Robust Image Hashing Based on Multiple Histograms

Zhenjun Tang, Liyan Huang, Yumin Dai, and Fan Yang
Department of Computer Science, Guangxi Normal University, Guilin 541004, P. R. China
tangzj230@163.com, {236675317, 376084989, 715223952}@qq.com
*Corresponding Author: tangzj230@163.com, zjtang@gxnu.edu.cn

Abstract

Image hashing is a novel technology of multimedia and finds many applications such as image retrieval, image copy detection, digital watermarking and image indexing. This paper proposes a multiple histograms based image hashing, which can reach an acceptable trade-off between rotation robustness and discrimination. The proposed hashing is done by converting the input image into a normalized image, dividing it into different rings, extracting ring-based histograms and compressing them by discrete wavelet transform. Hash similarity is evaluated by L2 norm. Experiments show that our hashing is robust against content-preserving manipulations such as JPEG compression, watermark embedding, scaling, rotation, brightness and contrast adjustment, gamma correction and Gaussian low-pass filtering. Receiver operating characteristics (ROC) curve comparisons indicate that our hashing has better performances than two existing algorithms in classification between perceptual robustness and discriminative capability.

Keywords: Image Hashing, Image Rotation, Image Copy Detection, Image Indexing, Ring Division, Histogram

1. Introduction

Nowadays people get more and more images by digital cameras, smart mobile phones and the Internet. Thus, efficient technologies for managing large-scale images are in demand. For example, some images have several copies distributed in many websites. These image copies have the same visual appearance, but their file sizes or file names are different. When we download images from the Internet, we wonder whether or not the image copies are stored in our local computer. Clearly, conventional techniques of content-based image retrieval (CBIR) can achieve this goal, but they are inefficient due to their retrieval results containing many unexpected images. In this work, we discuss a novel technology of multimedia called image hashing, which can be applied to image copy detection, image retrieval, digital watermarking, image indexing, content authentication [1], phishing webpage detection [2], and so on.

Image hashing derives a content-based compact representation called image hash from an input image. It has two basic properties as follows. (1) Perceptual robustness: It maps visually identical images to the same or very similar hashes regardless of their digital representations. In other words, an image and its processed versions generated by content-preserving operations should have the same or similar representations. (2) Discriminative capability: Different images should have different image hashes. This means that the probability of the hashes of different images judged as those of similar images should be very low. Conventional cryptographic hash algorithms, e.g., SHA-1 and MD5, can convert any size input message into a short string and also satisfy discriminative capability. However, they are very sensitive to digital representation of the input message. One bit changed will lead to a completely different hash. Therefore, cryptographic hashes are not suitable for image hashing.

The concept of image hashing is firstly introduced by Schneider and Chang [3] at the 3rd International Conferences on Image Processing (ICIP) in 1996. From then on, more and more researchers pay attention to image hashing and propose a lot of theories and algorithms. The existing hashing algorithms can be roughly classified into the following two categories in terms of their sensitivity to image rotation.

(1) Rotation-sensitive algorithms: This kind of algorithms is very sensitive to image rotation or only resilient to small-angle rotation. For example, Venkatesan et al. [4] exploited discrete wavelet transform (DWT) coefficients statistics to construct image hashes. This algorithm is robust
against JPEG compression, median filtering and rotation within 2°, but fragile to gamma correction and contrast adjustment. Fridrich and Goljan [5] used discrete cosine transform (DCT) coefficients to build robust hash for digital watermarking. This method is robust against JPEG compression, brightness and contrast adjustment, but sensitive to image rotation. Lin and Chang [6] proposed a technique using the invariant relations between DCT coefficients at the same position in separate blocks. This method is also fragile to image rotation. Monga and Evans [7] used the end-stopped wavelet transform to detect visually significant points for hash construction. This method is robust against small-angle rotation, JPEG compression and scaling. In [8], Tang et al. found invariant relation between adjacent non-negative matrix factorization (NMF) coefficients, and exploited it to design image hashing. The scheme is resilient to content-preserving operations except rotation. Recently, Ou and Rhee [9] applied Radon transform (RT) to the input image, randomly selected 40 projections to perform 1-D DCT, and took the first AC coefficient of each projection to produce hash. The RT-DCT hashing is resistant to rotation within 5°. In [10], Tang et al. designed a lexicographical image hashing based on DCT and NMF. This algorithm is resilient to image rotation within 1°. In another work [11], Tang et al. proposed a robust hashing by using local image entropies and DWT. This approach can tolerate rotation within 5°.

(2) Rotation-resistant algorithms: This kind of hashing is robust against rotation with arbitrary angle. For example, Lefèbvre et al. [12] used RT to obtain image features resilient to rotation and other basic image processing operations. Motivated by RT, Roover et al. [13] proposed a RASH method by dividing an image into a set of radial projections of image pixels, extracting Radial Variance (RAV) vector from these projections and applying DCT to conduct RAV vector compression. This hash is robust against image rotation and re-scaling. In [14], Kozat et al. viewed image and attacks as a sequence of linear operators, and proposed to calculate hashes using singular value decompositions (SVDs). The SVD-SVD hashing is resilient to geometric attacks, e.g., rotation, at the cost of significantly decreasing discriminative capability. In another study [15], Monga and Mihçak applied NMF to some sub-images, used factorization factors to construct a secondary image, used NMF to decompose the secondary image again, and exploited the factorization factors to construct robust hash. The NMF-based method is robust against image rotation, but sensitive to several digital manipulations, e.g., watermark embedding.

In general, discriminative capabilities of the rotation-sensitive algorithms are desirable, while those of the rotation-resistant algorithms should be improved. Simultaneously satisfying good discriminative capability and rotation robustness has become a challenging task of image hashing. In the work, we propose a robust image hashing based on multiple histograms. This algorithm can reach an acceptable trade-off between rotation robustness and discrimination. The rest of this paper is organized as follows. Section 2 describes the proposed image hashing. Section 3 presents the experimental results. Conclusions are finally drawn in Section 4.

2. Proposed image hashing

As shown in Figure 1, our image hashing consists of three steps. In the first step, we firstly convert the input image into a normalized image for robust feature extraction. Next, we divide the normalized image into different rings. Finally, we extract ring-based histograms and compress them by DWT.

![Figure 1. Block diagram of our image hashing](image.png)

The detailed steps of our algorithm are as follows.

1. **Preprocessing.** The input image is firstly converted to a normalized size \( m \times m \) by bi-cubic interpolation. The image resizing is to ensure that those images with different resolutions have the same or very similar hashes. The square image is then blurred by a Gaussian low-pass filter. The filtering operation is to reduce high frequency components and alleviate influences of
minor image modifications, e.g., noise contamination and filtering, on the hash values. If the input is color image, we conduct color space conversion from RGB color space to YCbCr color space and take the luminance component for image representation. The conversion can be done by the following equation.

\[
\begin{bmatrix}
Y \\
C_b \\
C_r
\end{bmatrix} = \begin{bmatrix}
65.481 & 128.553 & 24.966 \\
-37.797 & -74.203 & 112 \\
112 & -93.786 & -18.214
\end{bmatrix} \begin{bmatrix}
R \\
G \\
B
\end{bmatrix} + \begin{bmatrix}
16 \\
128 \\
128
\end{bmatrix}
\]

(1)

where \(R\), \(G\) and \(B\) are the red, green and blue components of a pixel, and \(Y\), \(C_b\) and \(C_r\) are its luminance, blue-difference chroma and red-difference chroma, respectively.

Next, histogram equalization is conducted to reduce effect of contrast adjustment.

(2) Ring division. Since rotation manipulation often takes image center as origin of coordinates, those pixels in the inscribed circle of an image are kept unchanged. So we divide the normalized image into different rings with equal area and extract ring-based features to make our hash resilient to rotation. Figure 2 is a schema of ring division. It is clear that pixels in the smallest ring can be determined by the innermost radius and those in other ring can be calculated by the adjacent radii. Let \(r_i (i=1, 2, ..., n)\) be the \(i\)-th radius labeled from small value to big value, where \(r_1\) is the innermost radius, \(r_n\) is the outermost radius, and \(n\) is the ring number. Thus, for the \(m \times m\) normalized image, \(r_0 = \lceil m/2 \rceil\), where \(\lceil \cdot \rceil\) means downward rounding. So the area of the inscribed circle is:

\[S_0 = \pi r_0^2\]

(2)

and the average area of each ring is:

\[S_i = \frac{S_0}{n}\]

(3)

Therefore, \(r_1\) can be determined by the following equation.

\[r_1 = \sqrt{\frac{S_0}{\pi}}\]

(4)

Other radii \(r_k (k = 2, 3, ..., n - 1)\) can be calculated by

\[r_k = \sqrt{\frac{S_1 + \pi r_{k-1}^2}{\pi}}\]

(5)

Let \((x_c, y_c)\) be the coordinates of the image center. Thus, \(x_c = m/2 + 0.5\) and \(y_c = m/2 + 0.5\) if \(m\) is an even number. Otherwise, \(x_c = (m+1)/2\) and \(y_c = (m+1)/2\). We calculate the Euclidean distance between the pixel \(p(x, y)\) (1 \(\leq x, y \leq m\)) and the image center as follows.
So we can classify pixels into \( n \) subsets by the following rules.

\[
A_1 = \{ p(x, y) \mid d_{x,y} \leq r_1 \}
\]

(7)

\[
A_k = \{ p(x, y) \mid r_{k-1} < d_{x,y} \leq r_k \} \quad (k = 2, 3, \ldots, n)
\]

(8)

where \( A_1 \) is the subset containing pixel values in the innermost ring and \( A_k \) is the \( k \)-th subset including those in the \( k \)-th ring.

(3) **Histogram calculation.** We calculate histograms of the subsets and use them for representation. Let \( H_k[l] \) be the \( l \)-th bin of the histogram of \( A_k \), where \( l = 0, 1, 2, \ldots, L \) and \( k = 1, 2, \ldots, n \). For the luminance component of a color image, \( L = 255 \). Thus, \( H_k[l] \) can be defined as

\[
H_k[l] = t_k(l)
\]

(9)

where \( t_k(l) \) is the total number of the pixels with value equaling \( l \) in \( A_k \). To further reduce the influences of image modifications such as noise corruption, we apply a single-level 1-D DWT to the histogram of each subset. This can be done by passing it through a low-pass wavelet filter \( g \) and a high-pass wavelet filter \( w \) as follows.

\[
C_k^{(\text{low})} = (H_k * g) \downarrow 2
\]

(10)

\[
C_k^{(\text{high})} = (H_k * w) \downarrow 2
\]

(11)

where ‘*’ represents convolution operation, ‘\( \downarrow \)’ denotes sub-sampling operation, \( C_k^{(\text{low})} \) and \( C_k^{(\text{high})} \) are approximation coefficients and detail coefficients, respectively. Since \( H_k \) has \( L \) bins, the length of \( C_k^{(\text{low})} \) is \( L^{(\text{low})} = \lceil L/2 \rceil \), where \( \lceil \cdot \rceil \) means upward rounding.

To make a short hash, we divide \( C_k^{(\text{low})} \) into \( b \) segments whose element numbers are almost equal, and exploit variance of each segment to form image hash. Let \( n_1 = \lfloor L^{(\text{low})}/b \rfloor \) and \( n_2 = \text{mod}(L^{(\text{low})}, b) \), where \( \text{mod}(\cdot, \cdot) \) is the mod operation. Thus, the first \( n_2 \) segments have \( (n_1+1) \) elements and other segments have \( n_1 \) elements. Suppose that \( v_{i,j} \) is the array containing the DWT coefficients in the \( i \)-th segment of \( C_k^{(\text{low})} \) \( (i = 1, 2, \ldots, b) \). Thus, the variance can be calculated by

\[
\delta_{k,i}^2 = \frac{1}{n_3-1} \sum_{j=1}^{n_3} \left[ v_{i,j}(j) - \mu_{k,i} \right]^2
\]

(12)

where \( v_{i,j}(j) \) is the \( j \)-th element of \( v_{i,j} \), \( n_3 \) is the element number of \( v_{i,j} \) and \( \mu_{k,i} \) is the mean of \( v_{i,j} \) defined as

\[
\mu_{k,i} = \frac{1}{n_3} \sum_{j=1}^{n_3} v_{i,j}(j)
\]

(13)

Finally, the image hash is obtained by concatenating all variances \( \delta_{k,i}^2 \). So the hash length is \( N = nb \) decimal digits. For example, if the input image is divided into 5 rings and DWT result of each ring histogram is divided into 4 segments, i.e., \( n=5 \) and \( b=4 \), the hash length is \( N = 5 \times 4 = 20 \) decimal digits.

To measure similarity between two image hashes, we take \( L_2 \) norm as similarity metric. Let \( h_1 \) and \( h_2 \) be two hashes. Thus, the \( L_2 \) norm can be defined as follows.
\[ D(h_1, h_2) = \sqrt{\sum_{i=1}^{N} |h_1(i) - h_2(i)|^2} \] (14)

where \( h_1(i) \) and \( h_2(i) \) are the \( i \)-th elements of \( h_1 \) and \( h_2 \), respectively. The more similar the input images of the hashes, the smaller the \( L_2 \) norm. If the \( L_2 \) norm is smaller than a pre-defined threshold \( T \), the images are considered as visually identical images. Otherwise, they are viewed as different images.

3. Experimental results

To validate the performances of our hashing, we conduct experiments on classification between perceptual robustness and discriminative capability in Subsection 3.1. In Subsection 3.2, we compare our hashing with two existing algorithms: the RT-DCT hashing [9] and the SVD-SVD hashing [14]. The parameters of our hashing used in Subsections 3.1 and 3.2 are: the input image is resized to 512×512, a 3×3 Gaussian low-pass filter with a unit standard deviation is used, ring number is \( n=4 \), segment number is \( b=4 \), and the Haar wavelet is adopted. In Subsection 3.3, we discuss effect of the ring number on our hash performances.

3.1. Classification performances

To test the perceptual robustness, we take five standard color images sized 512×512, i.e., Airplane, Baboon, House, Peppers, and Lena, as test images. These images are attacked by Photoshop, MATLAB and StirMark 4.0 [16]. The used content-preserving manipulations and their detailed parameter values are presented in Table 1. Since image rotation will expand image size and the added regions are padded with black or white pixels, we crop the original and the rotated images and then take their central parts of size 361×361 for hash generation. After the above operations, each image has 60 attacked images. Thus, there are 300 pairs of visually identical images in total. We extract image hashes of the original images and their attacked images, and then calculate their similarities by the equation (14). Table 2 presents the minimum, mean, and maximum \( L_2 \) norms under different manipulations and their standard deviations. From the results, we find that the maximum \( L_2 \) norms of the test operations are smaller than 48000 except that of JPEG compression. This means that we can take \( T = 48000 \) as a threshold to resist most of the test operations. If the threshold reaches 250000, our hashing is robust against all the above operations.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Operation</th>
<th>Description</th>
<th>Parameter value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photoshop</td>
<td>Brightness adjustment</td>
<td>Photoshop's scale</td>
<td>10, 20, −10, −20</td>
</tr>
<tr>
<td>Photoshop</td>
<td>Contrast adjustment</td>
<td>Photoshop's scale</td>
<td>10, 20, −10, −20</td>
</tr>
<tr>
<td>MATLAB</td>
<td>Gamma correction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MATLAB</td>
<td>3×3 Gaussian low-pass filtering</td>
<td>Standard deviation</td>
<td>0.3, 0.4, ..., 1.0</td>
</tr>
<tr>
<td>StirMark</td>
<td>JPEG compression</td>
<td>Quality factor</td>
<td>30, 40, ..., 100</td>
</tr>
<tr>
<td>StirMark</td>
<td>Watermark embedding</td>
<td>Strength</td>
<td>10, 20, ..., 100</td>
</tr>
<tr>
<td>StirMark</td>
<td>Scaling</td>
<td>Ratio</td>
<td>0.5, 0.75, 0.9, 1.1, 1.5, 2.0</td>
</tr>
<tr>
<td>StirMark</td>
<td>Rotation and cropping</td>
<td>Angle in degree</td>
<td>1,2,5,10,15,30,45,90,—1,—2,—5,—10,—15,—30,—45,—90</td>
</tr>
</tbody>
</table>
Table 2. Minimum, mean and maximum $L_2$ norms under different manipulations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Max.</th>
<th>Min.</th>
<th>Mean</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brightness adjustment</td>
<td>13084</td>
<td>786</td>
<td>5228</td>
<td>4022.34</td>
</tr>
<tr>
<td>Contrast adjustment</td>
<td>45064</td>
<td>628</td>
<td>7677</td>
<td>9900.97</td>
</tr>
<tr>
<td>Gamma correction</td>
<td>47988</td>
<td>1614</td>
<td>10498</td>
<td>11932.05</td>
</tr>
<tr>
<td>3×3 Gaussian low-pass filtering</td>
<td>21517</td>
<td>13</td>
<td>7570</td>
<td>5903.95</td>
</tr>
<tr>
<td>JPEG compression</td>
<td>246784</td>
<td>1081</td>
<td>36288</td>
<td>63964.62</td>
</tr>
<tr>
<td>Watermark embedding</td>
<td>22117</td>
<td>585</td>
<td>5461</td>
<td>4034.54</td>
</tr>
<tr>
<td>Scaling</td>
<td>24274</td>
<td>1197</td>
<td>6930</td>
<td>5892.32</td>
</tr>
<tr>
<td>Rotation and cropping</td>
<td>25561</td>
<td>1432</td>
<td>8285</td>
<td>6397.35</td>
</tr>
</tbody>
</table>

We collect 200 different color images to construct an image database for discrimination validation, where 67 images are downed from the Internet, 33 images are captured by digital cameras and 100 images are taken from the Ground Truth Database [17]. The image sizes range from 256×256 to 2048×1536. We calculate image hashes of these different images, compute $L_2$ norm between each pair of hashes, and then obtain 19900 results. It is observed from the results that, the minimum and maximum $L_2$ norms are 9920 and 4041011, and the mean and the standard deviation of these distances are 344567 and 636826.05, respectively.

To view our performances in classification between the perceptual robustness and discriminative capability, we calculate true positive rate (TPR) and false positive rate (FPR), which are two important indices of the receiver operating characteristics (ROC) graph [18]. The TPR and FPR are defined as

$$TPR = \frac{\text{Pair number of images considered as similar}}{\text{Total pair number of visually identical images}} \quad (15)$$

$$FPR = \frac{\text{Pair number of different images considered as similar}}{\text{Total pair number of different images}} \quad (16)$$

It is clear that TPR and FPR indicate robustness and discriminative capability of a hashing algorithm, respectively. We choose different thresholds and calculate their TPRs and FPRs. The results are presented in Table 3. We can select proper threshold values in terms of specific applications.

Table 3. TPRs and FPRs under different thresholds

<table>
<thead>
<tr>
<th>T</th>
<th>TPR</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>2800</td>
<td>0.2433</td>
<td>0</td>
</tr>
<tr>
<td>5600</td>
<td>0.5233</td>
<td>0</td>
</tr>
<tr>
<td>9600</td>
<td>0.7667</td>
<td>0</td>
</tr>
<tr>
<td>38000</td>
<td>0.9600</td>
<td>0.0424</td>
</tr>
<tr>
<td>56000</td>
<td>0.9700</td>
<td>0.1310</td>
</tr>
<tr>
<td>90000</td>
<td>0.9833</td>
<td>0.2929</td>
</tr>
<tr>
<td>160000</td>
<td>0.9900</td>
<td>0.5865</td>
</tr>
<tr>
<td>300000</td>
<td>1.0000</td>
<td>0.7969</td>
</tr>
</tbody>
</table>
3.2. Performance comparisons

To show advantages, we compare our hashing with the RT-DCT hashing [9] and the SVD-SVD hashing [14]. The used parameters of the SVD-SVD hashing are: the first rectangle number is 100, the rectangle size is 64×64, the second rectangle number is 20 and the rectangle size is 40×40. To make fair comparisons, the test images used in Subsection 3.1 are also adopted to generate the hashes of the RT-DCT hashing and the SVD-SVD hashing. We exploit ROC graphs to visualize classification performances of the assessed algorithms. In the ROC graphs, if the algorithms have the same FPR, the hashing with big TPR is better than the one with small TPR. Similarly, if the algorithms have the same TPR, the method with small FPR outperforms the scheme with big value.

We choose respective thresholds of the assessed algorithms, calculate their corresponding TPRs and FPRs, and obtain the ROC curves as shown in Figure 3. The used thresholds of our hashing are: 0, 2800, 5600, 9600, 38000, 56000, 96000, 160000, 300000, 1100000, those of the RT-DCT hashing are: 0.01, 0.1, 0.3, 0.35, 0.38, 0.4, 0.42, 0.43, 0.45 and 0.5, and those of the SVD-SVD hashing are: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7 and 0.8. We find that the ROC curve of our hashing is above those of RT-DCT hashing and the SVD-SVD hashing. This means that our hashing is better than the two hashing algorithms in classification between the robustness and discriminative capability. For example, when FPR is near 0, TPR of our hashing is 0.7667 while those of the RT-DCT hashing and the SVD-SVD hashing are about 0.55 and 0.11, respectively. When TPR reaches 1, FPR of our hashing is about 0.62 while those of the RT-DCT hashing and the SVD-SVD hashing are 1.0 and 0.95, respectively.

![Figure 3. ROC curve comparisons between our hashing and other hashing algorithms](image)

3.3. Effect of ring number

To view effect of the ring number on hash performances, we vary the ring number \(n\) and the segment number \(b\), and keep other parameters used in the above subsections unchanged. In the experiments, the used ring numbers are: 1, 2, 3, 4 and 5. For each ring number, we choose a proper \(b\) value to ensure that hash lengths under different \(n\) values are almost equal. Table 4 presents the used \(n\) values, the \(b\) values, and the corresponding hash lengths. We calculate the ROC curves under different ring numbers, and obtain the results as shown in Figure 4. It is observed that, when the ring number is small, \(e.g., n=1\), our hash performances are not good enough. As the ring number increases, the hash performances are gradually improved. However, if the ring number is bigger than a certain value, our hash performances will decrease. So a moderate ring number, \(e.g., 3 \text{ or } 4\), can reach a good trade-off between the robustness and discrimination. For example, the ROC curves of \(n=3\) and \(n=4\) are above
those of $n=1$ and $n=2$, meaning that hash performances under $n=3$ or $n=4$ are better than those under $n=1$ or $n=2$. When $n=5$, the whole performances are not better than those of $n=3$ or $n=4$.

Table 4. Hash lengths under different $n$ and $b$ values

<table>
<thead>
<tr>
<th>$n$</th>
<th>$b$</th>
<th>Hash length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>15</td>
</tr>
</tbody>
</table>

Figure 4. ROC curve comparisons among different ring numbers

4. Conclusions

In this work, we have proposed a robust image hashing based on multiple histograms. As ring pixels are invariant to image rotation, ring pixels based histograms are resistant to rotation with arbitrary angle. We conducted many experiments to validate our hashing performances. The results show that our hashing is robust against normal digital processing such as JPEG compression, watermark embedding, scaling, rotation, brightness and contrast adjustment, gamma correction, and Gaussian low-pass filtering. ROC curve comparisons indicate that our hashing has better performances than the RT-DCT hashing and the SVD-SVD hashing in classification between the robustness and discriminative capability.

Acknowledgments

This work was partially supported by the Natural Science Foundation of China (61165009), the Guangxi Natural Science Foundation (2012GXNSFBA053166, 2012GXNSFGA060004, 2011GXNSFD018026, 0832104), the ‘Bagui Scholar’ Project Special Funds, the Project of the Education Administration of Guangxi (200911MS55), the Scientific Research and Technological Development Program of Guangxi (10123005–8), and the Scientific Research Foundation of Guangxi Normal University for Doctor Programs.
References


