Multimedia Systems (2005) 11(2): 159–173
DOI 10.1007/s00530-005-0199-y

REGULAR PAPER

Chun-Shien Lu · Chao-Yong Hsu

Geometric distortion-resilient image hashing scheme and its applications on copy detection and authentication

Published online: 17 October 2005

Abstract Media hashing is an alternative approach to many applications previously accomplished with watermarking. The major disadvantage of the existing media hashing technologies is their limited resistance to geometric attacks. In this paper, a novel geometric distortion-invariant image hashing scheme, which can be employed to perform copy detection and content authentication of digital images, is proposed. Our major contributions are threefold: (i) a mesh-based robust hashing function is proposed; (ii) a sophisticated hash database for error-resilient and fast matching is constructed; and (iii) the application scalability of our scheme for content copy tracing and authentication is studied. In addition, we further investigate several media hashing issues, including robustness and discrimination, error analysis, and complexity, with respect to the proposed image hashing system. Exhaustive experimental results obtained from benchmark attacks confirm the excellent performance of the proposed method.

Keywords Authentication · Copy detection · Geometric distortion · Hash · Robustness · Searching

1 Introduction

With the advances in multimedia and networking technologies, it has become easy to copy original material completely and distribute illegal copies rapidly over the Internet. In order to trace the unauthorized use of digital contents, media hashing technologies have been applied to digital content management. In contrast to data hiding, the main characteristic of media hashing is its non-invasive property, which means that no information has to be embedded in the digital content. On the other hand, a hash sequence for specific media data needs to be extracted to obtain a condensed representation. Technologies analogous to media hashing have been reported in the literature, including fingerprinting, digital signature, and passive/non-invasive watermarking. The major feature that distinguishes media hashing from watermarking is that the former measures "similarity" and needs to work together with a feature database, while the latter measures "originality" and can operate as a standalone system. On the other hand, media hashing is also similar to media retrieval in that both need to transform media data into a short string for the sake of compact representation. The technical difference between them is that media hashing must resist (either malicious or incidental) attacks. Therefore, several applications that call for robust identification of media contents require the use of robust media hashing methods [1, 2]. The technical requirements for media hashing cannot be satisfied by means of traditional cryptographic hashing functions because even a one bit error in a hash sequence can lead to entirely different contents. However, limited distortions are not harmful to the visual quality and commercial value of multimedia data.

Many existing media hashing methods were developed for audio identification [2] and video authentication [3, 4]. In this paper, we will focus on image hashing. First of all, a review of the literature on image hashing will be presented in the following. In 1998, Chang et al. proposed a wavelet-based Replicated IMage dEtector (RIME) [5] to search for unauthorized image copying on the Internet. They used color features only to represent an image and then used the vector quantization (VQ) technique to index images. Their system cannot resist extensive geometric distortions. To speed up the detection of near-replicas of images in their RIME system, Chang et al. proposed a new clustering approach [6] that can improve input/output efficiency by clustering and retrieving relevant information sequentially on and from the disk. Recently, Meng and Chang [7] used multi-scale color and texture features to characterize images and employed the dynamic partial function (DPF) to measure the perceptual similarity of images. Basically, the idea behind DPF [8] is to dynamically activate partial features (thereby discarding the other larger feature distance) in order to reveal the similarity between a pair of images. Although DPF

C.-S. Lu (✉) · C.-Y. Hsu
Institute of Information Science, Academia Sinica Taipei, Taiwan 115, Republic of China
E-mail: {lcs, cyhsu}@iis.sinica.edu.tw

outperformed traditional distance metrics, the adopted image feature was global in that resistance to geometric distortions was inherently limited. In [9, 10], the digital signature was proposed for image authentication. Lin and Chang [9] created the mutual relationship of pairwise block-DCT coefficients to distinguish JPEG compressions from malicious modifications. Lu and Liao [10] built the so-called "structural digital signature," based on the multiscale structure of the wavelet transform, to tolerate incidental manipulations and reflect intentional manipulations. However, the ability to resist geometric manipulations was lacking [9, 10]. In [11, 12], Fridrich and Goljan proposed a robust/visual hash function for digital watermarking. Their hash digests of digital images were created by projections of DCT coefficients to key-dependent random patterns. In [13], Venkatesan et al. proposed an image hashing technique, which includes (i) random tiling of an image; and (ii) hash generation based on statistical feature extraction of tiles. However, the two methods [11, 13] only achieve limited resistance to geometric distortions. In [14–16], image-hashing methods were proposed based on the Radon transform and exploited its affine invariance. However, resistance to geometric distortions was found to be greatly limited if the incoming attacks went beyond affine distortions. In [17], Mihcak and Venkatesan proposed an iterative geometric image hashing method, which contains two major steps. First, an image is converted into a binary image by thresholding the lowest-frequency subband in the wavelet domain in order to identify the geometric shapes. Second, iterative filtering is applied to the resultant binary image to obtain the hash by enhancing geometrically strong components and erasing geometrically weak components. This method can only withstand slight geometric distortions. In [18], Kim proposed an image copy detection scheme that employs ordinal measures of AC coefficients in the 8×8 DCT domain; i.e., the magnitudes of the AC coefficients in a block are ranked in descending order to represent an image. Extensive signal processing attacks were conducted to test the method's robustness and discrimination in the case of a large database. However, this system basically could not resist geometric distortions.

It is evident from the above survey that a common disadvantage of the existing image hashing techniques is their limited robustness against geometric distortions (for instance, resistance to rotations is restricted to very small angles). In view of this fact, the purpose of this paper is to deal with this challenging problem. We shall propose a robust mesh-based image hashing scheme for content copy detection and tracing in a large database. Our major contribution is the capability of achieving robustness against extensive geometric distortions (e.g., standard benchmarks like Stirmark3.1 and Stirmark4.0 [19, 20]). Although the concept of image meshing has been applied to watermarking before [21], we consider the stability of mesh generation, which is closely related to mesh-based applications. Consequently, we present here a robust mesh extraction technique that cannot easily degrade the performance of mesh-based hashing. We also present a robust mesh-based hash extraction technique that considers content position-dependent features. Extensive results obtained from benchmark attacks further confirm the robustness of the proposed scheme.

In addition to robustness, the practical use of an image hashing system requires an ability to quickly search a large database. In this paper, we will also show how an efficient hash database can be built to facilitate fast hash matching. Moreover, we shall present error analyses and investigate the complexity, granularity, and scalability of the proposed image hashing system. In particular, we will demonstrate how our image hashing system can be applied to content authentication.

The remainder of this paper is organized as follows. Sect. 2 discusses the difference between cryptographic hashing and media hashing, and states the media hashing problem that need to be dealt with. In Sect. 3, the proposed image hashing system, which includes mesh generation, mesh-based hash generation, coarse-to-fine hash database construction, and fast hash matching, is described. Media hashing issues, including robustness, error analysis, complexity, and scalability, are discussed in Sect. 4. Extensive experimental results are given in Sect. 5 to verify the performance of our scheme. Finally, concluding remarks are given in Sect. 6.

2 Problem statement

Media hashing is recognized as an alternative approach to several applications that were previously performed using digital watermarking. Here, a scenario of copy detection and tracing is given to outline how an image hashing approach can be employed to manage digital image contents. Given an image owned by its creator, an image copy detection system needs to find out whether illegal copies of the image exist on the Internet and, if they exist, return a list of suspect URLs. This content searching strategy can be accomplished by means of image hashing, and the output of the hashing system can offer owners information about unauthorized use of their precious media data.

Referring to the image space shown in Fig. 1, let $\mathbf{I}$ denote an image, and let $\mathcal{X}$ denote the set of images that are modified from $\mathbf{I}$ by means of content-preserving operations (e.g., filtering, compression, and geometric distortions) and are defined as being perceptually similar to $\mathbf{I}$. Although perceptual similarity is still an ill-posed concept [8, 22], we will propose a hash-based matching metric in the next section for image searching. We further use $\mathcal{Y}$ to denote those images that are modified from $\mathbf{I}$ but can hardly be recognized as originating in $\mathbf{I}$. For example, severe noise adding and severe cropping are two representative attacks that can generate the elements of $\mathcal{Y}$. In addition, we denote using $\mathcal{Z}$ a set which contains all the other images that are irrelevant to $\mathbf{I}$ and its modified versions. Consequently, $\{\mathbf{I}\} \cup \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$ is a case that forms an entire image space.

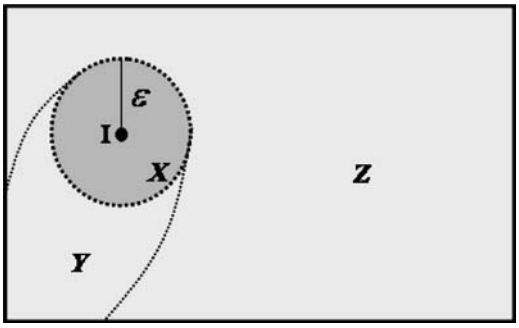


Fig. 1 The image space. **I** is an element in the image space. $\mathcal{X}$ denotes the set of images modified from **I** that are still perceptually similar to **I**. $\mathcal{Y}$ denotes the set of images modified from **I** that are perceptually different from **I**. $\mathcal{Z}$ is the set of images that are irrelevant to **I**

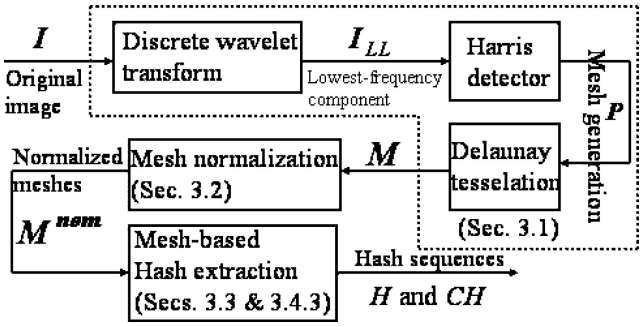


Fig. 2 Block diagram of the proposed mesh-based image hashing system

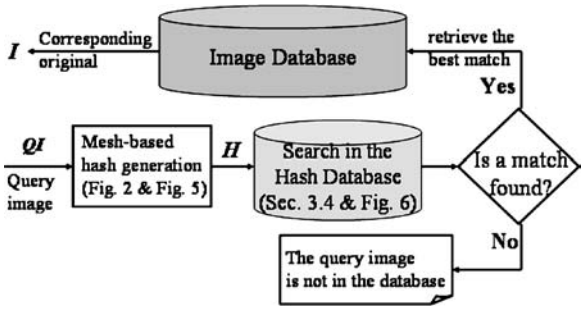


Fig. 3 Block diagram of the proposed image query system: a query image (**QI**) enters the hash database for possible retrieval of its original image from the image database

In order to represent the condensed essence of an image for perceptual similarity measurement, a hash function is usually employed. Conventionally, a cryptographic hash function, H^{c}, is used to map an image **I** as a short binary string, $H^{\mathrm{c}}(\mathbf{I})$. One of the most important properties of cryptographic hashing is that it is collision-free, which means that it is hard to find two different images that can be transformed to produce the same hashes. Let $z \in \mathcal{Z}$, and let z and **I** be distinct. The collision-free property of cryptographic hashing will yield $H^{\mathrm{c}}(\mathbf{I}) \neq H^{\mathrm{c}}(z)$. Furthermore, let $x \in \mathcal{X}$; then, cryptographic hashing will yield $H^{\mathrm{c}}(\mathbf{I}) \neq H^{\mathrm{c}}(x)$. This implies that cryptographic hashing inherently produces totally different hash sequences if the media content has been modified.

However, this characteristic is too restricted to be suitable for multimedia applications since multimedia content permits acceptable distortions. As a result, it is necessary to develop a media hashing function, H^{m}, that can provide error-resilience. The error-resilience property of media hashing is defined as follows. It is said that x $(\in \mathcal{X})$ is successfully identified as having been modified from **I** if $d(H^{\mathrm{m}}(\mathbf{I}), H^{\mathrm{m}}(x)) \leq \varepsilon$ holds, where $d(\cdot,\cdot)$ indicates a Hamming distance function. In other words, if two images are perceptually similar, their corresponding hashes must be highly correlated. In addition, the desired media hash function still needs to possess the collision-free property, like cryptographic hashing, except that the distance measure is changed to $d(H^{\mathrm{m}}(\mathbf{I}), H^{\mathrm{m}}(x)) > \varepsilon$. On the other hand, it is insignificant whether y $(\in \mathcal{Y})$ can be identified as having been modified from **I** or not because y is severely degraded from **I** and they are perceptually dissimilar in terms of similarity measurement. It should be noted that the traditional cryptographic hash function is a special case of the media hash function in that its ε value is set to 0. Overall, the main idea behind media hashing is to develop a robust hash function that can identify perceptually similar media contents and possess the collision-free property. Issues related to media hashing will be discussed in Sect. 4.

3 Proposed image hashing approach

The block diagrams of the proposed mesh-based image hashing system and image query system are depicted in Figs. 2 and 3, respectively. In our method, the mesh generation algorithm is executed in the lower-frequency component of an image in the wavelet domain so that more robust salient points can be detected (Sect. 3.1). Then, a mesh normalization process (Sect. 3.2) is used to transfer the decomposed meshes as normalized meshes of fixed sizes. Since the conventional image transformation methods (DCT or wavelet) cannot be directly applied to the triangle mesh to extract its feature, each triangle mesh is made square before hash extraction. Finally, a mesh-based hash extraction algorithm (Sect. 3.3) can be used to generate hash sequences of equal length to facilitate hash matching (Sect. 3.4). Basically, hash extraction can be conducted in either the DCT [9] or wavelet [10] domain. Instead of the wavelet-based hash extraction method, in this paper, the DCT-based hash extraction method is adopted because a shorter binary hash sequence is easy to create. The major components of our proposed image hashing system will be sequentially described in this section.

3.1 Robust image mesh generation

The extraction of robust meshes plays an important role in our method since it is a prerequisite for withstanding

geometric distortions. To generate meshes, the first step is to detect the salient points of an image. Among the ubiquitous feature point extraction methods, the Harris detector has been widely used. However, the original Harris detector is not yet robust enough to be used for our purposes since we need to deal with query images that may have been manipulated. If the Harris detector is directly applied in the spatial domain, the noise component of an image will affect the detection of salient points. Thus, we propose to improve its robustness by applying it in the lowest-frequency subband of the downsampled-discrete wavelet transform (DS-DWT) domain. Our intention is twofold: (i) to avoid detection of salient points in the high-frequency subbands that are inherently contaminated with noises; and (ii) to reduce the space for mesh and hash generations. Empirical results obtained from hundreds of images (manipulated by means of Stirmark3.1 and Stirmark4.0 [19, 20]) verify the robustness of salient point detection.

Once the feature point extraction process is finished, the Delaunay tessellation is exploited to decompose the image into a set of disjointed triangles. Each triangle (called a mesh in this paper) is regarded as the minimum unit for robust hash extraction. The overall mesh generation process is summarized as follows: (i) the original image $\mathbf{I}$ is transformed via downsampled discrete wavelet decomposition, and the lowest-frequency subband signal, $\mathbf{I}_{\mathrm{LL}}$, is selected; (ii) the set of feature points $\mathcal{P}$ is generated by applying the Harris detector to $\mathbf{I}_{\mathrm{LL}}$; and (iii) Delaunay tessellation is performed using $\mathcal{P}$ to obtain a set, $\mathcal{M}$, of meshes ($|\mathcal{M}|$ is used to denote the number of meshes in $\mathcal{M}$).

Although a few differences between the meshes generated from the original image and its modified versions may exist, these differences are not certain to affect the hash-based similarity measurement, as will be explained in Sect. 3.4.

3.2 Mesh normalization

Once the set of meshes in an image has been produced, each original mesh $\mathbf{M}_k (\in \mathcal{M})$ is normalized as $\mathbf{M}_k^{nom}$ to generate a mesh-based hash $\mathbf{H}_k$, where $\mathbf{M}_k^{nom}$ is a right-angled triangle. The aim of normalization is to keep all normalized meshes the same size and the extracted mesh-based hashes the same length to make mesh-based hash comparisons possible. Figure 4 illustrates the relationship between $\mathbf{M}_k$ and $\mathbf{M}_k^{nom}$, where $\langle A, B, C\rangle$ and $\langle \acute{A}, \acute{B}, \acute{C}\rangle$ denote the corners of $\mathbf{M}_k$ and $\mathbf{M}_k^{nom}$, respectively. In addition, let $\langle A, B, C\rangle$ be arranged to satisfy $\angle BAC \geq \angle ABC \geq \angle ACB$, where $\angle BAC$ denotes an angle centered at corner A, and let $\langle \acute{A}, \acute{B}, \acute{C}\rangle$ be arranged to satisfy $\angle \acute{B}\acute{A}\acute{C} > \angle \acute{A}\acute{B}\acute{C} \geq \angle \acute{A}\acute{C}\acute{B}$, where $|\bar{\acute{A}}\bar{\acute{B}}| \leq |\bar{\acute{A}}\bar{\acute{C}}|$. When the normalization process is performed, $\langle A, B, C\rangle$ is first mapped to $\langle \acute{A}, \acute{B}, \acute{C}\rangle$ sequentially. That is, this "angle order" must be maintained to keep uniform warping in order to not affect the generation of normalized meshes and their corresponding hashes. Here, non-uniform warping implies that an original mesh and its attacked mesh are normalized in a different

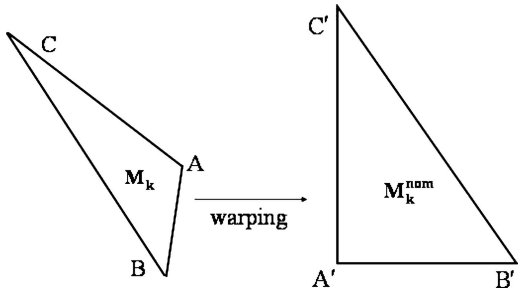


Fig. 4 The angle ordering between an original mesh and its normalized mesh. The angle ordering is determined by sorting the three angles in descending order. The length of the two shorter sides in the right-angled triangle satisfies $|\bar{\acute{A}}\bar{\acute{B}}| \leq |\bar{\acute{A}}\bar{\acute{C}}|$

angle order; thus, the resultant normalized meshes will produce different hashes. After angle order-based corner mapping is performed, $\mathbf{M}_k$ is transformed into $\mathbf{M}_k^{nom}$ through the procedures of affine transformation and interpolation (Chap. 5 of [23] and Chap. 12 of [24]).

There are two major problems that will affect the angle order. One involves severe geometric distortions that change the order of three angles in a mesh. However, this factor will lead to apparent destruction of the visual quality of images, which will, thus, lose their commercial value. Therefore, we can ignore this problem. The other problem occurs when two or three angles are nearly the same in magnitude such that even a slight distortion can change their order. This problem must be dealt with since the visual quality of an image is only imperceptibly modified. Our solution is to generate two (if two angles are nearly the same) or six (if three angles are nearly the same) different hashes for such a mesh by changing the angle order sequentially with respect to a query image. In addition, we keep one mesh in order to have one corresponding hash in the hash database to save space.

In the implementation, the length of both shorter sides, $\bar{\acute{A}}\bar{\acute{B}} = \bar{\acute{A}}\bar{\acute{C}}$, in a normalized mesh that is a right-angled triangle needs to be chosen carefully based on the following considerations. First, if the side is short enough, partial information of an original mesh will be lost, and not enough discriminable features will be provided among different meshes (i.e., does not obey collision-free). Second, if the side is long enough, the execution time will be mostly spent on mesh warping, which will become a bottleneck in the scheme. Finally, since the extracted feature points may have deviated from their original positions, larger normalized meshes will enlarge this impact to degrade the robustness of hash generation. In addition to the above considerations, we have performed extensive experiments (the same as the one that will be described in Sect. 5.1) based on using different sizes of normalized meshes and obtained the following results: (i) if $|\bar{\acute{A}}\bar{\acute{B}}| = |\bar{\acute{A}}\bar{\acute{C}}| = 64$ pixels is adopted (the hash's length is 256), the robustness is significantly degraded; (ii) if $|\bar{\acute{A}}\bar{\acute{B}}| = |\bar{\acute{A}}\bar{\acute{C}}| = 48$ pixels (the hash's length is 144) and $|\bar{\acute{A}}\bar{\acute{B}}| = |\bar{\acute{A}}\bar{\acute{C}}| = 32$ pixels (the hash's length is 64) are, respectively, adopted, their robustness capabilities are comparable. By taking the above three considerations

into account, the length of both sides is empirically verified to be $|\bar{\acute{A}}\acute{B}| = |\bar{\acute{A}}\acute{C}| = 32$ pixels so as to obtain a trade-off among discrimination and robustness of the hash, and the complexity of normalization. This choice will become clear in Sect. 3.3 that the size of a mesh-based hash sequence is linearly proportional to the size of a normalized mesh.

3.3 Robust mesh-based hashing

The goal of image hashing is to transform image content into a feature sequence in order to obtain a condensed representation. This feature sequence must be short enough for fast matching and preserve distinguishable features for similarity measurement to be feasible. In addition, a robust hashing algorithm is necessary to accommodate possible changes of normalization results. With the above crucial factors taken into consideration, in this paper, the robust hash of each normalized mesh $\mathbf{M}_k^{nom}$ is extracted in the block-DCT domain (our experience showed that it is not easy to obtain a shorter hash in the wavelet domain [10]). The extracted hash bits are position-dependent and belong to a certain type of local feature.

First, each triangle $\mathbf{M}_k^{nom}$ is flipped and padded with its flipped version to form a 32×32 block. Each 32×32 block is divided into sixty four 4×4 blocks, as illustrated in Fig. 5a. Second, the 4×4 DCT transform is performed, and the first AC coefficient (located at the lowest frequency subband except for the DC term) of each 4×4 block is selected. All the

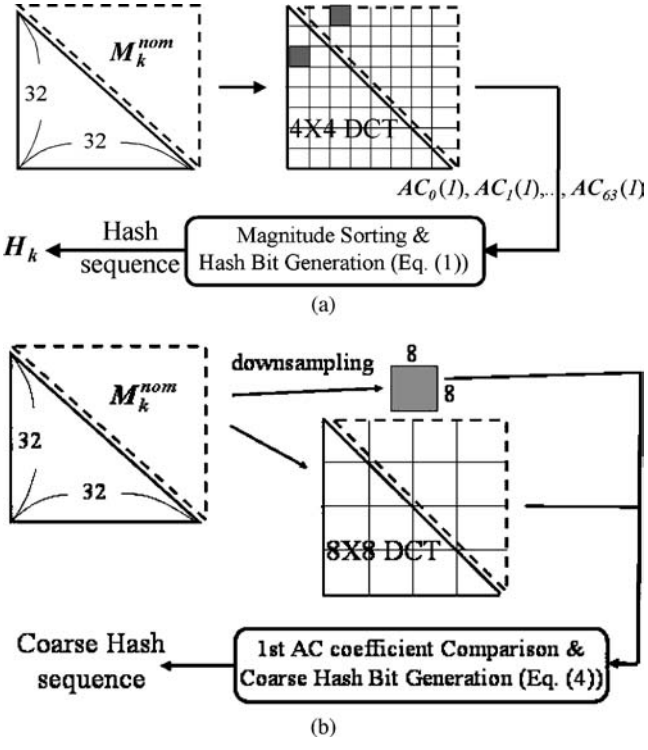


Fig. 5 Mesh padding for hash extraction in the block-DCT domain: **a** full hash generation and **b** coarse hash generation

selected AC coefficients form an AC sequence with a length of 64. It should be noted that due to the effect of padding, the upper triangular region and the lower triangular region capture different features. For example, as shown in Fig. 5a, if the dark-gray block in the lower triangular region captures the horizontal feature, then its padding counterpart will capture the vertical feature. The DC coefficients are not selected because they are not helpful for capturing identifiable features. Finally, this AC sequence is sorted according to the magnitudes of its 64 elements, and the hash bits, $H_k(s)$'s, are assigned as follows:

$$H_k(s) = \begin{cases} 1, & \text{if } |AC_k^s(1)| \text{ belongs to the first 32} \\ & \text{largest AC coefficients} \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where $H_k(\cdot)$ is a hash bit in a binary hash sequence $\mathbf{H_k}$ and $AC_k^s(1)$ $(0 \le s \le 63)$ denotes the first AC coefficient in a 4×4 block s of a normalized mesh $\mathbf{M}_k^{nom}$.

It is worth mentioning that the hash bits determined by Eq. (1) are image position-dependent (i.e., s). Unlike other hashing methods that have adopted global or statistical features, this additional security measure should be used to avoid the collision problem, where two dissimilar images have similar hashes. We shall further discuss this problem in Sect. 4.1.

In Eq. (1), one hash bit is generated from a 4×4 block. In addition, the mesh-based hash designed according to Eq. (1) guarantees that the number of 1's and 0's is the same, i.e., uniform distribution is achieved, in order to avoid any bias that will affect hash matching. This uniform distribution of hash bits is necessary to ensure that the proposed image hashing function will be collision-free and the false matching between different meshes can be avoided (this will be discussed in Sects. 4.1 and 4.2, respectively). We say that this feature value $H_k(\cdot)$ is robust because this magnitude relationship obtained after sorting can be approximately preserved. Please refer to Sect. 4.1 for detailed robustness analyses.

Note that the number of hashes is equal to the number of meshes in an image. $|\mathbf{H}_k|$ is used to denote the length of a binary hash sequence $\mathbf{H}_k$. In this paper, the hash dimensionality, $|\mathbf{H}_k|$, is fixed at 64, as explained previously. After mesh generation and mesh-based hash extraction are performed, the feature vector of an image $\mathbf{I}$ can be expressed as

$$\{\mathbf{H}^{\mathbf{I}}{}_1, \mathbf{H}^{\mathbf{I}}{}_2, \ldots, \mathbf{H}^{\mathbf{I}}_{|\mathcal{M}^{\mathbf{I}}|}\}, \quad (2)$$

where $|\mathcal{M}^{\mathbf{I}}|$ and $\mathbf{H}^{\mathbf{I}}{}_k$ denote the number of meshes and the k-th hash sequence in image $\mathbf{I}$, respectively.

3.4 Hash database creation and mesh-based fast matching

In this section, we will discuss how to create an image hash database with which the mesh-based matching process can be performed when an incoming query is received. Our hash database is designed to be suitable for two-stage fast search,

where the first stage produces potential candidates through a coarse matching process, and the second stage is a full matching process used to identify the final winner (if there is one) from the candidates. The overall image query system is depicted in Fig. 3.

3.4.1 Similarity measurement

Since the objective of this paper is to provide a high degree of robustness against various attacks (including geometric distortions), partial matching is considered when similarity is measured. In content copy detection and tracing applications, two images ($\mathbf{I}_m$ and $\mathbf{I}_n$) are considered to be similar if at least N mesh-pairs are matched. Moreover, it is said that a pair of meshes is matched if the bit error rate (BER) between their corresponding hashes is smaller than a threshold T ($0 \leq T \leq 1$), i.e.,

$$\mathrm{BER}(\mathbf{H_i^{I_m}}, \mathbf{H_j^{I_n}}) = \frac{\sharp\{t | H_i^{I_m}(t) \neq H_j^{I_n}(t)\}}{|\mathbf{H_i^{I_m}}|} \leq T, \tag{3}$$

where $H_i^{\mathbf{I_m}}(t)$ denotes the t–th element of the i–th hash in $\mathbf{I}_m$ and $\sharp\{\}$ denotes the number of bit errors. The two thresholds, T and N, will be derived based on a sufficiently small false positive probability in Sect. 4.2.

3.4.2 Full matching

Let $|\mathcal{M}_m^{\mathbf{I}}|$ denote the number of meshes in an image $\mathbf{I}_m$. Conventionally, the image hash database collects and stores the hashes of all images. Thus, $|\mathcal{M}_m^{\mathbf{I}}| \times |\mathcal{M}_n^{\mathbf{I}}|$ mesh pairs have to be compared in order to determine whether two images, $\mathbf{I}_m$ and $\mathbf{I}_n$, are similar or not according to the similarity measurement described in Sect. 3.4.1. However, we cannot rely on the exhaustive matching process only. This is because when the database is huge, the amount of time consumed by exhaustive hash matching becomes tremendous. Thus, this kind of search is not suitable for many applications that require real-time processing. We call this kind of matching process "full matching." Therefore, full matching is only performed on those candidates that have been retrieved through a rapid coarse matching process (this will be described in Sect. 3.4.4). A coarse-to-fine image hash database with the error-robustness capability for fast search will be described in the next section.

3.4.3 Creation of error-resilient tree-structured image hash database for fast search

In order to speed-up the matching process, we propose a fast matching technique that has two stages: (i) "coarse matching" for rapid selection of candidates; (ii) "full matching" for determining the final target from the selected candidates. In fact, this technique looks like a coarse-to-fine searching paradigm. The so-called coarse matching stage is mainly used to coarsely find a set of candidates (whose size is usually significantly smaller than the entire search space) that

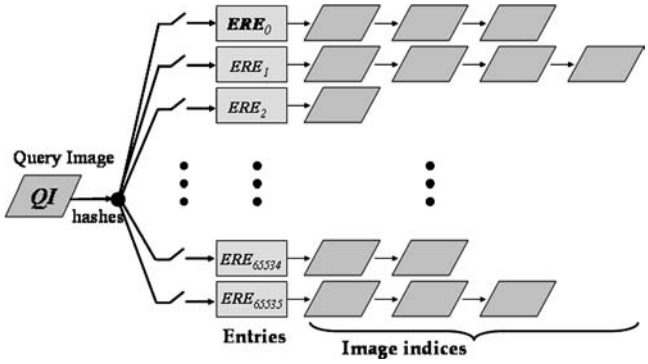


Fig. 6 Creation of an image hash database for fast search in a coarse-to-fine manner. Our image hash database includes (i) error-resilient entries; (ii) image indices; and (iii) image hashes

may contain the desired target. Next, full matching is conducted on the set of candidates to exhaustively find the final result. Therefore, our fast matching paradigm needs to be used with a hash database that is designed in a sophisticated manner. This sophisticated hash database is built as shown in Fig. 6. The hash database consists of "entries," each of which links to a chain that contains the indices of images. It is said that an image can be linked to a specific entry if one hash of that image and the entry are similar. In practice, each entry is the seed of a group. It is also observed that a group associated with an entry can proliferate rapidly if this entry is a common feature among images. As a result, our approach to building the hash database for fast searching is a kind of clustering method. However, unlike other clustering methods that have been proposed for content-based image retrieval, our clustering paradigm adopts partial clustering instead of global clustering. That is to say, an image can be linked to different entries as long as its hashes are similar to more than one entry.

There are two issues that should be considered when constructing a hash database; i.e., entries should (i) be short enough for practical implementation and (ii) possess the error-robustness capability in order to accommodate modifications of meshes due to attacks. By taking the above two issues into account simultaneously, in this paper, we design each entry as a 16-bit long hash sequence that also represents a coarse representation of a mesh. This "coarse hash," which is generated in a way similar to the one described in Sect. 3.3, is described as follows. As shown in Fig. 5b, each normalized mesh of size 32×32 is downsampled to obtain an 8×8 coarse block from which 8×8 DCT is performed and the first AC coefficient, denoted as B_{ds}, is selected. Meanwhile, each normalized mesh is also partitioned into sixteen 8×8 blocks with which 8×8 DCT is performed and 16 1st AC coefficients, denoted as $B_{\mathrm{p}}(b)$ ($0 \leq b \leq 15$), are obtained. With the above setting, the coarse hash bits are the results obtained by comparing each $B_{\mathrm{p}}(b)$ with B_{ds}. More specifically, the coarse hash bit of a mesh is defined as

$$CH(b) = \begin{cases} 1, & \text{if } B_{\mathrm{p}}(b) \geq B_{\mathrm{ds}} \\ 0, & \text{otherwise;} \end{cases} \tag{4}$$

where $CH(b)$ is a hash bit in a coarse hash sequence **CH**. As indicated in Eq. (4), each coarse hash is a 16-bit vector, which implies that each entry is also composed of 16 bits, and that there are a total of 65536 entries. The entries are expressed as $ERE_0, ERE_1, \ldots, ERE_{65535}$, where ERE_i is a binary representation of i. In our implementation, the size of the ERE_i's needs to be controllable so that they can be stored in an array for rapid indexing. This is related to the first issue. As for error resilience, this means that even if an image has been modified, its coarse hashes are largely unaffected. Since the proposed coarse block has the low-frequency characteristic and the coarse hash bits are designed based on the magnitude relationship between two blocks, both are stable and hard to change. Readers can refer to [10] for robustness analyses. Consequently, coarse matching is able to reliably select candidates that contain the desired target.

The clustering operation associated with each entry is performed as follows. It is said that an image's index id is linked to an entry ERE_i if at least one coarse hash **CH** of the image $\mathbf{I}_{id}$ and ERE_i are the same, i.e.,

$$BER(\mathbf{CH}, ERE_i) = 0. \tag{5}$$

Through the above process, the image hash database can be built in an off-line manner. Basically, the built image hash database is error-resilient and tree-structured, and it permits newcomers to join at any time.

The relationship between the image database and hash database will be discussed in Sect. 4 with respect to the complexity issue. Based on the proposed error-resilient coarse-to-fine image hash database, the coarse matching part of the proposed fast matching process will be described in the next section.

3.4.4 Coarse matching

For an incoming query image, **QI**, each of its mesh-based hashes tries to enter the hash database through the entries. It is said that the j–th coarse hash of **QI**, $\mathbf{CH}_j^{\mathbf{QI}}$, is allowed to enter an entry E_i if $\mathbf{CH}_j^{\mathbf{QI}}$ and E_i satisfy Eq. (5). Since the idea behind our coarse matching method is to rapidly select candidates for advanced full matching, we first exploit the entries of the hash database to filter out those targets in the image database that are identified as being dissimilar to the incoming query. The goal is to reduce the number of images that are needed for full matching and, thus, to save time.

In our coarse matching process, if a coarse hash of an incoming query **QI** is allowed to enter an entry E_i, then the hit indicators of all the image indices that are linked to E_i will be increased by 1 to indicate the gradual increase of the possibility that the images are similar to the query. Let us denote by $\delta(id)$ the hit indicator of an image $\mathbf{I}_{\mathrm{id}}$. When all the coarse hashes of **QI** have gone through the above process, we retain those images (in the database) that have hit indicators larger enough as candidates for full matching in order to determine the final winner, i.e., the target with the best match. In fact, our empirical observations indicate that the desired target can be found from only a few candidates (e.g., smaller than 10). Compared with the millions of images in a database, this number of candidates greatly reduces the time required for searching. This also implies that most of the target images have been eliminated through coarse matching.

3.4.5 Valid or invalid retrieval

In the proposed two-stage matching paradigm, "valid retrieval" is defined as follows. Given a query image (**QI**), a hash database, and an image database, it is said that a target image is effectively retrieved to match **QI** if (i) candidates are retrieved that satisfy Eq. (5) during the coarse matching process (Sect. 3.4.4); (ii) the target image is the candidate, together with **QI**, that has N^v mesh pairs satisfying Eq. (3) (see Sect. 3.4.2) and $N^v \geq N$. Furthermore, the importance of valid retrievals is determined according to their N^v values. For example, the top n valid retrievals are the ones that have N^v values ranked among the top n of all the valid retrievals. In content searching and retrieval, the top 1 valid retrieval is regarded as the best match.

On the other hand, if all N^v's are smaller than N, then this search is considered invalid. As a result, it is concluded that the query image does not exist in the image database.

4 Analyses of media hashing issues

In this section, several challenging issues [25], including error analyses, robustness, granularity, complexity, and scalability of image hashing, will be discussed along with the parameters used in our implementation.

4.1 Robustness and discrimination

Robustness refers to the ability of image hashes to resist digital operations (including filtering, compression, geometric distortions, etc) such that the hashes generated before and after attacks are similar. Usually, these digital operations are "incidental" in that they might inevitably be applied for different purposes. Therefore, it is necessary for an image hashing scheme to resist incidental modifications. As indicated in Eq. (1), an image hash is generated as a binary sequence, in which the hash bits are determined according to the magnitudes of the AC coefficients in a normalized mesh. In this situation, keeping the magnitude relationship nearly unchanged (that is, the hash bits nearly unchanged) is necessary to achieve robustness. In the following, we will discuss how the magnitude relationship between AC coefficients can be approximately maintained when incidental operations are employed. It is said that a false negative or miss detection occurs if a query that is a modified version of an image in the image database cannot be correctly identified.

It was reported in [26] that the intrinsic content of an image can be sufficiently reconstructed from larger transformed coefficients only if they carry significant

information. In our previous work [10], we investigated the magnitude relationships between wavelet coefficients as the digital signature of an image for content authentication. These magnitude relationships were found to be mostly preserved under incidental manipulations and to be easily destroyed by malicious distortions. This implies that if a larger (or smaller) coefficient becomes smaller (or larger), then the content of an image will be changed as well. In particular, if more wavelet coefficients in the first image $\mathbf{I}_1$ are adjusted significantly, based on the same positions of the significant wavelet coefficients in the second image $\mathbf{I}_2$, then the content of $\mathbf{I}_1$ will gradually come to look like that of $\mathbf{I}_2$. (The reader may refer to [27] for more detailed analyses.) Our previous works [10, 27] revealed that it is necessary to keep larger (smaller) wavelet coefficients larger (smaller) in order to keep the perceptual content of an image relatively unchanged, and we can apply the same principle to the proposed normalized mesh-based hashing that is operated in the block-DCT domain. The major difference is that the inter-scale relationship is considered in the wavelet domain, while the intra-scale relationship is considered in the block-DCT domain to construct an image hash.

On the other hand, discrimination means that a pair of image hashes that are randomly obtained from two different sources should ideally be uncorrelated; i.e., the BER obtained using Eq. (3) should be approximately 0.5. Otherwise, any query is likely to be falsely recognized as existing in the image database. This false detection probability is also known as the false positive probability. Figure 7 shows the numerical distribution of BERs calculated from 1 million hash pairs that were randomly generated. The length of each hash is 64 to fit the proposed method. The mean and standard deviation of this distribution are 0.5 and 0.062, respectively. It can be found from these results that random hash pairs can statistically achieve an uncorrelated distribution.

In addition, our observations of real images are exactly consistent with the theoretical result shown in Fig. 7. One practical result is shown in Fig. 8 for comparison. The result shown in Fig. 8 was obtained by comparing a hash pair that was composed of a hash of Lenna and a hash of an image within a database (which contained all the hashes of 20,000 images, excluding Lenna). The mean and standard deviation of this distribution are, surprisingly, found to be 0.5 and 0.062, respectively. These results indicate that our hash extraction method achieves sufficient randomness.

When the best match criterion, as shown in Eq. (3), is taken into consideration, the mean of the BERs obtained from real hash pairs will be shifted. Figure 9 shows the result obtained by using each hash of Lenna as a query to find the best match (with the lowest BER) in a large hash database (which contained all the hashes of 20,000 images, excluding Lenna). Comparing Figs. 7–9, we can see that the mean of the BERs in Fig. 9 has shifted to a value lower than 0.5. The main reason for this result is that the best match that is produced from all matches with the lowest BER is considered.

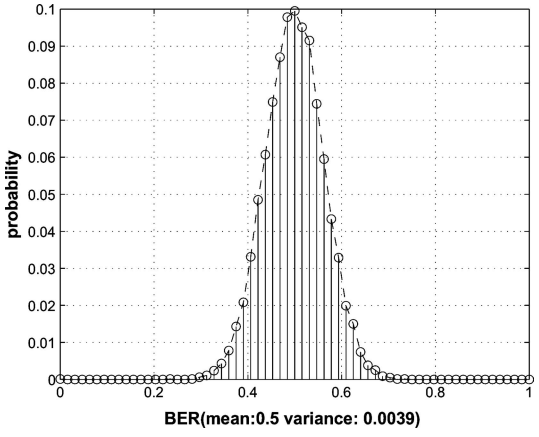


Fig. 7 Distribution of BERs calculated from random hash pairs (each hash has a length of 64). The horizontal axis indicates the bit error rate, while the vertical axis indicates the probability of occurrence of a bit error rate. In this numerical distribution, the mean and standard deviation are 0.5 and 0.062, respectively

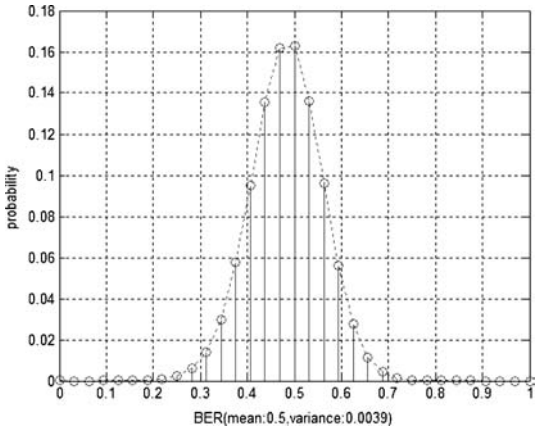


Fig. 8 Distribution of BERs calculated from hash pairs (each hash has a length of 64), where one hash is Lenna's hash, and the other hash is from the hash database. In this distribution, the mean and standard deviation are 0.5 and 0.062, respectively

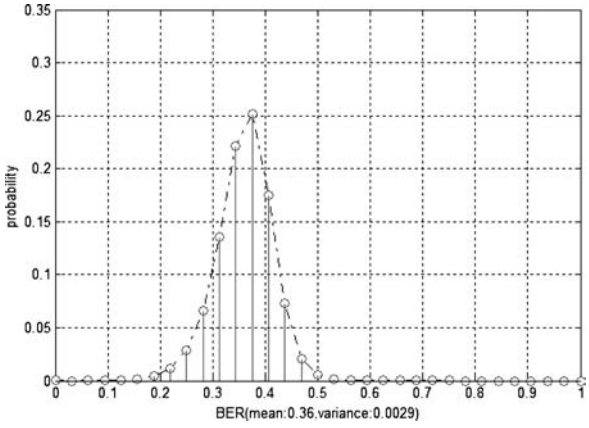


Fig. 9 Distribution of BERs calculated based on the best matched hash pairs (each hash has a length of 64), where one hash is Lenna's hash, and the other hash is from the hash database. In this distribution, the mean and standard deviation are 0.36 and 0.054, respectively

Finally, we also conducted experiments based on the best match criterion (Eq. (3)) when the hash database contains only hash sequences that are extracted from modified versions (by means of Stirmark) of a query image. In this case, each of the ten standard images ($\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_{10}$), which will be discussed in Sect. 5.1, was individually used as the query image, and its corresponding hash database was created. The mean and standard deviation of the distribution of BERs for each query image similar to that in Fig. 9 are summarized in Table 1. It is observed from Table 1 that the mean BERs are almost all smaller than 0.25.

As previously mentioned in Sect. 3.4, two thresholds, T and N, are relevant to both the false positive and false negative probabilities. Based on the practical results (shown in Fig. 9 and Table 1) we have obtained, it has been experimentally verified that T can be reasonably set to 0.25. The method used to determine the remaining threshold N will be discussed in the next section. Of course, T can also be varied to find different N's. This tedious task will be ignored in this paper.

4.2 Error analysis

In this section, we will discuss error analysis for threshold selection, including false positive and false negative probability analysis. Under a sufficiently small false positive and with $T = 0.25$ (note that T can also be used as a variable for the purpose of analysis), the parameter N that defines the number of matched mesh pairs (previously mentioned in Sect. 3.4) can be derived. Recall that the hash size of a normalized mesh is 64 bits. It is said that two random meshes (one from the query and the other from the database) are similar if their hash comparison satisfies Eq. (3); i.e., 16 bits at most are different. Let p_m^b be the probability of finding a pair of hashes that have $2b$ bit errors. It is expressed as

$$p_m^b = \frac{\left(C_b^{32}\right)^2}{\sum_{l=0}^{l=32}\left(C_l^{32}\right)^2}, \tag{6}$$

where C denotes a combinatorial function and $(C_l^{32})^2$ denotes the number of possible cases in which $2l$ bits in two compared hashes are found to be different. As shown in Eq. (1), an AC coefficient with a larger (or smaller) magnitude is assigned hash bit 1 (or 0). It is said that a bit error occurs if a larger (or smaller) AC coefficient becomes smaller (or larger). Under this circumstance, it is not hard to realize based on Eq. (1) that bit errors will appear in pairs, i.e., 2 bit errors, 4 bit errors, and so on. Similarly, the denominator of Eq. (6) stands for the total hash sequences that have equal numbers of 0's and 1's. Furthermore, let p_m denote the probability of finding a pair of hashes that satisfy a matching score (i.e., BER) that is equal to or smaller than T, which can be expressed as

$$p_m = p_m^0 + p_m^1 + \cdots + p_m^{\frac{64T}{2}} \approx 6.70 \times 10^{-5}. \tag{7}$$

Similarly, $(1 - p_m)$ denotes the probability of mismatch for a pair of hashes.

Let $\mathcal{H}_0$ and $\mathcal{H}_1$ be two hypotheses, where $\mathcal{H}_0$ specifies the queried hash does not exist in the hash database and $\mathcal{H}_1$ specifies the queried hash exists in the hash database. Based on Eq. (7) and a given value of N, the false positive probability given $\mathcal{H}_0$, $p_{fp}|\mathcal{H}_0$, is defined as

$$\begin{aligned} p_{fp}|\mathcal{H}_0 &= \sum_{n=N}^{|\mathcal{M}|} C_n^{|\mathcal{M}|}(1-p_m)^{|\mathcal{M}|-n} p_m^n \\ &\geq C_N^{|\mathcal{M}|}(1-p_m)^{|\mathcal{M}|-N} p_m^N \\ &\approx C_N^{|\mathcal{M}|} p_m^N, \end{aligned} \tag{8}$$

where $|\mathcal{M}|$ denotes the number of meshes in an image; $C_n^{|\mathcal{M}|}(1-p_m)^{|\mathcal{M}|-n} p_m^n$, with $n > N$, is sufficiently smaller than $C_N^{|\mathcal{M}|}(1-p_m)^{|\mathcal{M}|-N} p_m^N$; and $(1-p_m)^{|\mathcal{M}|-N}$ is approximately 1. Accordingly, $p_{fp}|\mathcal{H}_1$, denoting the probability of correct match, means true positive.

On the other hand, the probability of failing to retrieve the desired target given that $\mathcal{H}_1$ holds should be smaller than the false negative probability, $p_{fn}|\mathcal{H}_1$. However, it is hard to analyze $p_{fn}|\mathcal{H}_1$ since various discrepancies between the characteristics of digital image operations exist. Based on our observations, the false negative probability can increase if the query image is modified to a certain extent such that most of the detected meshes are different from the originals. Accordingly, $p_{fn}|\mathcal{H}_0$ will denote true negative.

Substituting some cases of $|\mathcal{M}|$ and N into Eq. (8) and solving p_{fp}, we find several relationships between $|\mathcal{M}|$, N, and sufficiently small p_{fp}. Based on these numerical results, we can conclude that $N = 3$ or 4 is feasible for obtaining a sufficiently small p_{fp}. Of course, we can use a larger N value to get an even lower false positive probability. However, a larger N value also implies that false negatives can easily occur.

4.3 Complexity

The complexity of the image hashing system that will be discussed here includes the time spent on mesh-based hash extraction and hash comparison, and the memory required to store the constructed hash database. They will be, respectively, discussed below.

As for the size of the proposed hash database, it is the sum of the total size of the entries, the total size of the image indices, and the total size of the image hashes, as illustrated in Fig. 6. Because each entry is a binary vector of 16 bits, the entry size is, in total, $2^{16} \times 16$ bits, i.e., 131,072 bytes. In addition, the total size of the image indices can be calculated as $\frac{|ID| \times \lceil \log_2 |ID| \rceil \times |\bar{\mathcal{M}}|}{8}$ bytes, where $|ID|$ denotes the number of images in the image database, $\lceil \log_2 |ID| \rceil$ denotes the number of bits used to represent an image's index ($\lceil \cdot \rceil$ denotes the ceiling operation), and $|\bar{\mathcal{M}}|$ is the average number of meshes in an image. Furthermore, the total size of the

Table 1 Distribution of BERs calculated based on the best matched hash pairs (each hash has a length of 64), where one hash is from the query image, and the other hash is from the hash database that is generated from the Stirmark attacked query image

BER's Statistics	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	$\mathbf{I}_4$	$\mathbf{I}_5$	$\mathbf{I}_6$	$\mathbf{I}_7$	$\mathbf{I}_8$	$\mathbf{I}_9$	$\mathbf{I}_{10}$
Mean	0.207	0.192	0.236	0.232	0.226	0.212	0.301	0.233	0.224	0.224
Standard deviation	0.128	0.124	0.133	0.133	0.131	0.133	0.120	0.117	0.136	0.136

image hashes can be calculated as $\frac{|\mathrm{ID}|\times|\bar{\mathcal{M}}|\times 64}{8}$ bytes. Therefore, the ratio σ between the size of a hash database and the size of its corresponding image database can be calculated as follows:

$$\sigma = \frac{131072 + \frac{|ID|\times\lceil\log_2|ID|\rceil\times|\bar{\mathcal{M}}|}{8} + \frac{|ID|\times|\bar{\mathcal{M}}|\times 64}{8}}{|ID|\times\bar{L}\times\bar{W}} \approx \frac{(\lceil\log_2|ID|\rceil + 64)\times|\bar{\mathcal{M}}|}{8\times\bar{L}\times\bar{W}}, \tag{9}$$

where $\bar{L}\times\bar{W}$ is used to denote the average size of various images. To clarify Eq. (9), let us take an image database of size 131,072, i.e., $|ID| = 2^{17}$, as an example. In addition, suppose $\bar{L}\times\bar{W} = 2^{16}$ and $|\bar{\mathcal{M}}| = 2^6$ approximately hold. Then, we find that $\sigma \approx 0.01$, which means that the size of the hash database when compared with the size of the image database can be limited within the order of 10^{-2}. Basically, the ratio, derived in Eq. (9), depends on the characteristic of an image database and is said to be application-dependent. However, Eq. (9) approximately indicates the relationship between the hash database and its corresponding image database.

As for mesh-based hash extraction, it is actually the most time-consuming step in our method because most of the time is spent on warping during the mesh normalization process. Basically, the number of arithmetic operations for pixel transformation during mesh normalization is constant and is proportional to the number of pixels in an original mesh. Since the total number of pixels in all meshes that are required to execute normalization is approximately equal to the size of an image, as a result, it can be concluded that the time complexity of mesh normalization is proportional to image's size.

As for hash comparison, this step has been previously described in Sect. 3.4. Since a sophisticated hash database is created for fast coarse matching followed by full matching, the time required for hash comparison, when compared with that required for full matching [28], is significantly reduced. This is because (i) the time-cost of full matching depends on the number of images in the image database; (ii) the time-cost of fast matching depends on the number of meshes in the query image and on the small but fixed number of candidates that should be used for full matching. In fact, the difference between (i) and (ii) lies on the number of comparisons between one hash sequence from the query image and the other one from the hash database. More specifically, the major difference is the size of the whole hash database, which is $|\mathrm{ID}|\times\bar{\mathcal{M}}$ averagely for condition (i), and that of the candidates determined in the coarse matching process, which is significantly smaller than $|\mathrm{ID}|\times\bar{\mathcal{M}}$, for condition (ii).

Overall, in the proposed scheme, the time required for fast matching is smaller than that required for hash extraction, which is further smaller than the time required for full matching.

4.4 Granularity

Granularity here means the minimum size of a query that can be identified in a feasible way in an image hashing system. As explained in Sect. 3.4.5, a valid retrieval is defined as one in which at least N hash pairs are matched; thus, the granularity is basically the size of a query that can accommodate at least N meshes. However, it is rather difficult to settle on a fixed value for this size since the mesh's size is image-dependent and may vary under different attacks.

4.5 Scalability

The scalability of an image hashing system as discussed here is the application scalability; i.e., the same hash database can be used for different purposes. In this paper, copy detection/tracing and content authentication applications will be investigated. Since we have described the proposed image hashing method for copy detection, in the following, we will explain how the proposed hash database can be adopted for image content authentication.

Imagine a scenario in which Alice would like to send an image to Bob through a network. During the transmission stage, in addition to the transmitted image, $\mathbf{I}$, Alice will also send the hashes of $\mathbf{I}$ to Bob, who can verify the authenticity of the received image, $\mathbf{I}^a$, according to the hashes, where $\mathbf{I}^a$ is a modified version of $\mathbf{I}$. Image content authentication [3, 9, 10] demands that hashes be robust against incidental modifications and sensitive to malicious distortions.

The mesh-based image authentication scheme proposed in this paper operates as follows. First, the mesh-based hashes are extracted from the received image, $\mathbf{I}^a$, and compared with the received hashes of $\mathbf{I}$ using Eq. (3). Then, those meshes whose corresponding hashes do not satisfy Eq. (3) will be marked as "incredible."

5 Experimental results

In this study, several experiments were conducted to evaluate the performance of the proposed mesh-based image hashing and query system. In Sects. 5.1 and 5.2, the performance of our copy detection scheme will be demonstrated, while in

Sect. 5.3, the performance of our image content authentication approach will be demonstrated.

5.1 Robustness: resistance to miscellaneous attacks

First, ten color images with different contents ($\mathbf{I}_1$: Pepper; $\mathbf{I}_2$: Lenna; $\mathbf{I}_3$: Bridge; $\mathbf{I}_4$: Sailboat; $\mathbf{I}_5$: Goldhill; $\mathbf{I}_6$: F16; $\mathbf{I}_7$: Baboon; $\mathbf{I}_8$: Clock; $\mathbf{I}_9$: Tank; and $\mathbf{I}_{10}$: Splash) were used to verify the robustness of our scheme against miscellaneous attacks. The standard benchmark, Stirmark versions 3.1 and 4.0, was quite suitable for simulating various manipulations of the digital images. The reader may refer to [19, 20] for more detailed parameters of Stirmark. In this test, the original image was used as a query to find out how many modified versions could be successfully detected. The results for robustness verification are summarized in Tables 2 and 3, respectively. In the two tables, each attack's name is followed by a digit in a parenthesis, which indicates the number of times that the attack was performed with different parameters. In addition, each field indicates the number of modified images that were successfully identified. Here, $N = 3$ and $T = 0.25$, as explained in Sects. 3.4 and 4.2, were adopted. According to Tables 2 and 3, among 1910 modified images, 1761 could be correctly identified, which indicates that the correct recognition rate was 92.2%.

From the above two tables, it can be observed that most of the modified images could be successfully detected. A few attacked images that failed to be identified are shown in Fig. 10 for visual inspection. We can observe from Fig. 10 that it was not possible to correctly extract meshes from attacked images involving markedly degraded fidelity and content elimination. In particular, severe cropping and heavy noise addition could break the connections among the meshes and thereby affect the hashes, thus defeating our system even though the attacked images might have lost their commercial value. However, compared with the existing methods[1] it is evident that our scheme indeed achieves promising resistance to extensive geometric distortions.

5.2 Identification: searching in a large database

The second group of experiments focused on the problem of searching in a large image database. In this searching system, the database was composed of so-called original color images (which consisted of the Corel image database, containing 20,000 images, and ten traditional images, such as Lenna, Baboon, and so on), while the query image was suspect in the sense that it could have been a modified version generated from our database or could have been totally irrelevant to the database. We used the attacked images, obtained from Stirmark 3.1 and 4.0, as queries of the database. Two measures, the recall rate and precision rate, were used to evaluate the searching performance. They are dependent on the parameters T, N, and n, and are, respectively, defined as follows:

$$\text{Recall}\,(T, N, n) = \frac{\text{No. of queries satisfying Eq.(3)}}{\text{No. of total queries}},$$

$$\text{Precision}\,(T, N, n) = \frac{\text{No. of queries satisfying Eq.(3)}}{\text{No. of detections satisfying Eq.(3)}},$$

where n means the number of valid retrievals that may include the desired target. In the above two equations, "No. of total queries" means the number of query images used to query the database, "No. of queries satisfying Eq. (3)" means the number of query images that can find a similar target from the database, and "No. of detections satisfying Eq. (3)" means the number of targets (in the database) that can match the query image. Basically, "No. of detections satisfying Eq. (3)" is an integer multiple (i.e., n) of "No. of total queries." Both the full matching and fast matching procedures were applied to searching, and their results were compared.

According to the full matching process (described in Sect. 3.4.2), a so-called "successful search" needs to be defined before the searching performance can be evaluated. Here, a successful search means that at least one of the valid retrievals (described in Sect. 3.4.5) contains the desired target. In this study, the top n valid retrievals were adopted to check whether at least one of them contained the desired target, where different values of n included 1, 2, 5, 10, and 100, respectively. In addition, we provide here information about the overall performance in terms of both the recall and precision rates, as depicted in Table 4[2]. As can be seen from Table 4, the overall searching results could improve significantly (up to ≈90%) only if the number of valid retrievals was less than 10. This outcome demonstrates that the proposed searching strategy is very efficient in finding the desired target without the need to check many valid retrievals. Note that the number of miss detections was slightly larger than that obtained in the robustness test (as explained in Sect. 5.1) because the search space had been broadened. Basically, these results reveal that the desired originals were hard to identify when the query images (e.g., Fig. 10) had been severely modified. Moreover, the queries that could not be found in the database were mostly consistent with those that could not be identified in the robustness test.

In order to speed up the search process in the case of a large database, the proposed fast matching process (as described in Sect. 3.4) was also employed. The time-cost

[1] We can see from Tables 4 and 5 of [18], Sect. 1 of [7], Table 1 of [17], Table 2 of [16], and the 2nd paragraph of Sect. 3 of [13] that their evaluations were limited to only a few geometric attacks. In Fig. 11 of [14], only four attacks were tested. We can see from [15] that only few rotation and scaling attacks were tested. As described in the 3rd paragraph of Sect. 5.1 of [6], only non-geometric attacks were tested.

[2] It should be noted that the traditional precision rate vs. recall rate measurement used in content retrieval may not be suitable for media hashing. This is because we are interested in identifying the correct target instead of just similar ones. As a result, increasing the number of detections will dramatically decrease the precision rate.

Table 2 Robustness of our scheme vs. Stirmark 3.1

Stirmark 3.1	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	$\mathbf{I}_4$	$\mathbf{I}_5$	$\mathbf{I}_6$	$\mathbf{I}_7$	$\mathbf{I}_8$	$\mathbf{I}_9$	$\mathbf{I}_{10}$
SPA (6)	6	6	6	6	6	6	6	6	6	6
JPEG (12)	12	12	12	12	12	12	12	12	12	12
GLGT (3)	3	3	3	3	3	3	3	3	3	3
CAR (8)	8	8	8	8	8	8	7	8	8	8
LR (5)	5	5	5	5	5	5	5	5	5	5
Flipping (1)	1	1	1	1	1	1	1	1	1	1
Cropping (9)	8	7	7	8	8	8	4	8	3	6
RC (16)	15	15	15	15	14	15	12	15	14	15
Scaling (6)	6	6	4	6	6	6	2	6	4	4
RRS (16)	15	15	15	16	15	16	10	16	12	14
Shearing (6)	6	6	6	6	6	6	6	6	6	6
RB (1)	1	1	1	1	1	1	1	1	1	1

Attacks are denoted as SPA: the signal processing attack, including median filtering, Gaussian filtering, sharpening, and frequency mode Laplacian removal (FMLR); JPEG: compression with quality factors ranging from 90% to 10%; GLGT: general linear geometric transform; CAR: change of the aspect ratio; LR: line removal; RC: rotation + cropping; Scaling: scaled with factors ranging from 0.5 to 2.0; RRS: rotation + re-scaling; RB: random bending.

Table 3 Robustness of our scheme vs. Stirmark 4.0

Stirmark 4.0	$\mathbf{I}_1$	$\mathbf{I}_2$	$\mathbf{I}_3$	$\mathbf{I}_4$	$\mathbf{I}_5$	$\mathbf{I}_6$	$\mathbf{I}_7$	$\mathbf{I}_8$	$\mathbf{I}_9$	$\mathbf{I}_{10}$
AffineT (8)	8	8	8	8	8	8	8	8	8	8
ConvF (2)	2	2	2	2	2	2	1	2	1	1
Cropping (4)	2	1	1	1	2	2	0	1	1	2
JPEG (12)	12	12	12	12	12	12	12	12	12	12
MF (4)	4	4	4	4	4	4	4	4	4	4
Noise (6)	1	1	1	2	1	2	1	1	1	1
SS (3)	3	3	3	3	3	3	3	3	3	3
Scaling (6)	5	6	4	6	5	6	4	6	4	6
RML (10)	10	10	10	10	10	10	10	10	10	10
PSNR (11)	11	11	11	11	11	11	11	11	11	11
Rotation (16)	16	16	16	16	16	16	16	16	16	16
RRS (10)	10	10	10	10	10	10	10	10	10	10
RC (10)	10	10	10	10	10	10	10	10	10	10

Attacks are denoted as AffineT: affine transformation; ConvF: convolution filtering; Cropping: cropped to $\frac{3}{4}$, $\frac{1}{2}$, $\frac{1}{4}$, and $\frac{1}{5}$ of the original size; JPEG: compression with quality factors ranging from 90% to 10%; MF: median filtering; Noise: noise addition; SS: self-similarities; Scaling: scaled with factors ranging from 0.5 to 2.0; RML: removing lines; PSNR: all pixel values increased by the same quantity; Rotation: pure rotation; RRS: rotation + re-scaling; and RC: rotation + cropping.

Table 4 Recall rate vs. precision rate for full searching with Stirmark

	Query searching									
	Stirmark 3.1 (890 queries)					*Stirmark 4.0 (*1020 *queries)*				
Top n matches	1	2	5	10	100	1	2	5	10	100
Recall rate (%)	82.1	86.5	90.7	93.3	94.5	84.4	87.1	89.4	90.4	91.2
Precision rate (%)	82.1	43.3	18.1	9.3	0.9	84.4	43.5	17.9	9.0	0.9

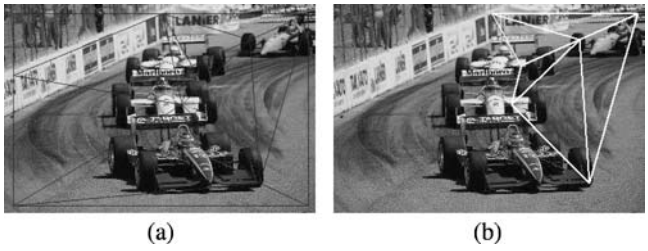
Fig. 10 Examples of failures in the robustness test. **a** and **c** were caused by Stirmark 4.0, and **b** and **d** were caused by Stirmark 3.1

(a) (b)

Fig. 11 Original images for content authentication. The detected meshes are imposed on the images

(a) (b)

Fig. 12 Content authentication under malicious tampering: **a** the image modified through face substitution in the middle part of Fig. 11 **a**; **b** the areas that were detected as having been maliciously tampered with are indicated by meshes with white lines



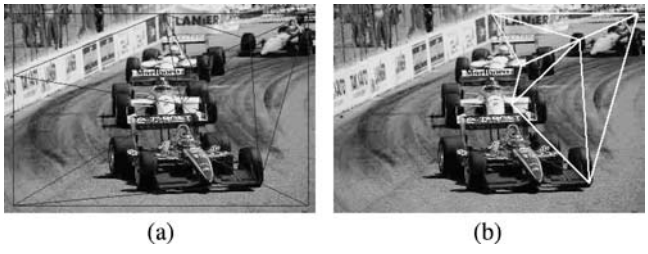
(a) (b)

Fig. 13 Content authentication under malicious tampering: **a** the image modified through object substitution applied to the top-right part of Fig. 11**b**; **b** the areas that were detected as having been maliciously tampered with are indicated by meshes with white lines

could be greatly reduced because entry entrance checking offers early elimination of those images that are dissimilar to the query. The overall performance of fast searching in terms of both the recall and precision rates is depicted in Table 5. Comparing Tables 4 and 5, we find that the proposed fast searching strategy is efficient and achieves performance comparable to that of full searching. In addition, it can be found that the precision rates drop significantly when the top n matches become large. This is because the target we would like to search is the "original" corresponding to the query image, and the number of originals for a query (if the query is modified from the one in the database) is only one.

(a) (b)

Fig. 14 Content authentication under malicious tampering+incidenal modification (affine transformation): **a** the modified image and its detected meshes; **b** the areas that were detected as having been maliciously tampered with are indicated by meshes with white lines



(a) (b)

Fig. 15 Content authentication under malicious tampering+incidenal modification(JPEG with a quality factor of 30%): **a** the modified image and its detected meshes; **b** the areas that were detected as having been maliciously tampered with are indicated by meshes with white lines

5.3 Content authentication

In this section, image content authentication performed using the proposed hash database will be analyzed. If the attacker only manipulates the image by means of incidental modifications, the authenticated results will be the same as the results obtained in the robustness test, as explained in Sect. 5.1. In addition, we will further consider two scenarios involving actions that the attacker may perform: (i) malicious modification; and (ii) malicious modification + incidental modification. Some results with respect to the above two scenarios are shown in Figs. 11–15, respectively.

Figure 11 shows two original images that were transmitted from the sender Alice to the receiver Bob. During transmission, the original images were maliciously tampered with by pasting in an additional object to form modified images, as shown in Figs. 12a and 13a. Since the newly added objects destroyed the original mesh structures and their corresponding hashes, the areas (containing a number of meshes) that contained the pasted object were located as shown in Figs. 12b and 13b. In addition, the images shown in Figs. 14a and 15a were further manipulated using some incidental operations. The resultant attacked images are shown in Figs. 14a and 15a, respectively. After our authentication scheme was performed, the areas that were detected as having been maliciously tampered with were those shown in Figs. 14b and 15b, respectively.

Table 5 Recall rate vs. precision rate for fast searching with Stirmark

	Fast searching									
	Stirmark 3.1 (890 queries)					*Stirmark 4.0 (*1020 *queries)*				
Top n matches	1	2	5	10	100	1	2	5	10	100
Recall rate (%)	80.5	84.5	87.2	87.2	87.2	83.6	85.1	86.0	86.2	86.2
Precision rate (%)	80.5	42.5	17.5	8.7	0.9	83.6	42.5	17.2	8.6	0.9

It can be observed from the above authentication results that the areas that were detected as having been maliciously tampered were composed of two kinds of meshes. The first kind was meshes newly generated from pasted objects, while the second kind was meshes generated due to some salient points that changed due to incidental manipulations. However, our method in its current form could not distinguish between the two kinds of meshes. A more robust salient point extraction technique is required.

6 Conclusions

A robust mesh-based image-hashing scheme for content management of digital images has been proposed in this paper. Our scheme is mainly composed of two components: (i) mesh-based robust hash generation and (ii) hash database construction for error-resilient and fast searching. In comparison with the existing methods, the main contribution of our approach is that it significantly improves the resistance of image hashing to geometric distortions. Furthermore, we have investigated several media hashing issues, including robustness and discrimination, error analysis, complexity, granularity, and scalability. We have also demonstrated application of the robust mesh-based image hashing system to both copy detection and content authentication.

However, our scheme is somewhat complex because most of the time is spent on mesh normalization. Fortunately, the hash database used for querying and searching can be built in an off-line manner. As a result, time is mainly spent on mesh-based hash generation of an incoming query image. However, our scheme compensates for this cost by offering robustness against geometric distortions. A fast matching process has also been proposed to speed up searching in a large image database. To understand the impact of different parameters on the false alarm rate, error analyses were conducted to derive guidelines for determining the necessary parameters.

Some directions for further research have been identified as follows. First, we will study the challenging problem of achieving more robust feature point extraction for mesh generation. This problem is particularly crucial for both identification of small images and mesh-based image authentication. Second, we will extend the scope of our method to searching and identifying images in URLs. Finally, the security of media hashing that may be application-dependent is worth studying.

Acknowledgements This research was supported, in part, by the National Science Council under NSC grant 92-2422-H-001-004. The authors thank Mr. Sun for providing the feature point extraction algorithm.

References

1. Oostveen, J., Lu, C.S., Sun, Q.: IEEE International Conference on Multimedia and Expo: Special Session on Media Identification, (co-organizers) (2004)
2. IEEE International Workshop on Multimedia Signal Processing (MMSP) Special Session on Media Recognition, Virgin Islands, USA (2002)
3. Dittmann, J., Steinmetz, A., Steinmetz, R.: Content-based digital signature for motion pictures authentication and content-fragile watermarking. In: Proceedings of the IEEE International Conference on Multimedia Computing and Systems, vol. II, pp. 209–213. Italy (1999)
4. Lin, C.Y., Chang, S.F.: Generating robust digital signature for image/video authentication. In: Proceedings of the ACM Multimedia and Security Workshop. U.K. (1998)
5. Chang, E.Y., Wang, J.Z., Li, C., Wiederhold, G.: RIME: A replicated image detector for the world wide web. In: Proceedings of the SPIE: Multimedia Storage and Archiving Systems, vol. III (1998)
6. Chang, E.Y., Li, C., Wang, J.Z., Mork, P., Wiederhold, G.: Searching Near-Replicas of Images via Clustering. In: Proceedings of the SPIE Symposium of Voice, Video, and Data Communications, pp. 281–292. Boston (1999)
7. Meng, Y., Chang, E.: Image copy detection using dynamic partial function. In: Proc. SPIE Storage Retrieval Media Database, vol. 5021, pp. 176–186 (2003)
8. Li, B., Chang, E., Wu, C.T.: DPF—A perceptual distance function for image retrieval. In: Proceedings of the IEEE International Conference on Image Processing, vol. II, pp. 597–600. Rochester (2002)
9. Lin, C.Y., Chang, S.F.: A robust image authentication method distinguishing JPEG compression from malicious manipulation. IEEE Trans. Circuits Systems Video Tech. **11**(2), 153–168 (2001)
10. Lu, C.S., Mark Liao, H.Y.: Structural digital signature for image authentication: An incidental distortion resistant scheme. IEEE Trans. Multimedia **5**(2), 161–173 (2003)
11. Fridrich, J.: Visual hash for oblivious watermarking. In: Proceedings SPIE: Security and Watermarking of Multimedia Contents II, pp. 286–294. San Jose, California (2000)
12. Fridrich, J., Goljan, M.: Robust hash functions for digital watermarking. In: Proceedings IEEE International Conference on Information Technology: Coding and Computing, pp. 178–183. Las Vegas, Nevada (2000)
13. Venkatesan, R., Koon, S.M., Jakubowski, M.H., Moulin, P.: Robust Image Hashing. In: Proceedings of the IEEE International Conference on Image Processing, vol. III, pp. 664–666. Vancouver (BC), CA (2000)
14. Lefebvre, F., Macq, B.: RASH: RAdon Soft Hash algorithm. In: Proceedings European Signal Processing Conference. Toulouse, France (2002)

15. Lefebvre, F., Czyz, J., Macq, B.: A robust soft hash algorithm for digital image signature. In: Proceedings of the IEEE International Conference on Image Processing. vol. II, pp. 495–498. Barcelona, Spain (2003)
16. Seo, J.S., Haitsma, J., Kalker, T., Yoo, C.D.: A robust image fingerprinting system using the radon transform. Signal Process.: Image Commun. **19**, 325–339 (2004)
17. Mihcak, M.K., Venkatesan, R.: New iterative geometric methods for robust perceptual image hashing. In: Proceedings of the ACM Workshop on Security and Privacy in Digital Rights Management. Philadelphia, PA (2001)
18. Kim, C.: Content-based image copy detection. Signal Process.: Image Commun. **18**, 169–184 (2003)
19. Petitcolas, F., Anderson, R.J., Kuhn, M.G.: Attacks on copyright marking systems. Proceedings of the International Workshop on Information Hiding, LNCS 1575, pp. 219–239 (1998)
20. Petitcolas, F.: Watermarking Schemes Evaluation. IEEE Signal Processing Magazine **17**(5), 58–64 (2000)
21. Bas, P., Chassery, J.M., Macq, B.: Geometrically invariant watermarking using feature points. IEEE Trans. Image Process. **11**, 1014–1028 (2002)
22. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
23. Al-Mualla, M.E., Canagarajah, C.N., Bull, D.R.: Video Coding for Mobile Communications, Academic Press (2003)
24. Heiny, L.: Advanced Graphics Programming Using C/C++. John Wiley & Sons, Inc. (1993)
25. Kalker, T.: Applications and challenges for audio fingerprinting. In: Proceedings of the 111th AES Convention, in the "Watermarking vs. Fingerprinting" Workshop, December 3 (2001)
26. Mallat, S., Zhong, S.: Characterization of signals from multiscale edges. IEEE Trans. Pattern Anal. Machine Intell. **14**(7), 710–732 (1992)
27. Lu, C.S.: On the security of structural information extraction/embedding for images. In: Proceedings of the IEEE International Symposium on Circuits and Systems, vol. II, pp. 169-172. Vancouver, Canada (2004)
28. Lu, C.S., Hsu, C.Y., Sun, S.W., Chang, P.C.: Robust mesh-based hashing for copy detection and tracing of images. In: Proceedings of the IEEE International Conference on Multimedia and Expo: Special Session on Media Identification. Taipei, Taiwan (2004)