Robust Image Hashing Using Image Normalization and SVD Decomposition

Ricardo Antoio Parrao Hernandez
and Mariko Nakano Miyatake
SEPI, ESIME Culhuacan
National Polytechnic Institute IPN
Av. Santa Ana Num. 1000, Mexico City, 04430 Mexico
Email: parrao744@yahoo.com.mx, nakano-m@ice.uec.ac.jp

Brian M. Kurkoski
Department of Information and Communications Engineering,
University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo, 182-8585 Japan
Email: kurkoski@ice.uec.ac.jp

***Abstract*— This paper introduces the use of image normalization and SVD decomposition hash functions to generate a hash value for a digital image. Image normalization is a technique that has been shown to be robust against different kinds of geometric attacks such as rotation, scaling and flipping. Using image normalization as a preprocessing step we can increase the robustness of the hash by 50% against rotation and scaling compared to without this algorithm.**

I. Introduction

During the information era, the interchange of digital content has had tremendous growth, but in the same way tools to manipulate digital content like Photoshop, Corel Draw and Illustrator, are becoming more popular and easier to handle, which leads an increase in the use of these tools for manipulating images for illegal purposes or to discredit people. To ensure trustworthiness, multimedia authentication techniques like watermarking and image hashing have emerged to verify content integrity and prevent forgery.

Traditionally, data integrity issues are addressed by message authentication functions or cryptographic hashes, which are key-dependent and bit sensitive. The integrity of the message can be validated only when every bit of the message is unchanged [2]. This sensitivity is usually applied to text message authentication. For digital images this sensitivity is not so straightforward because it should be able to tolerate some lossy modifications with graceful degradation. Therefore bit-by-bit authentication is no longer suitable, and a tool that can authenticate content is more appropriate.

An image hash is an image-based content digital signature. To generate the hash, a secret key is used to extract some features from the content of the image. These features are processed to generate the hash. In addition to content authentication, multimedia hashes are used in content-based retrieval from databases [6]. To search for multimedia content, methods such as sample-by-sample comparisons are computationally inefficient. Moreover, these methods compare the lowest level of content representation and do not offer robustness in situations such as geometric distortions or different kinds of compression. Robust image hash functions can be used to address this problem [10].

There are two important criterias in the design of the image hash functions, robustness and security [10]. By robustness, we mean that if the same key is used, perceptually similar images generate a similar hash. The similarity of hashes is measured in terms of metrics like Hamming or Euclidean distances. The security of the image hash function is through a secret key to generate the hash. Without the knowledge of this key, the hash value should not be easy to estimate or forge.

The Singular Value Decomposition (SVD) is an important linear algebra tool, which is often used in image compression, digital watermarking and other signal processing fields. The SVD decomposes the image into three matrices, U, S and V. The matrix S is a diagonal matrix containing the singular values of the image. The matrices U and V contain the singular vectors of the image, and by applying this decomposition we can reduce the dimensions of the data. The use of SVD decomposition in the image hashing problem was proposed by Kozat et all [5] where SVD decomposition is used twice and has been shown to be robust to some small variations in rotation and scaling.

On the other hand, image normalization is an algorithm based on the invariant moments of the image. These moments are used to geometrically normalize the image in terms of scale, orientation, and flipping. This method is well-known in pattern recognition problems.

This paper introduces image normalization as a preprocessing step in the SVD decomposition hash function to increase robustness against rotation, scaling and JPEG compression. For increasing the security and introducing unpredictability in the hash value we randomly select sub-images, with a random partition algorithm [8], and apply the SVD decomposition hash function to each sub-image. After that, we rearrange the matrices U and V and again apply SVD decomposition to generate the intermediate hash value, and finally quantize and compress to generate the final binary hash value. The proposed algorithm is shown in Figure 1. As will be shown by numerical results, the addition of the normalization function substantially improves the robustness against rotation, scaling and compression.

This paper contains five sections. Section II describes the SVD decomposition, the image normalization algorithm and

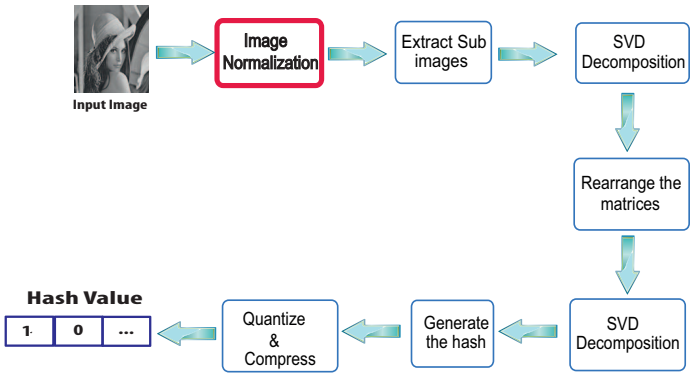


Fig. 1. Proposed algorithm for generate the hash value.

the random partition algorithm. Section III describes the proposed algorithm. Section IV present the results. Section V is the conclusion and the possible future work.

II. Definitions

In this section we introduce the theory behind the SVD decomposition, the image normalization algorithm and its characteristics and the random partition algorithm.

A. SVD Decomposition

We can see an image as an M-by-N matrix, so we can apply the SVD decomposition. We have an image A and apply the SVD decomposition as:

$$A = USV^T, \tag{1}$$

where U and V are orthogonal matrices that contain the left and right singular vectors and S is a diagonal matrix $S = \text{diag}(s_1, s_2, \ldots)$ with the singular values of A. We can express the decomposition as the following:

$$A = \sum_{i=1}^{r} U_i s_i V_i^T, \tag{2}$$

$$A = U_1 s_1 V_1^T + U_2 s_2 V_2^T + \ldots + U_r s_r V_r^T, \tag{3}$$

where r is the rank of A, U_i is a column i of U and V_i is a column of V.

The SVD decomposition for digital images has the following properties [4]:

1) The singular values change little when a small disturbance is applied.
2) The singular values represent the brightness features of the image, while the singular vectors represent the geometrical features.
3) The quality of the reconstructed image does not degenerate if we remove the small-valued singular values s_i.

B. Image normalization algorithm

Image normalization is an algorithm based on the geometric moments of the image. With this algorithm, an image I is robust to different kinds of affine transforms [3]. The algorithm is as follows.

Let I denote a digital image of size M-by-N. Its geometric moments m_{pq} and central moment μ_{pq}, where $p, q = 0, 1, 2, \ldots$ are defined as:

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x - \bar{x})^p (y - \bar{y})^q F(x, y). \tag{4}$$

And,

$$\mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q F(x, y), \tag{5}$$

where

$$\bar{x} = \frac{m_{1,0}}{m_{0,0}}, \quad \bar{y} = \frac{m_{0,1}}{m_{0,0}}. \tag{6}$$

An image G is said to be an affine transform of I if there is a matrix $H = \left(\begin{smallmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{smallmatrix}\right)$ and a vector $d = (d_1 \; d_2)^T$ such that $G(x, y) = I(x_a, y_a)$, where

$$\begin{pmatrix} x_a \\ y_a \end{pmatrix} = H \begin{pmatrix} x \\ y \end{pmatrix} - d. \tag{7}$$

The normalization procedure consists of the following steps for a given unnormalized image I:

1) Center the image I using the matrix $H = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ and the vector $d = (d_1 \; d_2)^T$ with:

$$d_1 = \frac{m_{1,0}}{\mu_{0,0}}, \quad d_2 = \frac{m_{0,1}}{\mu_{0,0}}. \tag{8}$$

This step performs translation invariance. Let I_1 denote the resulting centered image.

2) Apply a shearing transform to I_1 in the x direction with matrix $H = \left(\begin{smallmatrix} 1 & \beta \\ 0 & 1 \end{smallmatrix}\right)$ so the resulting image is denoted by I_2. To determine β we use:

$$\mu_{30} + 3\beta\mu_{21} + 3\beta^2\mu_{12} + \beta^3\mu_{03} = 0. \tag{9}$$

We can have up to three roots, if one of the roots is real and the others are complex, we take the real one, if two or three roots are real we pick the median of the real roots, if all roots are complex β is 0, and under some very unusual conditions β will have infinite number of solutions if all the moments are zero. This happens when the image is rotationally symmetric [9].

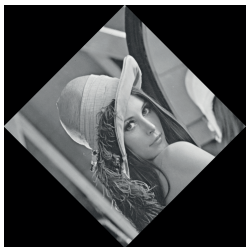
3) Apply a shearing transform to I_2 in the y direction with matrix $H = \left(\begin{smallmatrix} 1 & 0 \\ \lambda & 1 \end{smallmatrix}\right)$, the resulting image is denoted by I_3, where λ is defined with the equation:

$$\mu_{11} = \lambda\mu_{20} + \mu_{11}, \tag{10}$$

and setting $\mu_{11} = 0$ we obtain:

(a) Input: unmodified image

(b) Input: image rotate 45 degrees



(c) Output of image normalization algorithm

Fig. 2. Input and output of the image normalization algorithm

$$\lambda = \frac{\mu_{11}}{\mu_{20}}. \tag{11}$$

4) Finally, the image I_3 is resized to a specified sizeto obtain the normalizaed image G.

In Figure 2 we can see the result of the image normalization algorithm applied to the lena image. Figure 2(a) is the original image of Lena, Figure 2(b) is the Lena image rotated 45 degrees. Figure 2(c) is the output of the image normalization when either (a) or (b) is applied as an input. With this technique we observe that this algorithm is robust to scaling, rotation and flipping.

C. Random partition algorithm

For increasing the security of the hash value and adding unpredictability, we add some randomness by extracting sub-images. We form a number p pseudo-randomly selected sub-images with the random partition algorithm. This consists of:

1) With a key k pseudo-randomly generate a set W where $W = ((x_1, y_1), (x_2, y_2), \ldots, (x_p, y_p))$, which is the coordinate pair of the top-left corner of the sub-image $A_1, A_2, \ldots, A_p$.
2) For each coordinate pair generate a square of size m-by-m, and each square is the output that we call sub-image A_i.

III. Proposed algorithm

A robust image hash function H_k has two inputs: an image *I* and a secret key *k* and has one output, a hash, which is a short binary vector $\vec{h} = H_k(I)$. The hash function should possess the following two perceptual properties:

1) Hash values for all perceptually identical images, that is when a person cannot find any difference in the content of the image, should be the same.
2) Perceptually different images, when two images do not have the same content in terms of visual perception, should have different hash values.

Next, we present the description of our proposed algorithm:

1) Let the M-by-N input image be $I \in [0, 255]^{M \times N}$. The number of sub-images is denoted by p and the size of this sub-image is denoted by m where $1 < m < M, N$, and the secret key for generating the hash is denoted by k. And the output $\vec{h}$ is the final hash value.
2) To I we apply the image normalization described in Section II-B to generate the normalized image G.
3) From G,with the use of a key k, pseudo-randomly form p possibly overlapping squares $A_1, A_2, \ldots, A_p$ each of size m-by-m as described in Section II-C.
4) To each rectangle A_i apply the SVD decomposition to obtain U, S, V, as is shown in equation (3).
5) From equation 3 we take the vector $U_1^{(i)}$ and $(V_1^{(i)})^T$ compose a new matrix *B*,

$$B = \left[U_1^{(1)}, \ldots U_1^{(p)}, (V_1^{(1)})^T \ldots (V_1^{(p)})^T\right], \tag{12}$$

with size m-by-$2p$.
6) We apply the SVD decomposition to *B* to get the matrices *U', S'* and *V'*.
7) The first column of *U'* and the first row of V'^T are the intermediate hash with length $m + 2p$. This is a vector of real numbers.
8) And finally, we quantize the resulting statistics vector and apply Gray coding to obtain the binary hash sequence. This binary sequence is then passed through the decoding stage of an order-3 Reed-Muller decoder for compression [7].
9) The output of the Reed-Muller decoder is the binary hash $\vec{h}$.

IV. Results

To measure the performance of the proposed algorithm, we choose the Hamming distance between the binary hashes, normalized with respect to the length (L) of the binary hash as a performance metric. The normalized Hamming distance is defined as:

$$d_h(h_1, h_2) = \frac{1}{L} \sum_{k=1}^{L} \|h_1(k) - h_2(k)\|_1 . \tag{13}$$

For doing the numerical evaluation we used different standard test images as an input and considered different kinds of attacks such as rotation, compression and scaling, using the Stirmark Benchmark Tool 4.0 [1]. These images are a 512-by-512 grayscale and also this is the size we use for the output in the image normalization algorithm. The number of sub-images is $p = 15$ and the subimage size is $m = 100$. We compare the algorithm with and without image normalization [5].

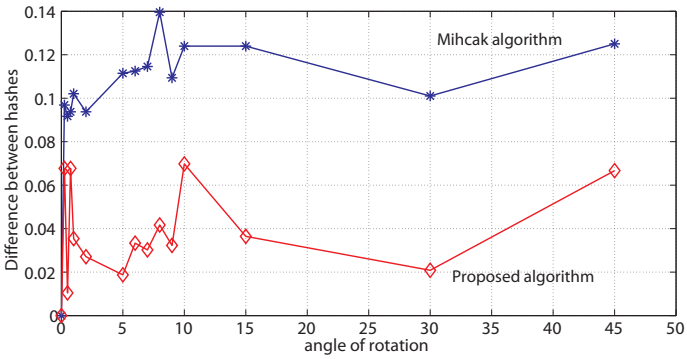


Fig. 3. Rotation attack, different angle of rotation vs distance between hashes.

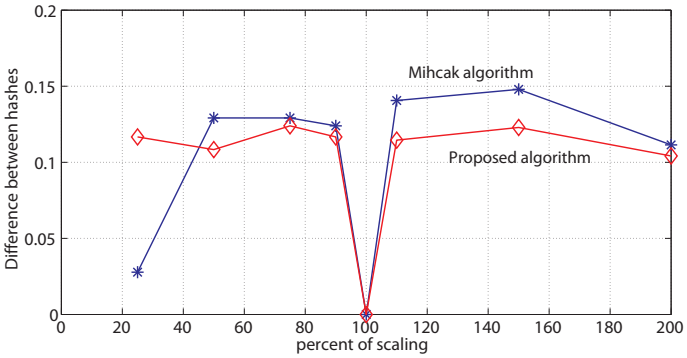


Fig. 4. Scaling attack, different percent of scaling vs distance between hashes, where 100 percent is the image in its original size.

For the rotatation attack in Figure 3 we can observe after applying image normalization we can improve the performance in terms of Hamming distance from 0.14 to 0.07, that is an improvement of 50%.

For the scaling attack in Figure 4 we can observe the reduction of the Hamming distance is lower after applying image normalization, while Mihcak looks lower when we reduce the image size, actually this effect occurs when the image is reduce to 128-by-128 and all subimages are almost the same.

In the same way, for JPEG compression in Figure 5 we get better performance after applying the image normalization, and we can reduce the Hamming distance to under 0.1, and for doing authenticationit appears a threshold of 0.1 is suitable.

V. Conclusion

We present the use of image normalization as a preprocessing step to increase the robustness of the image hashing function based on SVD decomposition. As a result, there was significant improvement in terms of Hamming distance against geometric attacks such as rotation and scaling. For applications to a databases this characteristic is very useful because we can search based on the content of the image without being concerned about the orientation or the size of the image.

We apply SVD decomposition twice because if we apply the SVD decomposition only once, then the hash is not sufficiently robust or secure. But, if it is performed more than

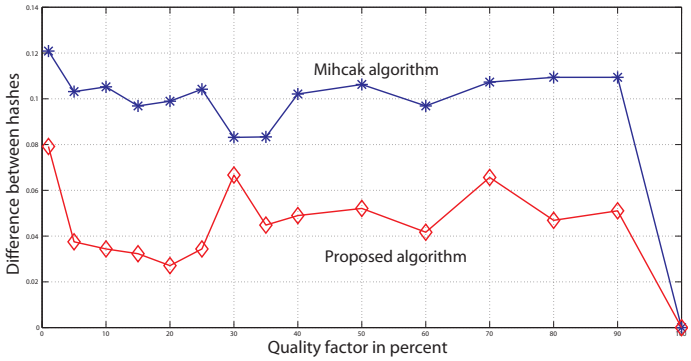


Fig. 5. JPEG compression attack, different quality factor in percent vs distance between hashes, where 100 percent is without compression.

twice, experiments have shown that it would fail [8]. A good SVD-based image authentication method should use the SVD decomposition exactly twice.

As future work, we need to explore different parameters for generating the sub-images to obtain a better content preservation, and also to try different codes for compressing the final hash value.

Acknowledgment

This research was sponsored by the UEC Japan through JUSST program, and CONACyT of Mexico.

This research was supported in part by the Ministry of Education, Science, Sports and Culture; Grant-in-Aid for Scientific Research (C) number 21560388.

References

[1] `http://www.petitcolas.net/fabien/watermarking/stirmark/`.

[2] A. Menezes, V. O., and Vanstone, S. *Handbook of Applied Cryptography*. CRC Pressr, 1998.

[3] Alghoniemy, M., and Tewfik, A. Geometric invariance in image watermarking. *Image Processing, IEEE Transactions on 13*, 2 (2004), 145 –153.

[4] Andrews, H., and Patterson, C. Singular value decompositions and digital image processing. *Acoustics, Speech and Signal Processing, IEEE Transactions on 24*, 1 (Feb. 1976), 26 – 53.

[5] Kozat, S., Venkatesan, R., and Mihcak, M. Robust perceptual image hashing via matrix invariants. In *Image Processing, 2004. ICIP '04. 2004 International Conference on* (2004), vol. 5, pp. 3443 – 3446 Vol. 5.

[6] Lin, S., Özsu, M. T., Oria, V., and Ng, R. T. An extendible hash for multi-precision similarity querying of image databases. In *Proceedings of the 27th International Conference on Very Large Data Bases* (San Francisco, CA, USA, 2001), VLDB '01, Morgan Kaufmann Publishers Inc., pp. 221–230.

[7] Mhak, M. K., and Venkatesan, R. A tool for robust audio information hiding: A perceptual audio hashing algorithm. In *in Proceedings of 4th Information Hiding Workshop* (2001), pp. 51–65.

[8] Monga, V., and Evans, B. Perceptual image hashing via feature points: Performance evaluation and tradeoffs. *Image Processing, IEEE Transactions on 15*, 11 (Nov. 2006), 3452 –3465.

[9] Shen, D., and Ip, H. Generalized affine invariant image normalization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 19*, 5 (May 1997), 431 –440.

[10] Venkatesan, R., Koon, S.-M., Jakubowski, M., and Moulin, P. Robust image hashing. In *Image Processing, 2000. Proceedings. 2000 International Conference on* (Feb. 2000), vol. 3, pp. 664 –666 vol.3.