A Secure Perceptual Hash Algorithm for Image Content Authentication

Li Weng, Bart Preneel

A Secure Perceptual Hash Algorithm for Image Content Authentication

Li Weng and Bart Preneel*

Katholieke Universiteit Leuven, ESAT/COSIC-IBBT
li.weng@esat.kuleuven.be,bart.preneel@esat.kuleuven.be

Abstract. Perceptual hashing is a promising solution to image content authentication. However, conventional image hash algorithms only offer a limited authentication level for the protection of overall content. In this work, we propose an image hash algorithm with block level content protection. It extracts features from DFT coefficients of image blocks. Experiments show that the hash has strong robustness against JPEG compression, scaling, additive white Gaussian noise, and Gaussian smoothing. The hash value is compact, and highly dependent on a key. It has very efficient trade-offs between the false positive rate and the true positive rate.

1 Introduction

In the Internet era, images are massively produced and distributed in digital form. Although digital images are easy to store and process, they are also susceptible to malicious modification. Due to widely available image editing software, even non-professionals can perform content modification. Consequently, people begin to suspect what they see from digital images. Sometimes, public incidents happen, due to fake images. Therefore, the need for protecting content authenticity is emerging.

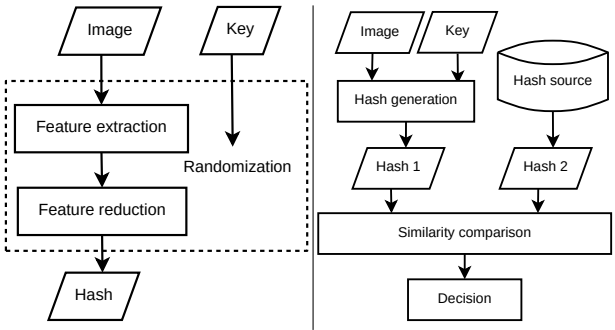
Among various techniques, perceptual hashing is a promising solution. Hashing means to compute a digest value from data. This digest value, typically a short binary string, is called a *hash value*. Perceptual hash algorithms are a particular kind of hash algorithms for multimedia data. They have the special property that the hash value is dependent on the multimedia content, and it remains approximately the same if the content is not significantly modified. Since a

* This work was supported in part by the Concerted Research Action (GOA) AM-BioRICS 2005/11 of the Flemish Government and by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy). The first author was supported by the IBBT/AQUA project. IBBT (Interdisciplinary Institute for Broad-Band Technology) is a research institute founded in 2004 by the Flemish Government, and the involved companies and institutions (Philips, IPGlobalnet, Vitalsys, Landsbond onafhankelijke ziekenfondsen, UZ-Gent). Additional support was provided by the FWO (Fonds Wetenschappelijk Onderzoek) within the project G.0206.08 Perceptual Hashing and Semi-fragile Watermarking.

perceptual hash value is a compact representation of the original content, it can be used for robust content authentication. Compared with conventional cryptographic hash algorithms [1], perceptual hash algorithms have the advantage that they can tolerate the difference in quality and format – the binary representation no longer matters; the same content always maps to the same hash value. This is particularly useful for the multimedia domain.

In this work, we focus on image content authentication by perceptual hash algorithms. In a typical application scenario, the authentic hash value is available; anyone who suspects the image can compute the hash value and compare it with the authentic one (Fig. 1b). For example, the authentic hash value can be published online, or electronically signed by digital signature techniques [1]. Although this application is known, there are some unsolved issues. In particular, there is the minor modification problem: when malicious modification is perceptually insignificant, the hash algorithm is unable to distinguish it from legitimate distortion. Most image hash algorithms compute the hash value from an image's global features. Since global features are not sensitive to local modification, these algorithms are generally vulnerable to the minor modification problem, thus are not suitable for content authentication applications with high security demand. In this work, a potential solution is provided. We propose an image hash algorithm with the ability of authenticating image blocks.

The rest of the work is organized as follows: Section 2 introduces image hashing and its limitation; Section 3 describes the proposed image hash algorithm; Section 4 shows some experiment results; Section 5 concludes the work.



(a) Hash generation. (b) Hash comparison.

Fig. 1. Diagrams of perceptual hash generation (a) and comparison (b).

2 Perceptual Hashing and Its Limitation

The basic components of a perceptual hash algorithm are feature extraction, feature reduction, and randomization (Fig. 1a). During feature extraction, robust features are extracted from the input signal. Typically, these features are

insensitive to moderate distortion, such as compression, noise addition, etc. Feature reduction, similar to a quantization procedure, includes means to efficiently represent extracted features. Randomization is a critical component for security and other performance. It includes means to achieve the key-dependent property (explained later). Many recent designs begin to support this property. Besides the above major components, there are also pre-processing and post-processing.

The requirements for perceptual hashing come from two aspects. From a generic security point of view, a perceptual hash algorithm should possess the following properties:

- One-way: it is hard to reveal the input from the hash value;
- Collision-resistant: it is hard to find different inputs that have similar hash values; given an input and its hash value, it is hard to find another input which has a similar hash value;
- Key-dependent: the hash value is highly dependent on a key.

The first property is useful for protecting the confidentiality of the input. The second property ensures that the chance of collision is negligible, so that the hash value can be considered only as a fair representation of the corresponding input. The last property is used for entity authentication, i.e., only the entity that knows the key can generate the correct hash value, see [1, message authentication code]. Additionally, from a multimedia security point of view, there is a more demanding requirement:

- The hash value is insensitive to legitimate media content distortion, but sensitive to malicious modification.

Unfortunately, these requirements are not all well fulfilled in practice. This is due to the intrinsic limitation of perceptual hash algorithms.

A limit of perceptual hashing is that perceptually insignificant but malicious distortion cannot be distinguished from legitimate distortion. It is defined here as the *minor modification* problem. Since a perceptual hash value is computed from robust features, it can be used for content authentication. However, the effect is limited. A perceptual hash value is only sensitive to significant content modification, while malicious attacks can be perceptually insignificant [2, 3]. Considering an image as a point in a vector space, the essence of perceptual hashing is dimension reduction [4]. A hash value is typically computed from low dimensional features. It is naturally resistant to distortion that only affects higher dimensions. However, the distortion brought by malicious attacks can be as small as legitimate distortion. As long as the distortion only affects high dimensions, no matter it is legitimate or malicious, it will be tolerated. For example, Fig. 2 shows two Lena images: a) a maliciously modified version; b) a compressed version. Existing image hash algorithms may not be able to distinguish the two images. For example, the ART-based image hash algorithm in [5] is used to compute hash values for these images. The hash values are compared to the original one. The resulted hash distances (normalized Hamming distance) are 0.0091 for the modified version and 0.0195 for the compressed version. Such small distances

normally indicate that the inputs are legitimate. The distances even imply that the modified version looks more authentic than the compressed version. In order to exclude insignificant but malicious distortion, a tight threshold can be used for hash comparison. However, that will also exclude some legitimate distortion, thus decreases the robustness. Therefore, conventional image hash algorithms are not suitable for applications with high security requirements.

(a) Modified Lena (b) Compressed Lena

Fig. 2. The minor modification problem.

The research on perceptual image hashing used to focus on robustness. The earliest work was probably proposed by Schneider and Chang in 1996 [6], based on the image histogram. Later in 1999, Fridrich proposed another algorithm based on randomized image block projection, which for the first time introduced the use of a secret key during hash generation [7, 8]. New algorithms come up with novel ways of feature extraction, such as [9–11, 5], and they typically strive for better robustness. For example, the radon transform based algorithm by Lefèbvre *et al.* [12] and the Fourier-mellin transform based algorithm by Swaminathan *et al.* [13] claim to have relatively good resistance to rotation and some other geometric distortion. Another research topic of interest is the security of the key, see e.g. [14, 15]. Nevertheless, the issue concerned in this work has never been specifically addressed.

3 A Secure Image Hash Algorithm

In order to alleviate the minor modification problem, we propose to design an image hash algorithm by a block-based approach. That means, we consider an image block as the basic unit for authenticity protection. We evenly divide an image into blocks, and apply a *"block" hash algorithm* to each block. The final hash value is the concatenation of all block hash values. In this way, malicious modification is restricted up to *the scale of a block.*

A straight-forward way to construct a block-based image hash algorithm is to apply an existing image hash algorithm to image blocks instead of the whole image. However, this approach might dramatically increase the hash size and the computational cost. A conventional image hash algorithm may have a hash size up to 1 kb. If an image has 64 blocks, it costs 64 kb to store the whole hash value. Besides, a large hash size also influences the speed of analysis for large-scale applications. Therefore, it is necessary to design block-based image hash algorithm specifically. The goal of this work is to design such an algorithm with a good balance between the performance and the cost.

Before we describe the proposed algorithm, we need to define the performance of such block-based algorithms in general. Since the algorithm is applied to image blocks, the performance of such an algorithm is defined as how well it protects the content of a block unit. Therefore, the size of the block plays an important role in the design. If the block is too small, the hash size becomes extremely large. Another observation is that a block begins to lose perceptual meaning if the size is too small. In the extreme case, a block shrinks to a point and has no perceptual meaning. Therefore, the block size must be carefully chosen. On the other hand, the perceptual importance of a block is also relative to the size of the image. For example, a 64×64 block may not be significant in a 2048×2048 image, but it is significant in a 512×512 image. It means we need to fix the block size and the image size when defining the authentication performance.

Based on these considerations, we define the protection level as the ratio between the block dimension and the default image dimension. The default image dimension is not the dimension of the original input image, but the maximum dimension before feature extraction. The protection level of our proposed algorithm is 64/512. We use a block size of 64×64 pixels. Before feature extraction, the image is resized so that the maximum dimension equals 512 pixels.

The basic idea of the proposed algorithm is to generate a hash value from low frequency phases of the two-dimensional discrete Fourier transform (DFT) coefficients of an image block. It begins with a pre-processing stage. An input image is first converted to gray and resized by bicubic interpolation to make the maximum dimension equal to 512 pixels. The resulted image is smoothed by a 3×3 average filter and processed by histogram equalization. These steps have several effects: 1) reduce the dimensionality of the feature space; 2) limit the hash length and the computation time; 3) remove insignificant noise and increase robustness; 4) synchronize the image size. The preprocessed image is padded with zero to make the size equal to multiples of 64.

The rest of the algorithm is applied to image blocks of 64×64 pixels. They are explained in detail below. The block hash values are concatenated to form the final hash value.

3.1 Feature Extraction from Image Blocks

The feature extraction is applied to image blocks. It works in the DFT domain. The DFT is an orthogonal transform. Since the coefficients are uncorrelated,

there is low redundancy in the extracted features. The two-dimensional DFT of an $M \times N$ image block $x_{m,n}$ is defined as

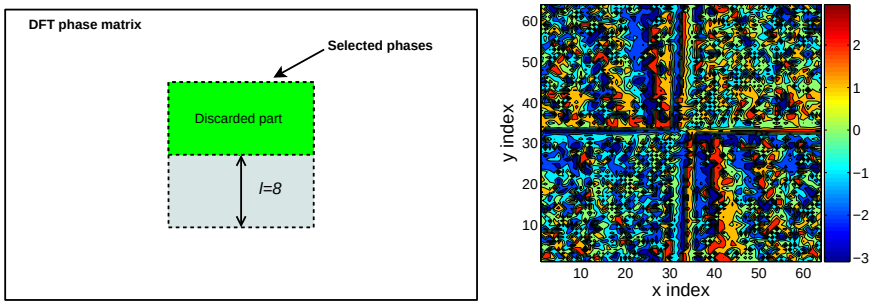
$$X_{k,l} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x_{m,n} \exp\left(\frac{-j2\pi nk}{N} + \frac{-j2\pi ml}{M}\right),$$
$$k = \{0, 1, \ldots, N-1\}, l = \{0, 1, \ldots, M-1\} .$$

The extracted feature is the *phase information* of the coefficients

$$\angle X_{k,l} \in [0, 2\pi) .$$

It is well known that the phase is critical in the representation of natural images, see e.g. [16–18].

After an image block is transformed by a 2D-DFT, the coefficient matrix is organized to have the zero frequency (DC) coefficient in the center. Since low frequency coefficients are less likely to be affected by incidental distortion, an initial feature vector is formed by low frequency phases. For implementation simplicity, the phases within a central square of width $2l+1$ are extracted, where l is an integral parameter that controls the length of the feature vector. This is illustrated in Fig. 3.



(a) Diagram (b) The phase map of an image block

Fig. 3. Selection of low frequency phases.

3.2 Feature Reduction and Randomization

The phase matrix in the frequency range specified by l is processed to compose the final feature vector. In our algorithm we set $l = 8$. Since pixel values are real numbers, the DFT coefficient matrix is conjugate-symmetric. Therefore, about half of selected phases are redundant. The phase matrix is divided into two parts and the redundant part is discarded, as shown in Figure 3. The zero phase of the DC coefficient is also discarded. There are 144 phase values left. They will be randomized and compressed. The randomization part requires a cryptographically

secure pseudo-random bit generator (PRBG). It generates uniformly distributed pseudo-random data from a secret key. It can be implemented by e.g. running the block cipher AES in the counter mode [1, 19].

Specifically, there are two randomization steps and two compression steps (Fig. 4). First, 144 phase values are combined into a column vector v . This vector is subjected to a secret permutation p generated by the secure PRNG. The second step is compression. A new feature vector v' is generated from the permuted one $p(v)$ by computing the mean of every 2 elements

$$v'_i = p(v)_{2i} + p(v)_{2i+1},\ i = 0, \cdots, 71. \tag{1}$$

This step not only makes the final hash value more compact, but also increases robustness and security. The third step is dithering. The final feature vector f is derived by adding a dither sequence s to v'; this step is motivated by Johnson and Ramchandran's work [20]. The dither sequence is generated by the secure PRNG. The elements of the dither sequence are uniformly distributed between 0 and 2π, and the addition operation is computed modulo 2π

$$f_i = (v'_i + s_i) \bmod 2\pi,\ i = 0, \cdots, 71\ . \tag{2}$$

These steps make the hash value highly dependent on the secret key. The last step is quantization of the feature vector f. Because legitimate distortion is likely to cause slight changes in DFT phases, coarse quantization can be applied to gain robustness. For implementation simplicity, an n-bit uniform quantizer with Gray coding is used. The parameter n controls the quantization accuracy. We use $n = 2$.

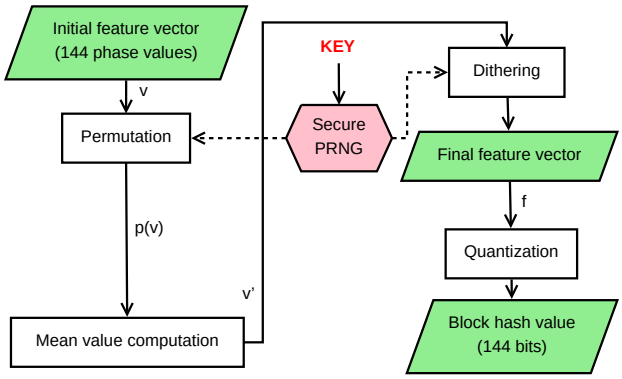


Fig. 4. Feature vector processing.

3.3 Hash Comparison

A metric must be defined to measure the distance between hash values. The hash distance metric used in the proposed scheme is the bit error rate (BER),

or the normalized Hamming distance. It is defined as

$$d_{xy} = \frac{1}{N} \sum_{i=0}^{N-1} |x_i - y_i| \ ,$$

where x and y are two binary vectors of length N. The (block) hash distance is compared with a threshold. The two images (or blocks) are considered similar if the distance is below the threshold. In this work, we mainly consider the similarity between image blocks.

4 Experiment Results

The proposed algorithm has been extensively tested. The results are shown in this section. We consider the performance in terms of robustness, discrimination, and key dependence. A database of natural scene photos[1] are used in the tests. It consists of different genres such as architecture, art, humanoids, landscape, objects, vehicles, etc. The image resolutions are larger than 1280×1280 pixels. All tests are performed on image block level, except for the key dependence test.

4.1 Robustness Test

A good algorithm is robust against legitimate distortion. We consider a few kinds of distortion as legitimate – JPEG compression, isotropic down-scaling, Gaussian smoothing, and additive white Gaussian noise (AWGN). They are commonly encountered in practice. The hash value is expected to be insensitive to these operations. In this test, we generate distorted versions for 900 original images in the database according to Table 1, and compute all the hash values. For each pair of original block and its distorted version, we compute the average block hash distance. The results are listed in Tables 2–5.

Table 1. Legitimate distortion

Distortion name	Distortion level (step)
JPEG compression	Quality factor: 5 – 25(5), 30 – 90 (10)
Down-scaling	Scale ratio: 0.3 – 0.9 (0.1)
AWGN	Signal to noise ratio: 10 – 40 (5) dB
Gaussian smoothing	Window size: 3 – 13 (2)

The distortion levels are selected to slightly exceed the normal ranges in practice. The results show that except for some extreme cases, e.g., AWGN with 10 dB signal to noise ratio (SNR), or JPEG with quality factor 5, all the average hash distances are quite small and generally increase with the distortion

[1] www.imageafter.com

Table 2. Robustness test for JPEG compression.

Quality factor	5	10	15	20	25	30
Average block hash distance	0.218	0.168	0.144	0.128	0.117	0.108
Quality factor	40	50	60	70	80	90
Average block hash distance	0.097	0.088	0.081	0.072	0.056	0.039

Table 3. Robustness test for down-scaling.

Scale ratio	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Average block hash distance	0.096	0.065	0.042	0.050	0.058	0.067	0.042

level. Gaussian smoothing has the least influence to the hash value – the distance is 0.031 for all distortion levels. This demonstrates the good robustness of the extracted features. For down-scaling, the hash distance does not monotonically increase with the distortion level. It is possibly because that the scaling operation and the resizing operation (in the pre-processing stage) involve pixel sub-sampling and interpolation where noise is not always proportional to the scale ratio; moreover, they may lead to slight changes of the aspect ratio and displacement of image blocks, thus introduce some randomness in the final results. Nevertheless, the distances are small compared to JPEG and AWGN.

4.2 Discrimination Test

In this test, we compare image blocks of different content. A pair of blocks is randomly chosen from two images in the same category, and their block hash values are compared. The purpose of this test is to see if the hash value is able to distinguish perceptually different blocks. It also shows the algorithm's potential to detect malicious modification on the block level.

Although randomly selected blocks are likely to be different, sometimes we meet similar blocks. Therefore, a metric is needed to decide whether two blocks are really different. We use the well-known structural similarity (SSIM) [21] for judging the ground truth. The SSIM is a widely used similarity metric for images. It compares two images, and returns a score between 0 (no similarity) and 1 (full similarity). We apply SSIM to image blocks, and compare the similarity score with a predefined threshold t. In our experiment, we set $t = 0.7$. Those block pairs, whose SSIM scores are below the threshold, are considered as perceptually different. A large hash distance is expected for different blocks. We compute the hash distance for about 800 thousand pairs of different blocks. The average hash distances for some image types are listed in Table 6. The overall average hash distance is 0.437.

Table 4. Robustness test for additive white Gaussian noise.

Signal to noise ratio (dB)	10	15	20	25	30	35	40
Average block hash distance	0.214	0.171	0.135	0.106	0.083	0.064	0.048

Table 5. Robustness test for Gaussian smoothing.

Window size	3	5	7	9	11	13
Average block hash distance	0.031	0.031	0.031	0.031	0.031	0.031

Table 6. Average hash distance between different blocks.

Image type	Average block hash distance (standard deviation)	Overall average (standard deviation)
Architecture	0.432 (0.040)	0.437 (0.043)
Art	0.441 (0.042)	
Landscape	0.435 (0.046)	
Objects	0.447 (0.042)	
Humanoids	0.440 (0.042)	
Vehicles	0.441 (0.043)	
...	...	

The discrimination performance is measured by the average block hash distance. Intuitively, if two image blocks are randomly chosen, they are most likely to be "half similar". If the hash distance uniformly represents the similarity, on average it is about half of the full distance, i.e., 0.5. From this point of view, the proposed hash achieves good discrimination. The deviation from the ideal situation can be due to several reasons. First, the small size of the block hash limits the discrimination power. Second, since the test is carried out for the same type of images, the bias can be understood as the essential similarity among images of the same kind. The results show that it is unlikely to find different blocks with similar hash values. Therefore, attempts to replace a block with another is unlikely to succeed without being detected.

4.3 Hypothesis Test

In a typical application scenario, after the block hash distance d is computed, it is compared with a threshold T. The decision is made from two hypotheses:

- $\mathbb{H}_0$ – the blocks correspond to different content;
- $\mathbb{H}_1$ – the blocks correspond to similar content.

If $d \leq T$, we choose $\mathbb{H}_1$; otherwise we choose $\mathbb{H}_0$.

The overall performance of the algorithm can be characterized by the true positive rate P_d and the false positive rate P_f. They are defined as:

- $P_d = \text{Probability } \{d \leq T | \mathbb{H}_1\}$;
- $P_f = \text{Probability } \{d \leq T | \mathbb{H}_0\}$.

When the threshold T is increased, the hash is likely to tolerate more distortion, but that also increases the chance of false positive. A good algorithm should suppress the false positive rate while maintaining a high true positive rate. Previously, we have generated about 1.4 million positive cases (in the robustness

test) and 0.8 million negative cases (in the discrimination test) on the block level. By adjusting the value of T, we compute P_d and P_f and plot their trends, as shown in Fig. 5. One can see that P_d increases with T, while P_f is negligible until $T = 0.3$.

In order to choose the most suitable threshold value, we also take into account of the false negative rate P_m, which is defined as

– $P_m = \text{Probability } \{d > T | \mathbb{H}_1\}$.

By definition, $P_m = 1 - P_d$. We can see that P_f and P_m are contradicting requirements. Depending on the severity of false positive and false negative, different applications give their own bias towards P_f or P_m. By default, we can choose the equal error rate point (EERP), where $P_f = P_m$, as the working point. In our case, $T = 0.344$ is the EERP, where $P_d = 0.977$, and $P_f = P_m = 0.023$. We also plot the relationship between P_d and P_f, known as the receiver operating characteristic (ROC) curve, in Fig. 6. The ROC curve illustrates the good overall performance from another angle – the proposed algorithm offers efficient trade-offs between P_d and P_f.

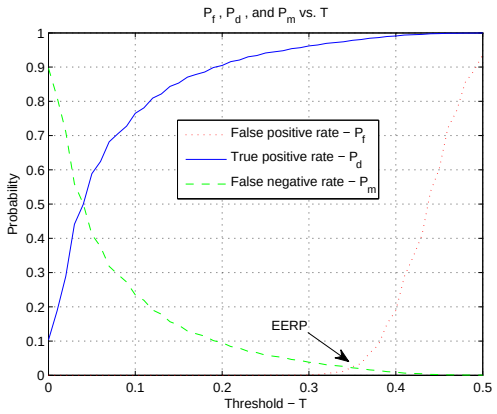
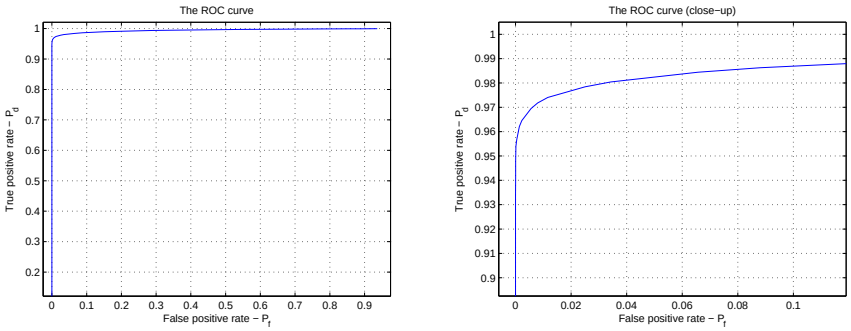


Fig. 5. True positive rate, false positive rate, and false negative rate.

4.4 Key Dependence Test

All the above tests use a default key for generating hash values. When the algorithm is used as a message authentication code, the hash value must be highly dependent on the key. In this test, we use 900 images to validate the key dependence property. For each image, we generate 100 hash values using different keys. They are pair-wise compared. There are 4950 hash comparisons for each image, and about 5 million comparisons in total. If two different keys are used for the same image, the corresponding hash values should be totally different, as



(a) The complete curve (b) A close-up

Fig. 6. The receiver operating characteristics.

if they correspond to different content. The average hash distances for all images are plotted in Fig. 7. All the average distances are centralized around 0.5 with a very small dynamic range within 0.4997 – 0.5004. This demonstrates the good randomization mechanism of the proposed scheme.

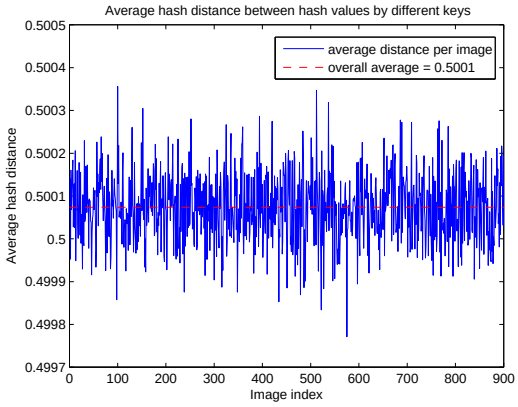


Fig. 7. Key dependence test.

5 Conclusion and discussion

In the multimedia domain, a challenge to content authentication is, that the same content may have different digital representations. A potential solution is perceptual hashing, because it provides robustness against incidental distortion, such as compression. However, due to the minor modification problem, conventional image hash algorithms only protect the overall content of an image. In

this work, we propose an image hash algorithm with a much higher security level. The algorithm aims at protecting the authenticity of image blocks. We define the protection level as 64/512, which typically corresponds to 1/48 area of a $4:3$ image. For each image block, features are computed from the phase values after the discrete Fourier transform. Experiments show that the hash has strong robustness against JPEG compression, scaling, additive white Gaussian noise, and Gaussian smoothing. The hash algorithm is key dependent, thus can be used as a message authentication code. Experiments confirm that the hash value is highly dependent on the key. The hash size is 144 bits per 64×64 block (6912 bits per $4:3$ image) after pre-processing. In spite of such a compact size, hypothesis test shows that we achieve very efficient trade-offs between the false positive rate and the true positive rate.

In our experiment, distortions such as rotation and translation are not taken into account, because it is questionable to consider them as authentic in the content protection circumstance. They typically generate much higher distortion than other non-geometric manipulations, thus give chance to malicious modification. In general, the performance of content authentication significantly degrades if geometric distortion is allowed.

What is not discussed in this work is the security of the key, see [22]. Given some known image/hash pairs, it is not obvious how much effort is needed to derive the key of our algorithm. In practice, we advise not using the same permutation and dithering for all blocks in an image. An in-depth security analysis will be given in the future.

References

1. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press (1996)
2. Coskun, B., Memon, N.: Confusion/diffusion capabilities of some robust hash functions. In: Proc. of 40th annual conference on information sciences and systems, Princeton, USA (Mar. 2006)
3. Weng, L., Preneel, B.: Attacking some perceptual image hash algorithms. In: Proc. of IEEE International Conference on Multimedia & Expo. (2007) 879–882
4. Voloshynovskiy, S., Koval, O., Beekhof, F., Pun, T.: Conception and limits of robust perceptual hashing: towards side information assisted hash functions. In: Proc. of SPIE. Volume 7254. (February 2009)
5. Weng, L., Preneel, B.: Shape-based features for image hashing. In: Proc. of IEEE International Conference on Multimedia & Expo. (2009)
6. Schneider, M., Chang, S.F.: A robust content based digital signature for image authentication. In: Proc. of International Conference on Image Processing (ICIP96). Volume 3. (1996) 227–230
7. Fridrich, J.: Robust bit extraction from images. In: Proc. of IEEE International Conference on Multimedia Computing and Systems. Volume 2. (1999) 536–540
8. Fridrich, J., Goljan, M.: Robust hash functions for digital watermarking. In: Proc. of International Conference on Information Technology: Coding and Computing. (2000)

9. Venkatesan, R., Koon, S.M., Jakubowski, M., Moulin, P.: Robust image hashing. In: Proc. of IEEE International Conference on Image Processing. Volume 3., Vancouver, CA (2000) 664–666
10. Mihçak, M.K., Venkatesan, R.: New iterative geometric methods for robust perceptual image hashing. In: Proceedings of ACM Workshop on Security and Privacy in Digital Rights Management, Philadelphia, PA, USA (Nov. 2001)
11. Monga, V., Evans, B.: Robust perceptual image hashing using feature points. In: Proc. of IEEE International Conference on Image Processing. Volume 1. (2004) 677– 680
12. Lefèbvre, F., Macq, B., Legat, J.D.: RASH: RAdon Soft Hash algorithm. In: Proc. of the 11th European Signal Processing Conference. Volume 1., Toulouse, France (September 2002) 299–302
13. Swaminathan, A., Mao, Y., Wu, M.: Robust and secure image hashing. IEEE Transactions on Information Forensics and Security **1**(2) (June 2006) 215–230
14. Swaminathan, A., Mao, Y., Wu, M.: Security of feature extraction in image hashing. In: Proc. of 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing, Philadelphia, PA, USA (Mar. 2005)
15. Radhakrishnan, R., Xiong, Z., Memon, N.: On the security of the visual hash function. Journal of Electronic Imaging **14** (2005) 10
16. Oppenheim, A., Lim, J.: The importance of phase in signals. Proceedings of the IEEE **69**(5) (May 1981) 529 – 541
17. Gegenfurtner, K., Braun, D., Wichmann, F.: The importance of phase information for recognizing natural images. Journal of Vision **3**(9) (2003) 519a
18. Ni, X., Huo, X.: Statistical interpretation of the importance of phase information in signal and image reconstruction. Statistics & Probability Letters **77**(4) (2007) 447–454
19. Barker, E., Kelsey, J.: Recommendation for random number generation using deterministic random bit generators. Technical report, NIST (2007)
20. Johnson, M., Ramchandran, K.: Dither-based secure image hashing using distributed coding. In: Proc. of IEEE International Conference on Image Processing. Volume 2. (2003) 751–754
21. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4) (April 2004) 600–612
22. Mao, Y., Wu, M.: Unicity distance of robust image hashing. IEEE Transactions on Information Forensics and Security **2**(3) (September 2007) 462–467