first row $C_i(0, v)$ and the first column $C_i(u, 0)$ are calculated by the following equations.

$$C_i(0, v) = \frac{\sqrt{2}}{s} \sum_{j=0}^{s-1} \sum_{k=0}^{s-1} B_i(j, k) \cos\left[\frac{(2k + 1)v\pi}{2s}\right] \quad (7)$$

$$C_i(u, 0) = \frac{\sqrt{2}}{s} \sum_{j=0}^{s-1} \sum_{k=0}^{s-1} B_i(j, k) \cos\left[\frac{(2j + 1)u\pi}{2s}\right] \quad (8)$$

For the sequence $C_i(0, v)$ and $C_i(u, 0)$, those coefficients at the end of $C_i(0, v)$ and $C_i(u, 0)$ are easily influenced by JPEG compression and noise. Therefore, the coefficients from the second element to the $(n + 1) - th$ element ($n = s/2$) are used to form a vector $\mathbf{Q}_i = [C_i(0, 1), C_i(0, 2), \cdots, C_i(0, n),$

$C_i(1,0), C_i(2,0), \cdots, C_i(n,0)]^T$. Thus, a DCT-based feature matrix $\mathbf{Q}$ is available as follows:

$$\mathbf{Q} = [\mathbf{Q}_1, \mathbf{Q}_2, \cdots, \mathbf{Q}_N], \tag{9}$$

Next, data normalization [29] is applied to each row of $\mathbf{Q}$, and the normalization result of $\mathbf{Q}$ is denoted as $\mathbf{U}$. A reference vector $\mathbf{U}_0 = [u_0(1), u_0(2), \cdots, u_0(2n)]^T$ is generated for distance calculation, where $u_0(l)(1 \le l \le 2n)$ is the mean from each row of $\mathbf{U}$. Let $\mathbf{U}_i = [u_i(1), u_i(2), \cdots, u_i(2n)]^T$ be the $i-th$ column of $\mathbf{U}$. The Euclidean distance $d(i)$ between $\mathbf{U}_i$ and $\mathbf{U}_0$ is calculated by.

$$d(i) = \sqrt{\sum_{l=1}^{2n} (u_i(l) - u_0(l))^2}, \tag{10}$$

As a result, the invariant Euclidean distance between dominant DCT coefficients is obtained to improve the hashing performance, represented as $\mathbf{D}$=[$d(1)$, $d(2)$, $\cdots$, $d(N)$].

2.3 Quantization

The intensity component is carried out by the operations of GLCM and dominant DCT coefficient distance, respectively. Therefore, the proposed hash can be generated with the texture feature $\mathbf{T}$ with a length of 16, and the dominant DCT coefficient distance feature $\mathbf{D}$ with a length of N, described as $\mathbf{H}_1 = [\mathbf{T}\ \mathbf{D}]$. In this paper, the standard size S is 512, and the block size s is 64 for getting the number of blocks $N = (512/64)^2 = 64$. As has been explained, the total length of the hash $\mathbf{H}_1$ is 80. However, a decimal requires many bits for storage, such as 32 bits for float number or 64 bits for double number. To reduce storage, the element $H_1(q)$ $(1 \le q \le 80)$ in $\mathbf{H}_1$ are quantized to an integer $h(q)$ as follows:

$$h(q) = round[H_1(q) \times 10 + 0.5] \tag{11}$$

Consequently, the image hash $\mathbf{h}$ is available as follows.

$$\mathbf{h} = [h(1), h(2), ..., h(80)] \tag{12}$$

To determine the hash length in binary form, hashes of 1338 color images [26] are taken as data source. Since every hash sequence contains 80 integers, there are $80 \times 1338 = 107040$ hash elements in total. It is found that the minimum and the maximum value are 1 and 300, respectively. As 9 bits can represent integers ranging from 0 to $2^9 - 1 = 511$, each hash element requires 9 bits for storage at most. Therefore, our hash length in binary form is $9 \times 80 = 720$ bits. As a reference, testing on the same 1338 different image, the hash length in binary form of MDS-based hashing [29], RT-DCT hashing [23], and NMF-NMF-SQ hashing [22] are 900 bits, 240 bits, and 960 bits, respectively.

3 EXPERIMENTS

3.1 Experimental Settings and Datasets

The used parameters in our experiments are as follows. The original image is resized to 512×512 and the convolution mask in Gaussian low-pass filtering is 3×3. The size of non-overlapping image block is 64×64, then DCT coefficients from the second element to the $33-th$ element in the first row/column of each block are taken to form the feature matrix, i.e., S=512, s=64, $N = (S/s)^2 = 64$, $n = s/2 = 32$. The correlation coefficient is exploited to evaluate similarity between two image hashes. The higher correlation coefficient is, the more similar the corresponding images usually are. We conduct comprehensive experiments on three datasets, namely standard benchmark image, Copydays database [9] and UCID [26], and compare our proposed method with several state-of-the-art hashing methods.

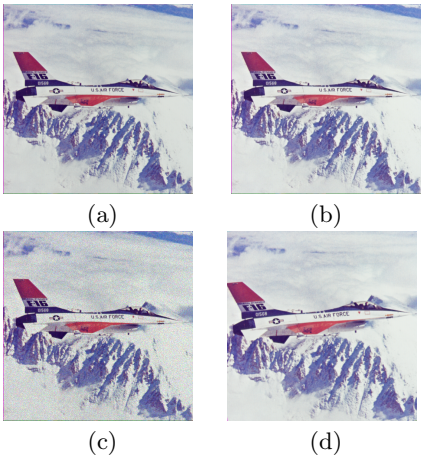
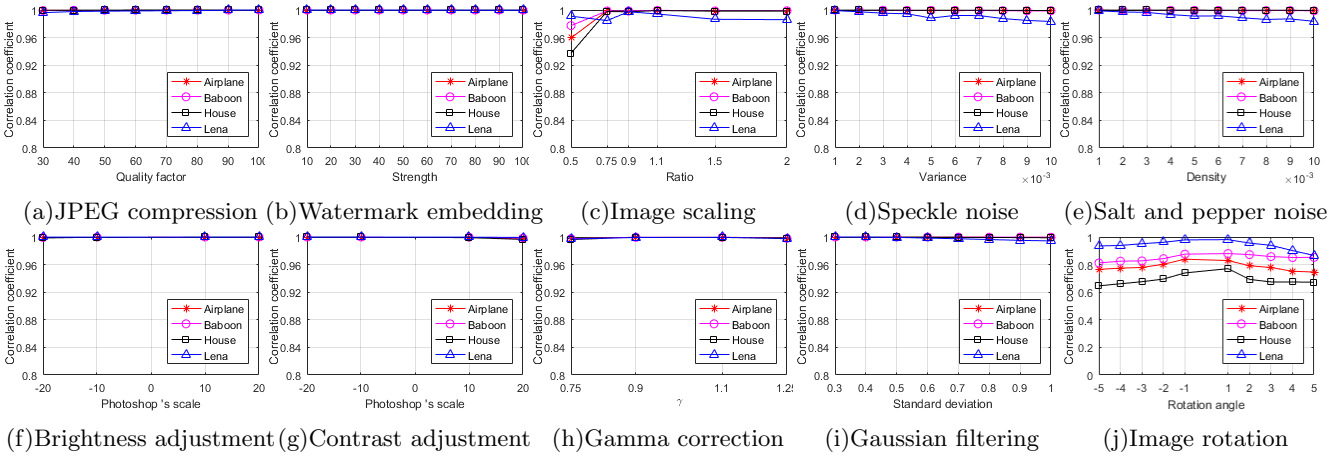


Figure 1: An example of visually similar images.

Standard benchmark image consists of 4 images with size 512×512, which are Airplane, Baboon, House, and Lena. To produce visually similar images, we exploit the well-known tools, i.e., Photoshop, MATLAB and StirMark 4.0 [24], to perform robustness attacks. The adopted content-preserving operations include: JPEG compression (parameters: 30, 40, $\cdots$, 100), watermark embedding (parameters:10, 20, $\cdots$, 100), image scaling (parameter: 0.5, 0.75, 0.9, 1.1, 1.5, 2.0), speckle noise (parameters: 0.001, 0.002, $\cdots$, 0.01), salt and pepper noise (parameters: 0.001, 0.002, $\cdots$, 0.01), brightness adjustment (parameters: $\pm$10, $\pm$20), contrast adjustment (parameters: $\pm$10, $\pm$20), gamma correction (parameters: 0.75,0.9,1.1,1.25), Gaussian filtering (parameters: 0.3, 0.4,$\cdots$, 1.0), and image rotation (parameters: $\pm$1, $\pm$2, $\pm$3, $\pm$4, $\pm$5). Clearly 74 different content-preserving operations are implemented, and every test image has 74 similar versions and thus the number of the visually similar images is $4 \times 74 = 296$. Figure 1 presents a group of visually similar images, where (a) is the original image named Airplane, (b) is the JPEG compression image with the quality factor 100, (c) is the Speckle noise image with variance 0.01, and (d) is the rotation image with 5°. Obviously, the specific data of these images are different, but their visual perception is similar.

Copydays database contains 157 color images, whose pixel resolution range from 1200×1600 to 3008×2000. All content-preserving operations mentioned above are taken to attack these 157 color images and $157 \times 74 = 11618$ pairs of visually similar images are obtained.



(a)JPEG compression (b)Watermark embedding (c)Image scaling (d)Speckle noise (e)Salt and pepper noise

(f)Brightness adjustment (g)Contrast adjustment (h)Gamma correction (i)Gaussian filtering (j)Image rotation

Figure 2: Correlation coefficient results under specific digital operations.

UCID is used to validate the discrimination of our hashing. The UCID contains 1338 different color images, whose pixel resolution are 512×384 or 384×512.

3.2 Performance of Robustness

We first use the standard benchmark images and its visually similar images to test the hashing robustness. Specifically, we extract hashes of these test images and their similar versions, evaluate their similarity with correlation coefficient. In theory, the value of correlation coefficients between test images and their similar versions should be infinitely close to 1. As has been explained, Figure 2 presents robustness results under different operations, where the x-axis is the specific parameter of digital operations and the y-axis is the correlation coefficient. It is observed from Figs. 2 (a)$\sim$(i) that all the correlation coefficients are close to 1, which means that our hashing algorithm can effectively identify visually similar images and has good robustness. Fig. 2 (j) represents the correlation coefficients under rotation operations, in which the data are relatively small. The rotation operation will inevitably increase the image size, in fact, those images for Fig. 2 (j) undergo three manipulations, i.e., rotation, cropping and rescaling, which will introduce more distortions. That is why most correlation coefficients of rotation are smaller than those of other single operation. In spite of this, our hashing method is still robust to most digital operations under the appropriate threshold.

In addition, for the every graph in Figure 2, it is not difficult to find that the correlation coefficients do not fluctuate greatly under different parameters of the same digital attack. For example, in Fig. 2 (j), there are no obvious jumps on each line at different rotation angles, which also proves that the design theory of this paper is correct and reasonable. That is, the Euclidean distance between vectors of dominant DCT coefficients is basically invariant to commonly-used digital operations, which helps make hashing more compact and robust. In order to further verify the perceptual robustness of the proposed hashing and determine the optimal thresholds for applications, a large open database called Copydays database is selected. Those hashes of the original and 11618 pairs of visually similar images are extracted, correlation coefficients between each pair of hashes are calculated, and statistics of correlation coefficients under different digital operations are computed. The results are shown in Table 1. It is found that the means of all digital operations are bigger than 0.92. In particular, when T=0.92, 95.42% visually similar images (including rotated versions) can be correctly detected. And further, if there is no rotated similar image, the true positive rate (P_{TPR}) will reach 99.98%. All these show that our hashing method can correctly identify the visually similar images and have a higher P_{TPR}.

Table 1: Statistics of Correlation Coefficient under Different Operations

Operation	Maximum	Minimum	Mean
JPEG compression	1	0.9690	0.9983
Watermark embedding	1	0.9612	0.9983
Image scaling	1	0.8967	0.9978
Speckle noise	1	0.9455	0.9984
Salt and Pepper noise	1	0.9541	0.9988
Brightness adjustment	1	0.9603	0.9975
Contrast adjustment	1	0.9685	0.9987
Gamma correction	0.9999	0.9514	0.9964
Gaussian filtering	1	0.9707	0.9991
Image rotation	0.9960	0.4052	0.9269

3.3 Performance of Discrimination

The open image database called UCID is used to validate the discrimination of our hashing. The UCID contains 1338 color images with completely different visual content from each other. Those hashes of these 1338 color images are firstly extracted. The $1338 \times (1338-1)/2 = 894453$ correlation

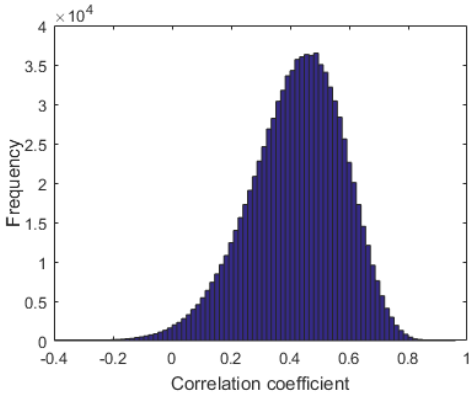


Figure 3: Distribution of correlation coefficients between hashes of different images.

Table 2: P_{FPR} and P_{TPR} under Different Thresholds

Threshold	P_{FPR} (UCID)	P_{TPR} (Copydays database)
T=0.90	1.34×10^{-5}	96.81%
T=0.91	8.94×10^{-6}	96.11%
T=0.92	5.59×10^{-6}	95.42%
T=0.93	4.47×10^{-6}	94.53%
T=0.94	3.35×10^{-6}	93.52%
T=0.95	2.24×10^{-6}	92.41%
T=0.96	0	91.19%

coefficients between each pair of these hashes are calculated, and also the frequency distribution of these correlation coefficients is presented in Figure 3, where the x-axis represents the correlation coefficient, and the y-axis is the frequency. As can be seen from Figure 3, the correlation coefficients are mainly concentrated in the interval [0.2, 0.6], indicating that our hashing can effectively identify different images. Moreover, the statistics, i.e., minimum, maximum, mean, and standard deviation of these correlation coefficients are -0.3994, 0.9597, 0.4215, and 0.1579, respectively. As shown in Table 1, the minimum mean of correlation coefficient for visual similar image is 0.9269, which is far greater than the mean 0.4215 of correlation coefficients for different images. This illustrates that our hashing has lower false positive rate (P_{FPR}) and good discrimination under the desirable robustness, and reaches better balance between them. Table 2 illustrates the P_{FPR} and P_{TPR} under different thresholds. Clearly, for the similarity metric with correlation coefficient, the bigger the T value is, the smaller the P_{FPR} is, i.e., the better the discriminative capability. However, as T increases, the robustness performance (P_{TPR}) will be declined. Therefore, we should choose T in terms of specific applications to give a satisfactory balance between discrimination and robustness.

3.4 Impact of Image Block Size on Hashing Performance

In this section, we will discuss the impact of image block size on hashing performance in Receiver Operating Characteristics (ROC). The ROC [6] is exploited to make visual balance comparisons with respect to robustness and discrimination. In the ROC, the x-axis is defined as P_{FPR} and the y-axis is the P_{TPR}, which can be determined by the following equations.

$$P_{\text{FPR}}(r \geq T) = \frac{n_1}{N_1} \tag{13}$$

$$P_{\text{TPR}}(r \geq T) = \frac{n_2}{N_2} \tag{14}$$

where n_1 is the number of pairs of different images mistakenly considered as similar images, N_1 is the total pairs of different images, n_2 is the number of pairs of visually similar images correctly identified as the similar images, and N_2 is the total pairs of visually similar images. It is obvious that P_{FPR} and P_{TPR} can indicate discrimination and robustness, respectively. A small P_{FPR} represents good discrimination, and a big P_{TPR} means good robustness. Note that ROC curve is obtained by varying threshold T to generate a set of points (P_{FPR}, P_{TPR}). The ROC curve near the top-left corner implies a small P_{FPR} and a big P_{TPR}, and shows better balance than those curves far away from the top-left corner.

In the experiment, we only alter the image block size and keep other parameters invariant. The used block size settings are 16×16, 32×32, 64×64 and 128×128. The test image databases are also adopted. i.e., 11618 pairs of similar images for robustness and 894453 pairs of different images for discrimination. For each block size, we use our hashing algorithm to extract hashes, calculate correlation coefficients between hashes, design different thresholds to find their P_{FPR} and P_{TPR}, and then obtain the ROC curve. Figure 4 is the ROC curve comparisons under different block sizes. It can be seen that all ROC curves are close to the top-left corner, which illustrates that the size of the image block has little effect on hashing performance. In particular, the ROC curves near the top-left corner are enlarged and presented in the right-bottom part of Figure 4. Clearly, the ROC curve of

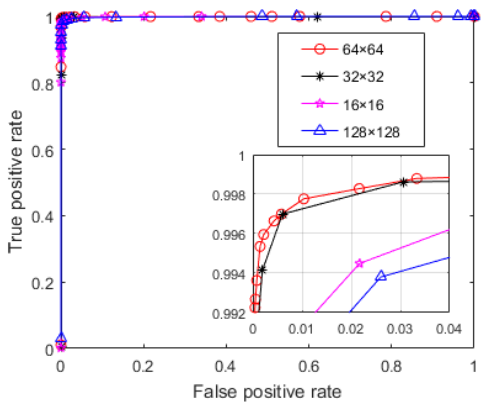


Figure 4: ROC curves under different block sizes.

Table 3: Performance Comparisons under Different Block Sizes

Block size	AUC	Hash length	Time (s)
16×16	0.9994	1040 digits (9360 bits)	0.161
32×32	0.9995	272 digits (2448 bits)	0.118
64×64	0.9997	80 digits (720 bits)	0.106
128×128	0.9989	32 digits (288 bits)	0.102

64×64 is closer to the top-left corner than the curves of other block sizes. Therefore, it is intuitively concluded that the block size 64×64 has better balance performance than the other block sizes. To make quantitative analysis, the Area under the ROC Curve (AUC) [6] is calculated. Note that the maximum value of AUC is 1. The bigger the AUC is, the better balance between robustness and discrimination. The AUCs of 16×16, 32×32, 64×64 and 128×128 are 0.9994, 0.9995, 0.9997 and 0.9989, respectively. This means that performance of 64×64 is better than that those of 16×16, 32×32 and 128×128. This can be understood as follows. A small block size means more features in the hash, which will help to robust the content-preserving operations, but the discrimination will slightly decrease. Therefore, for 512×512 images, a moderate block size, e.g., 64×64, is a good choice for keeping a better balance between robustness and discrimination.

The average running time and hash length comparisons under different block sizes are also conducted. For each block size, we apply our hashing to extract hashes of 1338 different images used in the discrimination test, record the total running time, and calculate the average time for computing one image. Our hashing is implemented with MATLAB 2016a, running on a desktop PC with 3.6 GHz IntelCore i7-7700 CPU and 8.0 GB RAM. The average time of 16×16, 32×32, 64×64 and 128×128 are 0.161, 0.118, 0.106, and 0.102 seconds, respectively. Obviously, as the image block size becomes large, the average running time slightly decreases. This is because a bigger block size means less calculations needed in extracting image features. Moreover, the hash length is closely related to the block size. The bigger the block size is, the shorter the hash length. Performance comparisons under different block sizes are summarized in Table 3.

3.5 Performance Comparison among Different Methods

We list the details of the comparison methods as follows:

(1) **DCT-LLE hashing** [30] extracted the DCT coefficients from CVA, and then used LLE to construct hash function. For this method, all parameter settings are consistent with the original paper to get the best hashing performance. Consequently, the hash lengths of it are 64 bits, and adopts Hamming distance to evaluate similarity between two hashes.

(2) **CVA-Canny hashing** [28] investigated the use of CVA and Canny operator in image hashing and achieved to resist to any-angle rotation. For this method, the normalized image size is 512×512, the number of concentric circles is 40, and the threshold for error control is 3. The hash lengths are 40 integers. Correlation coefficient is the similarity metric.

(3) **GF-LVQ hashing** [13] incorporated random Gabor Filtering (GF) that describes texture and dithered lattice vector quantization (LVQ) to design image hashing. The normalized image size is 512×512 and other parameter values are the same with original paper. Accordingly, the hash lengths of this method are 120 bits, and the hash similarity metric is normalized Hamming distance.

(4) **MDS-based hashing** [29] used DFT and Log-polar transform (LPT) to get the feature matrix, and then MDS was conducted to generate hash. Its image size is 512×512, and the low-dimensional derived from high dimensional is 30, which means the hash lengths are 180 integers. Correlation coefficient is the hash similarity metric.

(5) **SVD-CSLBP hashing** [5] exploited SVD and Center-Symmetric Local Binary Patterns (CSLBP) to design image hashing for authentication. Its optimal parameters reported in the original paper are taken. Consequently, the hash lengths are 64 floats, and it chooses correlation coefficient as the hash similarity metric.

It is not a coincidence that we consider these algorithms to compare hashing performance. First, these algorithms pay attention to the local [5] and global [13, 28, 29] features of an image separately, or combine them together [30] to study hash function, which is very comparable to our method because we also consider local and global features to design hashing. Second, these methods comprehensively cover the main technologies used in current hashing algorithms, namely frequency domain hashing [13, 30], matrix hashing [5, 28] and manifold learning hashing [29, 30]. Next, we will quantitatively analyze the performance of different hashing methods. The similar images and different images are used again to validate the robustness and discrimination of the compared algorithms. We exploit the compared algorithms to extract hashes of those images, and evaluate hash similarity with the respective metric. For each method, we use a set of thresholds to calculate its ROC curve. Then, the ROC curve

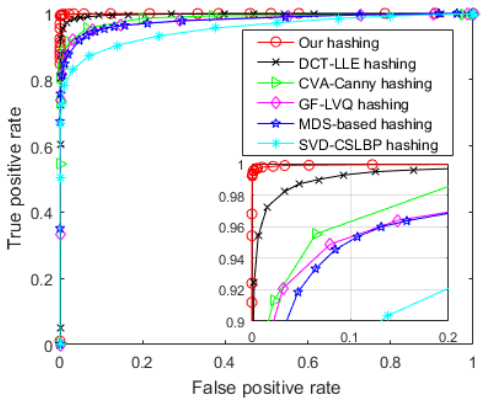


Figure 5: ROC curves among different algorithms.

Table 4: P_{TPR} Comparisons among Different Algorithms under Different Operations

Operation	P_{TPR} when $P_{\mathrm{FPR}} \approx 0$					
	Our hashing	DCT-LLE hashing	CVA-Canny hashing	GF-LVQ hashing	MDS-based hashing	SVD-CSLBP hashing
JPEG compression	1	0.2078	0.5422	0.1664	0.5924	0.1186
Watermark embedding	1	0.2968	0.5713	0.1401	0.6019	0.2255
Image scaling	0.9998	0.2622	0.5594	0.1285	0.5807	0.2527
Speckle noise	0.9996	0.4662	0.6713	0.4968	0.5701	0.4389
Salt and Pepper noise	1	0.3854	0.6497	0.4662	0.5739	0.3408
Brightness adjustment	1	0.4459	0.6831	0.4347	0.5892	0.3232
Contrast adjustment	1	0.4729	0.6688	0.4363	0.6099	0.5287
Gamma correction	1	0.4475	0.5669	0.4666	0.6178	0.2787
Gaussian filtering	1	0.5788	0.7396	0.3159	0.6576	0.5239
Image rotation	0.7268	0.0127	0.0191	0.2854	0.1650	0.2420

Table 5: Performance Comparisons among Different Hashing Algorithms

Performance	Our hashing	DCT-LLE hashing	CVA-Canny hashing	GF-LVQ hashing	MDS-based hashing	SVD-CSLBP hashing
AUC	0.9997	0.9971	0.9854	0.9794	0.9761	0.9527
Hash length	720 bits	64 bits	400 bits	120 bits	900 bits	64 floats
Time (s)	0.106	0.052	0.086	0.288	0.214	0.112

comparisons among different hashing algorithms are available, as shown in Figure 5. It is observed that our ROC curve is significantly above those of compared hashing algorithms. Therefore, it is concluded that our hashing has better balance performance than the compared algorithms. The AUCs of compared algorithms are also calculated, and the AUCs of DCT-LLE hashing, CVA-Canny hashing, GF-LVQ hashing, MDS-based hashing, SVD-CSLBP hashing and our hashing are 0.9971, 0.9854, 0.9794, 0.9761, 0.9527 and 0.9997, respectively. Clearly, our AUC is the largest in these compared algorithms, and our hashing outperforms the compared algorithms in balance performances with respect to robustness and discrimination. Meanwhile, we also find that the balance performance of DCT-LLE hashing is inferior to our method but better than other algorithms, which means the local and global features provide complimentary information to each other so that achieving optimal performance. This also proves that it is reasonable for our method to combine these two kinds of complementary structures together. Moreover, in the ROC graph, if some hashing methods have the same P_{FPR}, the methods with big P_{TPR} are better than those with small P_{TPR}. For each compared hashing method, we calculate the optimal P_{TPR} under different operations when $P_{\mathrm{FPR}} \approx 0$, and the results are presented in Table 4. It can be observed that our P_{TPR} all reach the maximum value 1, except image scaling, Speckle noise and image rotation. The P_{TPR} of image scaling, Speckle noise are 0.9998 and 0.9996 that are very close to 1. For image rotation, the P_{TPR} is 0.7268, but much bigger than those of the compared algorithms. As a result, compared with other hashing algorithms, our hashing method is more robust to common digital operations and can correctly identify visually similar images.

Moreover, computational time of the compared algorithms is also compared. The average time of DCT-LLE hashing, CVA-Canny hashing, GF-LVQ hashing, MDS-based hashing, and SVD-CSLBP hashing are 0.052, 0.086, 0.288, 0.214 and 0.112 seconds, respectively. Our average time is 0.106 seconds, which is faster than GF-LVQ hashing, MDS-based hashing and SVD-CSLBP hashing and slightly slower than DCT-LLE hashing, CVA-Canny hashing. Finally, Table 5 summarizes performance comparisons among different hashing algorithms. Our AUC is the largest one among the compared algorithms, illustrating that our balance performance is better than the compared algorithms. For the CVA-Canny and MDS-based methods, the hash elements of 1338 color images are taken to analyze the hash length in binary form. The hash length of CVA-Canny hashing is 40 integers, and maximum element is 689 that needs 10 bits (representing the number from $0\sim 2^{10}-1=1023$). Therefore, its hash length in binary form is $40\times 10=400$ bits. For MDS-based hashing, its length is 180 integers and each integer requires 5 bits, thus its hash length in binary form is $180\times 5=900$ bits. For the SVD-CSLBP hashing, according to the IEEE Standard [8], 32 bits at least are required for storing a float. Therefore, its hash length is $64\times 32=2048$ bits in binary form at least.

4 CONCLUSIONS

This paper innovatively combined coefficient features with global image texture to make up for the local limitation of frequency domain coefficients so as to achieve a good balance between robustness and discrimination. Moreover, another contribution of our work was the observation that those DCT coefficients in the first row/column could fully

indicated the changes of whole block pixels. The Euclidean distance between vectors of dominant DCT coefficients was invariant to commonly used operations, which made hash compact. Various experiments were conducted and the results demonstrated that the proposed hashing was robust to many content-preserving operations, and reached good discrimination. ROC curve comparisons indicated that our method was superior to the state-of-the-art algorithms in balance performances between robustness and discrimination. In particular, future research will attempt to focus on reducing hash length by the Huffman coding method, which will also improve the security.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their insightful comments. This work was supported in part by the Natural Science Foundation of China under grant nos. 61672375 and 61170118.

REFERENCES

[1] F. Balado, N. Hurley, E. Mccarthy, and G. Silvestre. 2007. Performance analysis of robust audio hashing. *IEEE Transactions on Information Forensics and Security* 2, 2 (2007), 254–266.
[2] S. Battiato, G. Farinella, E. Messina, and G. Puglisi. 2011. A robust forensic hash component for image alignment. In *Proceedings of International Conference on Image Analysis and Processing*. 473–483.
[3] S. Battiato, G. Farinella, E. Messina, and G. Puglisi. 2012. Robust image alignment for tampering detection. *IEEE Transactions on Information Forensics and Security* 7, 4 (2012), 1105–1117.
[4] Y. Chen, H. Zhang, X. Zhang, and R. Liu. 2017. Regularized seminonnegative matrix factorization for hashing. *IEEE Transactions on Multimedia* 26, 6 (2017), 1–13.
[5] R. Davarzani, S. Mozaffari, and K. Yaghmaie. 2016. Perceptual image hashing using center-symmetric local binary patterns. *Multimedia Tools and Applications* 75, 8 (2016), 4639–4667.
[6] T. Fawcett. 2006. An introduction to ROC analysis. *Pattern Recognition Letters* 27, 8 (2006), 861–874.
[7] E. Hassan, S. Chaudhury, and M. Gopal. 2012. Feature combination in kernel space for distance based image hashing. *IEEE Transactions on Multimedia* 14, 4 (2012), 1179–1195.
[8] IEEE. 2008. IEEE standard for floating-point arithmetic. *IEEE Std* (2008), 1–70.
[9] H. Jegou, F. Perronnin, M. Douze, J. Nchez, P. Perez, and C. Schmid. 2012. Aggregating local image descriptors into compact codes. *IEEE Transactions Pattern Analysis and Machine Intelligence* 34, 9 (2012), 1704–1716.
[10] F. Khelifi and J. Jiang. 2010. Perceptual image hashing based on virtual watermark detection. *IEEE Transactions on Image Processing* 19, 4 (2010), 981–994.
[11] S. Kozat, R. Venkatesan, and M. Mihcak. 2004. Robust perceptual image hashing via matrix invariants. In *Proceedings of IEEE International Conference on Image Processing*. 3443–3446.
[12] Y. Kuo, K. Chen, C. Chiang, and W. Hsu. 2009. Query expansion for hash-based image object retrieval. In *Proceedings of ACM International Conference on Multimedia*. 65–74.
[13] Y. Li, Z. Lu, C. Zhu, and X. Niu. 2012. Robust image hashing based on random gabor filtering and dithered lattice vector quantization. *IEEE Transactions on Image Processing* 21, 4 (2012), 1963–1980.
[14] C. Lin and S. Chang. 2001. A robust image authentication method distinguishing JPEG compression from malicious manipulation. *IEEE Transactions on Circuits and Systems for Video Technology* 11, 2 (2001), 153–168.
[15] Q. Liu, R. Safavi-Naini, and N. P. Sheppard. 2003. Digital rights management for content distribution. In *Proceedings of the Australasian Information Security Workshop on ACSW Frontiers*. 49–58.
[16] W. Lu and M. Wu. 2010. Multimedia forensic hash based on visual words. In *Proceedings of IEEE International Conference on Image Processing*. 989–992.
[17] X. Lv and Z. Wang. 2012. Perceptual image hashing based on shape contexts and local feature points. *IEEE Transactions on Information Forensics and Security* 7, 3 (2012), 1081–1093.
[18] A. Manno-Kovacs. 2016. Content based image retrieval using salient orientation histograms. In *Proceedings of IEEE International Conference on Image Processing*. 2480–2484.
[19] M. Mishra and F. L. M. C. Adhikary. 2013. Digital image tamper detection techniques - a comprehensive study. *Computer Science* 2, 1 (2013), 1–12.
[20] V. Monga, A. Banerjee, and B. L. Evans. 2006. A clustering based approach to perceptual image hashing. *IEEE Transactions on Information Forensics and Security* 1, 1 (2006), 68–79.
[21] V. Monga and B. Evans. 2006. Perceptual image hashing via a feature points: performance evaluation and tradeoffs. *IEEE Transactions on Image Processing* 15, 11 (2006), 3452–3465.
[22] V. Monga and M. Mihcak. 2007. Robust and secure image hashing via non-negative matrix factorizations. *IEEE Transactions on Information Forensics and Security* 2, 3 (2007), 376–390.
[23] Y. Ou and K. Rhee. 2010. A key-dependent secure image hashing scheme by using Radon transform. In *Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems*. 595–598.
[24] F. Petitcolas. 2000. Watermarking schemes evaluation. *IEEE Signal Processing Magazine* 17, 5 (2000), 58–64.
[25] A. Rampun, H. Strange, and R. Zwiggelaar. 2013. Texture segmentation using different orientations of GLCM features. In *Proceedings of International Conference on Computer Vision*. 1–8.
[26] G. Schaefer. 2003. UCID: an uncompressed color image database. In *Proceedings of Storage and Retrieval Methods and Applications for Multimedia*. 472–480.
[27] A. Swaminathan, Y. Mao, and M. Wu. 2006. Robust and secure image hashing. *IEEE Transactions on Information Forensics and Security* 1, 2 (2006), 215–230.
[28] Z. Tang, L. Huang, X. Zhang, and H. Lao. 2016. Robust image hashing based on color vector angle and Canny operator. *AEU - International Journal of Electronics and Communications* 70, 6 (2016), 833 – 841.
[29] Z. Tang, Z. Huang, X. Zhang, and H. Lao. 2017. Robust image hashing with multidimensional scaling. *Signal Processing* 137 (2017), 240–250.
[30] Z. Tang, H. Lao, X. Zhang, and K. Liu. 2016. Robust image hashing via DCT and LLE. *Computers & Security* 62 (2016), 133–148.
[31] Z. Tang, L. Ruan, C. Qin, X. Zhang, and C. Yu. 2015. Robust image hashing with embedding vector variance of LLE. *Digital Signal Processing* 43 (2015), 17–27.
[32] Z. Tang, X. Zhang, and S. Zhang. 2014. Robust perceptual image hashing based on ring partition and NMF. *IEEE Transactions on Knowledge and Data Engineering* 26, 3 (2014), 711–724.
[33] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. Schuller. 2014. A deep semi-NMF model for learning hidden representations. In *Proceedings of International Conference on Machine Learning*. 1692–1700.
[34] R. Venkatesan, S. Koon, M. Jakubowski, and P. Moulin. 2000. Robust image hashing. In *Proceedings of IEEE International Conference on Image Processing*. 664–666.
[35] X. Wang, K. Pang, X. Zhou, Y. Zhou, L. Li, and J. Xue. 2015. A visual model-based perceptual image hash for content authentication. *IEEE Transactions on Information Forensics and Security* 10, 7 (2015), 1336–1349.
[36] C. Yan, C. Pun, and X. Yuan. 2016. Quaternion-based image hashing for adaptive tampering localization. *IEEE Transactions on Information Forensics and Security* 11, 12 (2016), 2664–2677.
[37] Y. Zhao, S. Wang, X. Zhang, and H. Yao. 2013. Robust hashing for image authentication using zernike moments and local features. *IEEE Transactions on Information Forensics and Security* 8, 1 (2013), 55–63.
[38] X. Zhu, X. Li, S. Zhang, Z. Xu, L. Yu, and C. Wang. 2017. Graph PCA hashing for similarity search. *IEEE Transactions on Multimedia* 19, 8 (2017), 2033–2044.
[39] F. Zou, Y. Chen, J. Song, K. Zhou, Y. Yang, and N. Sebe. 2015. Compact image fingerprint via multiple kernel hashing. *IEEE Transactions on Multimedia* 17, 7 (2015), 1006–1018.