ATTACKING SOME PERCEPTUAL IMAGE HASH ALGORITHMS

Li Weng, Bart Preneel

Katholieke Universiteit Leuven
Department of Electrical Engineering
Kasteelpark Arenberg 10, 3001 Heverlee, Belgium

ABSTRACT

Perceptual hashing is an emerging solution for multimedia content authentication. Due to their robustness, such techniques might not work well when malicious attack is perceptually insignificant. We designed an experiment and verified that some state-of-the-art image hash algorithms could not distinguish small malicious distortion and some authentic distortion. We proposed an enhancement framework as a remedy. It suggests extracting information from the content and combining it with the secret key to generate the perceptual hash, so that perceptually insignificant information can be protected.

1. INTRODUCTION

In the information era, multimedia content is increasingly being produced and distributed in digital form. Meanwhile, with the wide availability of editing software, content modification is no longer a difficult job. A negative consequence is that the trustworthiness of multimedia data is often questioned. Therefore, technical solutions are desired to ensure the integrity of multimedia content. One possible solution is to use cryptographic hash functions. However, they are sensitive to any bit change, thus are not suitable for the multimedia domain where the same content often exists in many digital forms – different formats, different quality, etc., and they are still considered authentic even after moderate processing, such as common enhancement and slight geometric distortion. Apparently, it is impractical to compute and store conventional hash values for all possible cases. Therefore, it is expected to have hash algorithms that only depend on the content and tolerate content-preserving distortion, i.e., for the same or similar content, the hash is always the same or similar, regardless of the digital form; and the hash changes drastically only after the content is significantly changed. This motivates the research in perceptual hash algorithms.

This work was supported in part by the Concerted Research Action (GOA) Ambiorics 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 IBBT (Interdisciplinary Institute for Broad-Band Technology), a research institute founded by the Flemish Government in 2004, and the involved companies and institutions (Philips, IPGlobalnet, Vitalsys, Landsbond onafhankelijke ziekenfondsen, UZ-Gent).

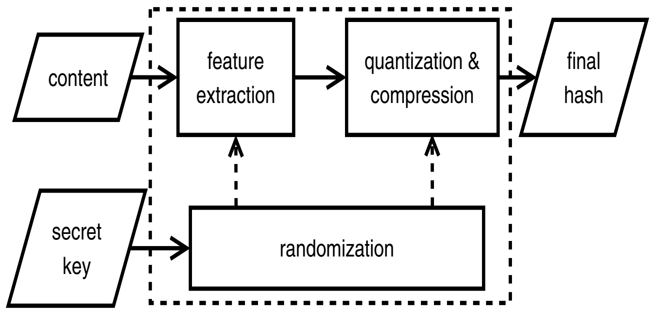
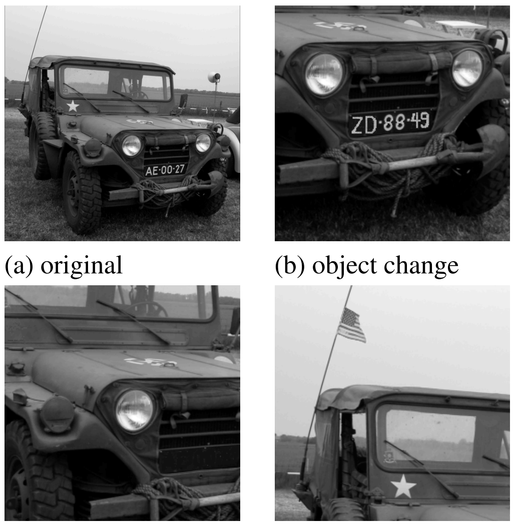


Fig. 1. Basic components of a perceptual hash algorithm.

A perceptual hash algorithm achieves certain *robustness* by extracting perceptual features from multimedia content. Ideally it tolerates moderate levels of content-preserving processing, and is only sensitive to perceptually significant content modification. Besides robustness, it also differs from a generic data hash by the incorporation of a secret key. Since the features are publicly known, using them alone is not secure – an attacker may perform two attacks: 1) counterfeiting both the content and the hash; 2) gradually introducing tiny changes until the content is severely distorted, while preserving the hash. In order to prevent such malicious manipulations, key-based randomization is used in the feature extraction. A secret key is used to generate the hash. For the same data, a different key should result in a completely different hash. In that case, the first attack is impractical; the second one is also difficult because the attacker cannot verify whether the attack is successful without knowing the key.

The basic components of a perceptual hash algorithm are drawn in Fig. 1. In the algorithm, perceptually significant features are extracted from the content, then quantized and compressed to a short binary string. For content authentication, the original hash is pre-computed and sent to the authenticator; knowing the secret key, the authenticator computes the hash again from the file in question and compares it with the received hash. In practice, the hash value does not remain exactly the same but changes slightly in case of authentic distortion. Therefore the hash comparison is usually considered as a hypothesis testing problem: a similarity measure or distance metric between the received hash and the computed one


(a) original (b) object change

(c) object removal (d) object insertion

Fig. 2. The original car and malicious attacks (zoomed).

is calculated and compared with a threshold; if it is below the threshold, the tested version is considered as authentic, otherwise inauthentic. The commonly used metrics are the Euclidean distance and the Hamming distance.

Many perceptual hash algorithms have been proposed with good robustness and security (in the sense of guessing the key). However, there has been little work to show *to what extent* malicious distortion can be detected. Note that malicious distortion might not be perceptually significant. In particular, if a malicious change is perceptually small (see Fig. 3), is it still detectable? We design an experiment to answer this question. Fig. 2a shows a gray-scale image of a car [1]. We perform three malicious attacks: modify the car number (Fig. 2b); remove the car number and the star label (Fig. 2c); insert a flag (Fig. 2d). Although these changes are perceptually small, they significantly change the expressed information. Therefore, the resultant content might be considered as inauthentic. However, a robust hash tends to tolerate them and consider attacked versions as authentic. In the following, we test some state-of-the-art perceptual image hash algorithms to see if they can detect these malicious changes.

2. THE DESIGNED EXPERIMENT

The algorithms under test are the work by Swaminathan *et al.* [1], Monga *et al.* [2], and Yu *et al.* [3][2]. They were recently proposed and are based on different principles, thus might give a representative view of the state of the art. We apply the aforementioned malicious attacks to Fig. 2(a), as well as some authentic manipulations: 1) rotation by $1°$, $2°$; 2) cropping by 1%, 2%; 3) JPEG compression with quality factor QF=10, 20; 4) additive white Gaussian noise (AWGN) with variance $\sigma^2 = 50,\ 100$. A metric is computed from the resulted hash vectors to see if the two classes of distortion can be distinguished successfully. If for authentic distortion the metric is always below a threshold and otherwise it is always above the threshold, then this algorithm is good for authentication. Since the detection of malicious attacks only depends on the feature extraction stage, we do not perform quantization and compression, which tend to increase robustness. We also ignore randomization for it only deals with security. Denoting the hash vector of the original image by H_o and the one for a modified version by H_{mod}, we use a relative Euclidean distance as the metric, defined as:

$$\frac{\|H_o - H_{mod}\|_2}{\|H_o\|_2}\ .$$

The following setting is used: gray-scale images are resized to 512×512 before running the algorithm; the cropping is conducted by retaining the central part and removing the boundaries; after rotation, the image is cropped to remove the zeros.

We first test the algorithm by Swaminathan *et al.* [1]. In particular, we implement its Scheme 1, which is based on the Fourier transform. Basically, this algorithm applies Fourier transform to an image, and sums the magnitude values along a circle centered at zero frequency to get a hash element h_j. Since spatial translation does not affect the magnitude of Fourier transform, and a spatial rotation gives the same rotation in frequency domain, h_j is invariant to rotation and translation. The final hash is formed by the set $\{h_j\}$ computed from different circles. In our implementation, we uniformly choose 35 radii between normalized frequency 0.1 and 0.8; for each radii we uniformly take 360 samples along the circle. Finally we get a hash vector of 35 elements by summing the 360 magnitude values along each of the circles.

We perform all the manipulations and list relative Euclidean distances between the original and modified images in the first column of Table 1. The smallest metric is given by object insertion. Other malicious attacks produce metrics which are quite close to those by authentic distortion. For example, object removal has a metric similar to AWGN 2 or cropping 2; object change is similar to AWGN 1. Moreover,

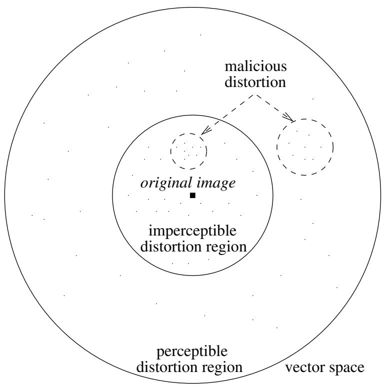

malicious distortion
original image
imperceptible distortion region
perceptible distortion region
vector space

Fig. 3. Imperceptible distortion vs. malicious distortion.

[1]Taken from www.imageafter.com, cropped and gray-scaled.

[2]This algorithm does not have key-based randomization.

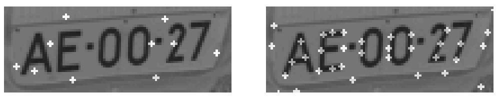
the largest metric is given by rotation 2. That means in order to tolerate authentic distortion, we also have to tolerate malicious attacks. Therefore, this algorithm cannot well distinguish authentic distortion and malicious attacks.

The second algorithm is by Monga *et al.* [2]. It is based on *feature points*. It adopts two-dimensional continuous wavelet transform to find feature points in an image and uses them as the hash. In general, human eyes respond strongly to corner like stimuli and points of high curvature. Bhattacherjee *et al.* [4] constructed "end-stopped" wavelets to capture this behavior. The end-stopped wavelet is a combination of a two-dimensional Morlet wavelet and a first derivative of Gaussian (FDoG). The Morlet wavelet can detect line structures in a specific orientation and the FDoG can detect the end-points of such lines. The following waveform is an end-stopped wavelet oriented along the *x*-axis and can be used to detect line ends and high curvature points in the vertical direction.

$$\psi_E(x,y) = \frac{1}{4} y \exp^{-\frac{1}{4}\left(x^2+y^2+k_0(k_0-2jx)\right)}$$

This algorithm creates K rotated versions of the above waveform to uniformly cover 180 degrees at a *fixed* scale and correlates each of them with the image. Only the magnitude of the correlation response is considered. At each pixel, the largest response among the K orientations is kept. A pixel is considered as a feature point candidate if it has the maximum correlation response within its neighborhood. Finally, only candidates with response above a threshold are kept as feature points. The above procedure is continuously applied to low-pass and order-statistic filtered versions of the image until the feature points are invariant. Because such a process reduces the extracted information, we skip it and only evaluate the first round of feature point extraction.

At first the wavelet is sampled in a dyadic fashion at scale 3, i.e., the sampling interval in space is $1/2^3$. We set $K = 18$, $k_0 = 6$ and the neighborhood as a 5×5 square. However, this algorithm cannot capture the car number at this scale, as shown in Fig. 4a. Until the scale is changed to 1, the numbers are approximately captured, as shown in Fig. 4b. Since the hash vector is produced by concatenating pixel values of the feature points, their relative positions should not change. To avoid misalignment, we only compare the feature points that capture the car number. We skip object insertion and removal, and assume they are distinguishable from authentic processing, because they either add or remove feature points, while authentic manipulations normally do not change the number of feature points. We only compare the metrics given by object change and those by authentic distortion. If in the car number region an attacked image has a different number of feature points than the original one, we exclude the extra points in the end of the longer hash. The results are listed in the second column of Table 1. Although object change gives the largest metric, thus can be distinguished from other distortion, all the metrics generally look too large, which makes the results less trustworthy. On the other hand, some bias is given to the algorithm, so its practical performance remains questionable.



(a) scale 3 (b) scale 1

Fig. 4. Feature points by Alg. 2 (denoted by "+").

The third algorithm was proposed by Yu *et al.* [3]. It is based on higher-order moments [5]. Higher order statistics are able to extract the non-Gaussianity of an image, which tends to be invariant after content-preserving processing. This algorithm resizes an image to 256×256 and divides it into non-overlapping 64×64 blocks. Each block is rearranged into a vector, then the following fourth order moment is calculated to arrive at a length 8193 real sequence.

$$\begin{aligned} & C_{4x}(m,n,t)|_{n=t=0} \\ & \quad = \frac{1}{K}\sum_{i=1}^{K} x(i)x(i+m)x(i+n)x(i+t)|_{n=t=0}, \\ & m = -K, \ldots, 0, \ldots, K \end{aligned}$$

A discrete cosine transform is applied to the sequence. The first 32 coefficients are reserved for each block to form the final hash. The test results are listed in the third column of Table 1. The metrics by object change and removal are far below those by most of the authentic distortion. The metric by object insertion is at the same order as those by rotation 1 and cropping 1. The highest metric is given by cropping 2. Again this algorithm fails to distinguish malicious attacks.

After the experiment, it is observed that in most cases the metrics by malicious attacks are below those by authentic distortion. That means the content manipulated by malicious attacks might be considered *more authentic* than the real authentic one! Therefore, we conclude that the tested algorithms might not be suitable to address the authentication scenario as described in our experiment, thus they must be used with great caution in similar occasions.

3. AN ENHANCEMENT FRAMEWORK

In the following, we propose a method to solve the car number modification problem. The aim is to authenticate Fig. 2(a) including the car number by a general perceptual hash. Recall that a secret key is used to generate the hash. An effective way to protect the car number is to use it as *part* of the key, i.e.,

$$hash\ key = F(secret\ key,\ car\ number)\ ,$$

where F is a function that combines the secret key and the car number, such as concatenation, XOR or a conventional hash, so that if the car number is changed, the hash key is also changed, and then the final hash will *drastically* change.

Table 1. Relative Euclidean distances by various distortion.

	Ref. [1]	Ref. [2]	Ref. [3]
object change	.0261	.6085	.0002
object removal	.0523	NA	.0040
object insertion	.0063	NA	.0203
rotation 1, by 1°	.0410	.1986	.0253
rotation 2, by 2°	.0607	.2672	.0469
cropping 1, by 1%	.0361	.3689	.0308
cropping 2, by 2%	.0503	.3968	.0645
JPEG 1, QF=10	.0327	.5040	.0057
JPEG 2, QF=20	.0182	.4364	.0017
AWGN 1, $\sigma^2 = 50$	.0314	.3538	.0087
AWGN 2, $\sigma^2 = 100$	.0536	.3574	.0099

Since the secret key is unknown to the attacker, incorporating the car number normally does not compromise the security level – guessing the hash key is still as difficult as guessing the secret key. In this way the car number can be successfully protected. For example, a typical scenario to apply this method might be the authentication of photos taken by traffic cameras where each photo contains a car number, so they become more convincing as proof in court. In order for this scheme to work in practice, the final hash must be sensitive to any bit flip in the hash key. Fortunately, this is usually true for contemporary perceptual hash algorithms.

In many cases it is not known in advance which part of the image needs to be protected; moreover, given an image part, there is no unique semantically extracted information. To cope with these issues, we further extend the above method to a more general framework, as shown in Fig. 5. The solution is to add an *information extraction stage* guided by an *information protection protocol*. The protocol is an agreement between the sender and the receiver on what information exists and needs to be protected. It dictates the information extraction stage to extract the corresponding features and represent them by a certain format. This stage can be a pattern recognition algorithm, or simply a human being. After the *information features* are extracted, they are fed into a key generation block F together with the secret key. In case of too many information features, they can be compressed first, e.g., by a conventional hash. Virtually this framework can be added to any perceptual hash algorithm as enhancement, so that perceptually small information can be protected as well.

4. CONCLUSIONS AND DISCUSSIONS

We designed an experiment to see if some perceptual image hash algorithms can distinguish small malicious attacks from authentic manipulations, and concluded that in most cases they fail this task. We proposed an enhancement framework to solve this problem. It suggests extracting information from the content and combining it with the secret key to generate the perceptual hash. In this way, perceptually insignificant

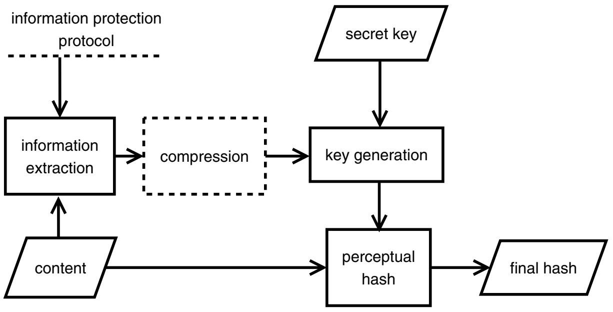


Fig. 5. The enhancement framework.

information can also be protected. However, there is some limitation – e.g., it might not be easy to establish the information protection protocol in some cases; the information extraction stage might not be efficient; the length of the secret key determines how much information can be protected without compression.

There also exist other ways to solve the problem of Fig. 3. For example, regions of interest can be extracted first, then they can be separately hashed and stored, or hashed together with global image features or their hash. The former might work for a perceptual hash, but it may not work well when the salient regions are small, and it requires additional storage. The latter is only practical for a conventional hash, since a perceptual hash cannot work on arbitrary concatenation of data. However, the output of a conventional hash is quite fragile, even if techniques such as forward error correction are applied to the data in advance. Compared to these solutions, our solution does not require additional storage and remains strong robustness, thus can be an alternative choice.

5. REFERENCES

[1] Ashwin Swaminathan, Yinian Mao, and Min Wu, “Robust and secure image hashing,” *IEEE Transactions on Information Forensics and Security*, vol. 1, 2006.

[2] Vishal Monga and Brian L. Evans, “Robust perceptual image hashing using feature points,” in *Proc. of IEEE Conf. on Image Processing*, 2004.

[3] Longjiang Yu, Martin Schmucker, Christoph Busch, and Shenghe Sun, “Cumulant-based image fingerprints,” in *Proc. of SPIE-IS&T Electronic Imaging*, 2005.

[4] S. Bhatacherjee and P. Vandergheynst, “End-stopped wavelets for detecting low-level features,” in *Proc. of SPIE Wavelet Appl. in Sig. & Image Proc.*, 1999.

[5] J.M. Mendel, “Tutorial on higher-order statistics (spectra) in signal processing and system theory: theoretical results and some applications,” in *Proc. of the IEEE*, 1991.