A Robust and Fast Video Copy Detection System Using Content-Based Fingerprinting

Mani Malek Esmaeili, Mehrdad Fatourechi, and Rabab Kreidieh Ward, *Fellow, IEEE*

***Abstract*—A video copy detection system that is based on content fingerprinting and can be used for video indexing and copyright applications is proposed. The system relies on a fingerprint extraction algorithm followed by a fast approximate search algorithm. The fingerprint extraction algorithm extracts compact content-based signatures from special images constructed from the video. Each such image represents a short segment of the video and contains temporal as well as spatial information about the video segment. These images are denoted by temporally informative representative images. To find whether a query video (or a part of it) is copied from a video in a video database, the fingerprints of all the videos in the database are extracted and stored in advance. The search algorithm searches the stored fingerprints to find close enough matches for the fingerprints of the query video. The proposed fast approximate search algorithm facilitates the online application of the system to a large video database of tens of millions of fingerprints, so that a match (if it exists) is found in a few seconds. The proposed system is tested on a database of 200 videos in the presence of different types of distortions such as noise, changes in brightness/contrast, frame loss, shift, rotation, and time shift. It yields a high average true positive rate of 97.6% and a low average false positive rate of 1.0%. These results emphasize the robustness and discrimination properties of the proposed copy detection system. As security of a fingerprinting system is important for certain applications such as copyright protections, a secure version of the system is also presented.**

***Index Terms*—Content-based fingerprinting, multimedia duplicate detection, multimedia fingerprinting, robust video hashing, video copy detection, video copy retrieval.**

I. Introduction

TENS of thousands of videos are being uploaded to the Internet and shared every day. A considerable number of these videos are illegal copies or manipulated versions of existing media, making copyright management on the Internet a complicated process. Today's widespread video copyright infringement calls for the development of fast and accurate copy-detection algorithms. As video is the most complex type of digital media, it has so far received the least attention regarding copyright management. Because videos are available in different formats, it is more efficient to base the copy detection process on the content of the video rather than its name, description, or binary representation. Multimedia fingerprinting (also known as robust hashing) has been recently proposed for this purpose [1]. A fingerprint is a content-based signature derived from a video (or other form of a multimedia asset) so that it specifically represents the video or asset. To find a copy of a query video in a video database, one can search for a close match of its fingerprint in the corresponding fingerprint database (extracted from the videos in the database). Closeness of two fingerprints represents a similarity between the corresponding videos; two perceptually different videos should have different fingerprints.

A. Properties of Fingerprints

A fingerprint should be robust to the content-preserving distortions present in a video. It should also be discriminant, easy to compute, compact, and easy to search for in a large database. In some applications such as copyright protection, the fingerprinting system should also be secure. *Robustness* of a fingerprint requires that it changes as little as possible when the corresponding video is subjected to content-preserving operations, i.e., operations that do not affect the *perceptual* content of the video. Content-preserving attacks (distortions) are changes that are made to the video unintentionally or intentionally by users of video-sharing websites. These changes can include format changes, signal processing operations, changes in brightness/contrast, added noise, rotation, cropping, logo insertion, compression, etc. The fingerprints should also be *discriminant*, to ensure that two perceptually different videos have distinguishable fingerprints. Because a change in the content can be considered as an extreme distortion of the video, there is a trade-off between robustness and discrimination. As a fingerprinting algorithm becomes more robust to distortions, it becomes less sensitive to changes in the content, i.e., it has less discrimination ability.

The fingerprint should also be *easy to compute*. More specifically, for online applications, a fingerprinting algorithm should be able to extract the signatures as the video is being uploaded. A computationally demanding algorithm is not suitable for online applications, where thousands of videos need to be examined simultaneously in order to find possible copyright infringements. For the same reason, fingerprints should be *compact* as well. If a fingerprint is not compact, finding a match for it in a very large database can become a time-consuming process. It should be noted that the compactness of a fingerprint does not guarantee that it can be easily matched or found in a large database. The fingerprint structure should be designed to allow utilization of fast approximate search algorithms, as will be discussed later.

Manuscript received January 07, 2010; revised September 30, 2010; accepted October 29, 2010. Date of publication December 20, 2010; date of current version February 16, 2011. This work was supported in part by NSERC Grant STPGP 365164-08. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Min Wu.

The authors are with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, V6T 1Z4, Canada (e-mail: manim@ece.ubc.ca; mehrdadf@ece.ubc.ca; rababw@ece.ubc.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2010.2097593

For some applications, the fingerprinting system should be *secure*, so as to prevent an adversary from tampering with it. Security is specifically important for copy-detection applications. The more secure a multimedia fingerprinting algorithm is, the more difficult it is for an adversary to generate similar fingerprints for different videos and thus manipulate the copy-detection system. For indexing applications, however, security of a fingerprinting system does not pose a problem.

B. Types of Fingerprints

Existing video fingerprint extraction algorithms can be classified into four groups based on the features they extract: color-space-based, temporal, spatial, and spatio-temporal (see [2] and [3] for a survey of existing video fingerprinting methods). *Color-space-based fingerprints* are among the first feature extraction methods used for video fingerprinting [2]. They are mostly derived from the histograms of the colors in specific regions in time and/or space within the video. Since color features change with different video formats [4], these features have not been very popular. Another drawback of color features is that they are not applicable to black and white videos. For this reason, most of the video fingerprinting systems are designed so that they can be applied to the luminance (the gray level) value of the frames. *Temporal fingerprints* are extracted from the characteristics of a video sequence over time [5]. These features usually work well with long video sequences, but do not perform well for short video clips since they do not contain sufficient discriminant temporal information. Because short video clips occupy a large share of online video databases, temporal fingerprints alone do not suit online applications.

Spatial fingerprints are features derived from each frame or from a key frame. They are widely used for both video and image fingerprinting. There is a large body of research in the area of image fingerprinting and many researchers have extended the concepts developed for image fingerprinting to the video fingerprinting field [6]–[13]. Spatial fingerprints can be further subdivided into global and local fingerprints. Global fingerprints depict the global properties of a frame or a subsection of it (e.g., image histograms), while local fingerprints usually represent local information around some interest points within a frame (e.g., edges, corners, etc.). These interest-points are conventionally used in the multimedia retrieval community mainly for object retrieval purposes. However, with adequate postprocessing they can also be applied to multimedia copy detection and many researchers have recently considered them for this task [14]–[16], and [17]. Many of these works use SIFT features [18] which are proven to be very robust against many distortions and have been successfully applied for retrieval purposes. SIFT features provide robustness to content-changing attacks, which is not achievable with global features. However, for a short segment (e.g., 1 s) of a video, there are a large number (thousands) of SIFT features, this makes them impractical from a memory management point of view when applied to large video databases containing billions of hours of videos [19]. Furthermore, the amount of postprocessing required by these methods makes them less appealing for copy detection purposes. We thus investigate using of global features in this paper, as they do not have the above limitations.

C. Summary and Organization of the Paper

One shortcoming of spatial fingerprints is their inability to capture the video's temporal information, which is an important discriminating factor. *Spatio-temporal fingerprints* that contain both spatial and temporal information about the video are thus expected to perform better than fingerprints that use only spatial or temporal fingerprints. In this paper, we have adopted spatio-temporal fingerprints because of their comprehensiveness. Some spatio-temporal algorithms consider a video as a three-dimensional (3-D) matrix and extract 3-D transform-based features [20], [21]. Others use spatio-temporal interest-point descriptors to generate the fingerprints [22], [23]. Ordinal spatio-temporal features have also been used in the literature [24]. Applying a 3-D transform to a video is a computationally demanding process and may pose problems in online applications. In [25], we propose a method for forming temporally informative representative images (TIRIs) from a video sequence. As a TIRI contains spatial and temporal information of a short segment of a video sequence, the spatial feature extracted from a TIRI would also contain temporal information. Based on TIRIs, in [26], we have proposed an efficient fingerprinting algorithm (TIRI-DCT) and compared it to that proposed by Coskun *et al.* in [20] [three-dimensional discrete cosine transform (3D-DCT)]. We showed that our proposed algorithm outperformed 3D-DCT except when subject to geometric attacks. We will discuss an improved version of this algorithm (TIRI-DCT) in Section II-B and demonstrate that our proposed algorithm yields consistently better performance compared to 3D-DCT.

Many of the existing algorithms for video fingerprinting have focused on the extraction of robust and discriminant features from a video. However, when designing a practical fingerprinting system for online video databases with a huge number of videos, the computational bottleneck is the search time in the matching process rather than the fingerprint extraction time. So far only few papers have addressed this problem. In this paper, we also address the search time of our proposed fingerprinting algorithm. We propose two fast search methods. The first is a generalization of [1] and the second is based on a novel approach involving clustering of the fingerprints in the database. We also show that the second method yields superior results. Our final proposed fingerprinting system, i.e., the TIRI-DCT method along with the search algorithm specifically developed for it introduces a fingerprinting system that is robust, discriminant, and fast. Fig. 1 shows the overall structure of this fingerprinting system.

The security of fingerprinting has been addressed by only few studies so far [20], [27]–[29]. It is, however, important for copyright protection applications as the system may be compromised by fake fingerprints generated by an adversary who knows the algorithm. By using a secret key, we show in Section V that the security of our fingerprinting system can be increased. We also articulate the trade-off between security and robustness.

The organization of the paper is as follows: in Section II, we briefly discuss how TIRIs that form the basis of the TIRI-DCT fingerprinting algorithm are generated. We also briefly explain the 3D-DCT algorithm, and TIRI-DCT in detail. In Section III,

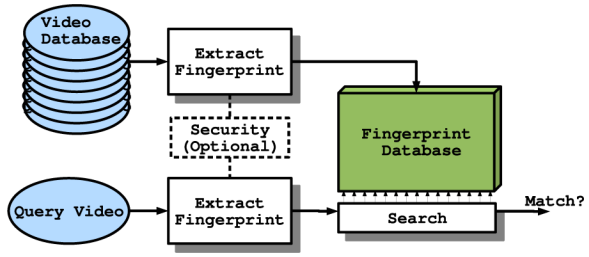

Fig. 1. Schematic of a complete fingerprinting system.

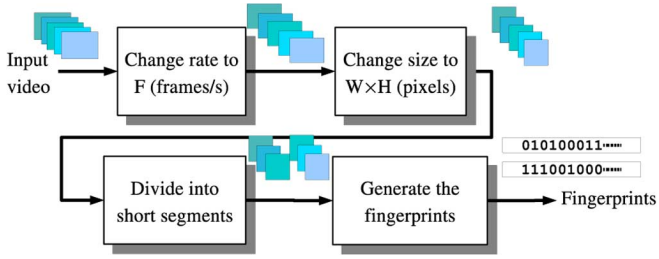

Fig. 2. Preprocessing steps employed in the discussed fingerprinting algorithms.

two fast search algorithms are proposed and analyzed. Simulation results are then presented in Section IV. In Section V, a secure version of the proposed fingerprinting system is introduced and analyzed. Finally, the conclusions are discussed in Section VI.

II. Extracting Robust and Discriminant Fingerprints

In this section, we first briefly explain an established fingerprinting algorithm [20], which will be used as the basis of the comparisons in our simulations. We then describe how we generate TIRIs, and discuss the details of our proposed fingerprinting algorithm. Before extracting the fingerprints, we preprocess the video signals. Copies of the same video with different frame sizes and frame rates usually exist in the same video database. As a result, a fingerprinting algorithm should be robust to changes in the frame size as well as the frame rate. Down-sampling can increase the robustness of a fingerprinting algorithm to these changes. As shown in Fig. 2, each video is down-sampled both in time and space. Prior to down-sampling, a Gaussian smoothing filter is applied in both domains to prevent aliasing. This down-sampling process provides the fingerprinting algorithm with inputs of fixed size ($W \times H$ pixels) and fixed rate (F frames/second). We choose $W = 144$, $H = 176$ (the size of QCIF sequences used widely in the video processing community), and $F = 4$. These parameter values have been chosen experimentally. After preprocessing, the video frames are divided into *overlapping* segments of fixed-length, each containing J frames. The fingerprinting algorithms are applied to these segments. The amount of overlapping is experimentally chosen to be 50%. Overlapping reduces the sensitivity of the fingerprints to the "synchronization problem" which we refer to as "time shift" in this paper. In Section III, we briefly explain the algorithm proposed in [20], which we call 3D-DCT in the rest of the paper.

A. Spatio-Temporal Fingerprinting Using 3D-DCT

Coskun *et al.* in [20] consider a video as a three-dimensional (3-D) matrix of luminance values. After the preprocessing phase described above, they apply a 3D-DCT to videos to extract spatio-temporal features. Binary fingerprints are then derived by thresholding the low-frequency coefficients of the transform, as shown in Fig. 3. The threshold is the median value of the selected coefficients. As a result, the generated fingerprint (hash) has an equal number of 0's and 1's. With this property, the maximum number of different fingerprints that can be generated from a binary vector of length L drops from 2^L to $\binom{L}{L/2}$. This property increases the robustness of the fingerprints; however, it decreases their discrimination. The robustness and discrimination of a fingerprint also depend on L, which is determined by the size of the low-frequency block. In [20], the 3D-DCT was shown to be resistant to different types of distortions that can happen to video signals, including changes in brightness/contrast, added Gaussian noise, rotation, spatial/temporal shift, and frame loss. A secure version of the algorithm, called random basis transform (RBT) was also proposed in [20] based on randomizing the frequency of the cosine basis functions of the DCT transform. This secure version, however, had a lower performance compared to the insecure version. In the rest of this section, we first discuss generating TIRI images to capture the temporal information in a video. We then provide the details of our proposed fingerprinting algorithm based on TIRIs.

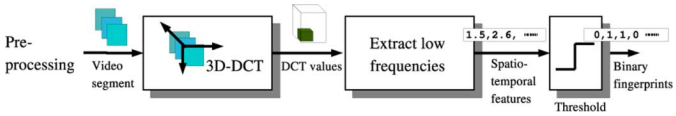

Fig. 3. Schematic of the 3D-DCT algorithm proposed in [20].

B. Spatio-Temporal Fingerprinting Using TIRIs

1) Generating TIRIs: This method (also used in [26]) calculates a *weighted average* of the frames to generate a *representative image*. The resulting image is basically a blurred image that contains information about possible existing motions in a video sequence. The TIRI is thus generated as follows: let $l_{m,n,k}$ be the luminance value of the (m,n)th pixel of the kth frame in a set of J frames. The pixels of TIRI are then obtained as a weighted sum of the frames

$$l'_{m,n} = \sum_{k=1}^{J} w_k \, l_{m,n,k}. \tag{1}$$

We have examined different weight factors (constant, linear, and exponential) and observed that *exponential weighting* generates images that best capture the *motion*. Fig. 4(a)–(c) show the first, the middle, and the last frames respectively, of a 1-s video segment (30 frames) extracted from the sequence COASTGUARD-QCIF ([30]). The corresponding TIRIs using three different weighting functions are shown in Fig. 4(d)–(f): constant [Fig. 4(d)], linear [Fig. 4(e)], and exponential [Fig. 4(f)]. As mentioned earlier, Fig. 4 shows that the exponential weighting function produces perceptually better results. Our experiments with other videos led to the same conclusion, thus we have chosen the exponential weighting function ($w_k = \gamma^k$) for generating TIRIs. A very large γ, or one close to 0, generates a TIRI with detailed spatial

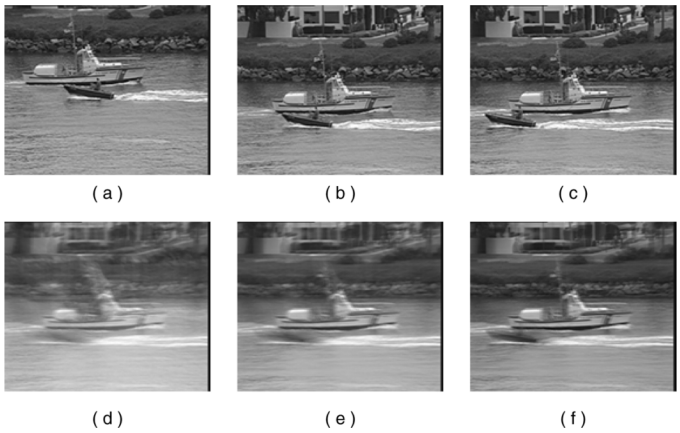

Fig. 4. Frames (a) 61, (b) 75, and (c) 90 of the sequence COASTGUARD-QCIF [30] and the resulting TIRIs with different weighting functions: (d) $w_k = 1$ (constant), (e) $w_k = k$ (linear), (f) $w_k = 1.2^k$ (exponential).

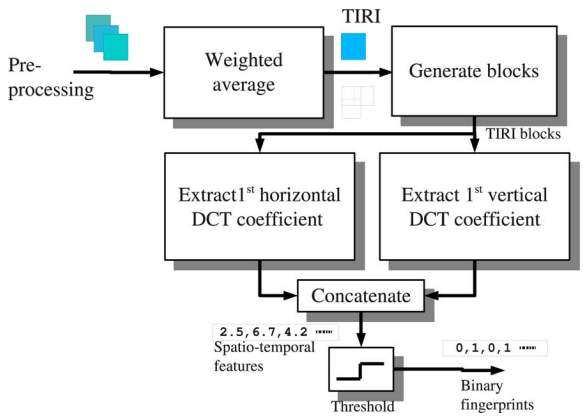

Fig. 5. Schematic of the TIRI-DCT algorithm.

information and low temporal information, resulting in a more discriminant representative. On the other hand, a γ close to 1 (giving the same weight to all the frames along the time line) results in a blurred image that is a more robust representative (containing averaged temporal information). By changing γ from 0 to 1, we can move from a single frame selection (high spatial information) to selecting all frames with equal weights (high temporal information). The effect of using different γ values on the performance of the fingerprinting system is demonstrated in Section IV.

2) TIRI-DCT Algorithm: Fig. 5 shows the block diagram of our proposed approach based on TIRIs [26]. Fig. 6 shows the details of this algorithm. As explained in Fig. 6, features are derived by applying a 2D-DCT on overlapping blocks of size $2w \times 2w$ from each TIRI (with 50% overlap). The first horizontal and the first vertical DCT coefficients (features) are then extracted from each block. The value of the features from all the blocks are concatenated to form the feature vector. Each feature is then compared to a threshold (which is the median value of the feature vector) and a binary fingerprint is generated.

A drawback of 3D-DCT is its *binarization* phase, where coefficients of different frequencies are binarized based on their median value. Fig. 7 shows the range of DCT coefficients belonging to different frequencies derived from 200 video frames. It can be seen that different coefficients have different ranges, thus having a common threshold for the binarization process is far from optimal. This problem does not exist with TIRI-DCT. For TIRI-DCT, all features are in the same frequency range, and binarization based on a common threshold is thus justified.

1: Generate TIRIs from each segment of J frames using $w_k = \gamma^k$ (Eq. 1).
2: Segment each TIRI into overlapping blocks of size $2w \times 2w$, as follows:

$$B^{i,j} = \{l'_{x,y} \,|\, x \in iw \pm w, y \in jw \pm w\}$$

where, $i \in \{0, 1, 2, \ldots, W/w - 1\}$ and $j \in \{0, 1, 2, \ldots, H/w - 1\}$. The TIRI image l' is padded with zeros where the indexes are outside of the image boundaries.
3: Extract two DCT coefficients from each block. These are the first horizontal and vertical coefficients adjacent to the DC coefficient. The first vertical frequency, $\alpha_{i,j}$, can be found for $B^{i,j}$ as follows:

$$\alpha_{i,j} = \mathbf{v}^T B^{i,j} \mathbf{1}$$

Where $\mathbf{v} = [\cos(0.5\pi/2w), \cos(1.5\pi/2w), \ldots, \cos(\pi - 0.5\pi/2w)]^T$ and $\mathbf{1}$ is a column vector of all ones. Similarly, the first horizontal frequency $\beta_{i,j}$ can be found as follows:

$$\beta_{i,j} = \mathbf{1}^T B^{i,j} \mathbf{v}$$

4: Concatenate all the coefficients to form vector f.
5: Find m, which is the median of the elements of f.
6: Generate the binary hash h from f using the following formula:

$$h_k = \begin{cases} 1, & f_k \geq m \\ 0, & f_k < m \end{cases}$$

Fig. 6. TIRI-DCT algorithm.

III. Fast Matching of Fingerprints Within a Large Video Database

As shown in Fig. 1, in order to determine whether a query video is an attacked version of a video in a database or not, its fingerprint is first extracted. The fingerprint database (previously created from the videos in the video database) is then searched for the closest fingerprint to the extracted query fingerprint. It should be mentioned that in copy detection, the problem is to determine if a specific query video is a pirated version of a video in the database. On the other hand, the problem of finding all copies of a video in a database is called copy retrieval and requires a different approach which is not the concern of this paper.

Fingerprints of two different copies of the same video content are similar but not necessarily identical. This is why we seek a close match of the query in the fingerprint database and not an exact match. This problem is a similarity search problem or a

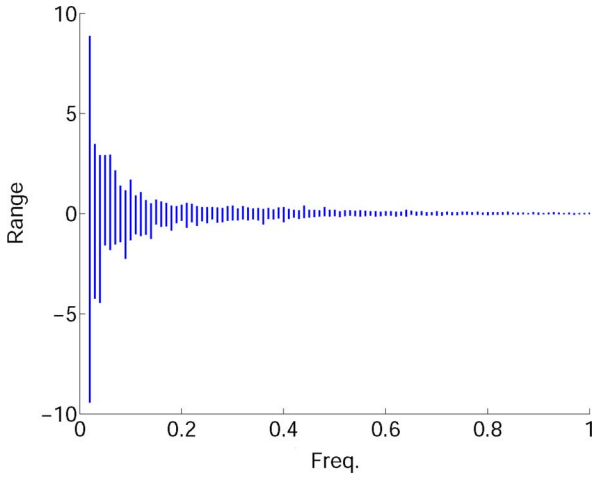

Fig. 7. Range of (non-DC) DCT coefficients corresponding to different (normalized) frequencies.

nearest neighbor problem in the binary space [1]. The trivial solution to this problem is to apply an exhaustive search which unfortunately imposes a large computational load on the fingerprinting system. Therefore, instead of the time-consuming "exact" nearest neighbor search, a fast "approximate" algorithm should be deployed.

In real-world applications, the size of online video databases can reach tens of millions of videos, which translates into a very large fingerprint database size. This means that even if the fingerprint of a query video can be extracted very quickly, searching the fingerprint database to find a match may take a long time. For online applications, however, a reliable match should be found in almost real-time. This is why fingerprint matching forms a practical bottleneck for online fingerprinting systems.

There is a large body of research on fast and reliable similarity search. Muja *et al.* have conducted a comprehensive study on state-of-the-art similarity search algorithms in Euclidian spaces [31]. There are a number of studies on similarity search on binary spaces as well [1], [32], [33]. However, only few papers in the video fingerprinting area have considered the fast search aspect in their design. Most of them have used a simple exhaustive search method which has a complexity of $O(N)$, where N is the number of the fingerprints in the database. As an example of a fast search algorithm, Oostveen *et al.* proposed a search algorithm for their video fingerprinting algorithm, based on the inverted file technique [1]. In Section III-A, we develop a modified version of [1] so that the proposed algorithm can be used in any fingerprinting search. In Section III-B, we propose a similarity search algorithm that uses a fingerprint clustering approach. We show the results of applying these two algorithms on the fingerprints derived by TIRI-DCT in Section IV-B. We should state that unlike most fingerprinting algorithms, both 3D-DCT and TIRI-DCT generate binary fingerprints that do not represent numerical values. This means that the Hamming distance (which is realizable using XOR binary operation) can be used as a metric for measuring the similarity of the fingerprints. This has the advantage of requiring less search time than algorithms that derive numerical features and subsequently use the Euclidian distance to measure similarities.

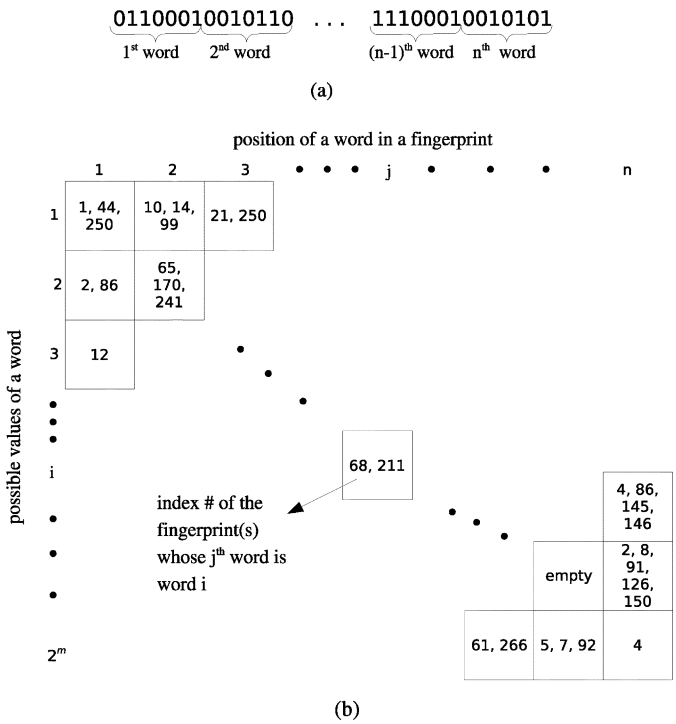

Fig. 8. (a) Dividing the fingerprint into words. (b) Sample inverted file for the fingerprint database.

A. Inverted-File-Based Similarity Search

This search method is based on the idea that for two fingerprints which are similar enough to be considered as matches, the probability of an exact match between smaller sub-blocks of those fingerprints is high [1]. We divide each fingerprint into small nonoverlapping blocks of m bits. We call these small blocks *words* (thus, there are 2^m possible words). Words are then used to create an inverted file from the fingerprints of database. All fingerprints have equal lengths, so the inverted file can be represented as a table of size $2^m \times n$, where n is the number of words in a fingerprint of length L ($n = \lfloor L/m \rfloor$). The horizontal dimension of this table refers to the position of a word inside a fingerprint, and the vertical direction corresponds to possible values of the word. To generate this table, we start with the first word of each fingerprint [Fig. 8(a)], and add the index of the fingerprint to the entry in the first column corresponding to the value of this word. We continue this process for all the words in each fingerprint and all the columns in the inverted file table.

A sample of the generated inverted file is shown in Fig. 8(b). Entry (i, j) in the table is a list of the indices of all the fingerprints that their jth word is word i. To find a query fingerprint in the database, first the fingerprint is divided into n words (of m bits). The query is then compared to all the fingerprints that start with the same word. The indices of these fingerprints are found from the corresponding entry in the first column of the inverted file table. The Hamming distance between these fingerprints and the query is then calculated. If a fingerprint has a Hamming distance of less than some predefined threshold, it will be announced as the match. If no such match is found, the procedure is repeated for the fingerprints that have exactly the same second word as the query's second word (the indices of these fingerprints are read from the corresponding entry in the

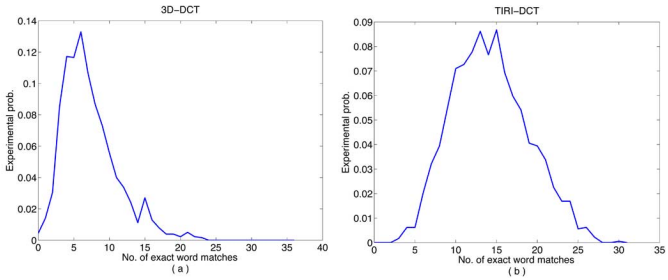

Fig. 9. Histogram of the number of exactly matching words between two similar fingerprints. (a) 3D-DCT; (b) TIRI-DCT.

second column of the inverted file table). This procedure is continued until a match is found or the last word is examined. When no match is found in the end, it is stated that the query does not belong to the database. The above algorithm can be used for finding the nearest neighbor of any binary fingerprint within a database. To show the validity of the assumption that two similar fingerprints have at least one exactly matching word, we ran some simulations. Fig. 9 shows the experimental probability distribution of the number of identical words between fingerprints of two similar videos (two different copies of the same video content) for both 3D-DCT and TIRI-DCT. It can be seen from the figure that the probability of not having any exactly matching words between two similar fingerprints is almost zero for both algorithms. In Section IV, we will demonstrate the results of applying the proposed search, to TIRI-DCT and compare the results to the exhaustive search both in terms of search time and performance.

If the word length is chosen correctly, then the worst-case scenario where the algorithm examines every word of the query will require searching $O(LN/(m2^m))$ queries. For an extremely robust fingerprinting algorithm that has a very low threshold, a large m can be chosen to reduce the search speed by several orders of magnitude. However, there is a trade-off between the speed and the required memory for storing the inverted file table. It is not practical for word lengths of larger than 16 bits to have a complete inverted file and only the existing words should be indexed. An analysis of a similar search algorithm used in a different context can be found in [34].

Assuming that the fingerprinting algorithm is perfect, i.e., no two perceptually different videos have fingerprints that are closer than a Hamming distance of th, the algorithm is guaranteed to find the correct match, if $th < n(= L/m)$. However, if $th >= n$, then the algorithm may generate false negatives due to the fact that there may exist a mismatch in each word. To calculate the probability of false negative, we assume that a fingerprint exists in the database that has a Hamming distance of th with the query fingerprint (i.e., they both belong to the same perceptual content). We also assume that differences (errors) are uniformly distributed along the fingerprint (a simple permutation can assure that this assumption holds true). Here, p_{FR} (the probability of false rejection) can be computed as

$$p_{\mathrm{FR}} = \sum_{k=0}^{k_{\max}} (-1)^k \binom{n}{k} \frac{\binom{L-k*m}{th}}{\binom{L}{th}} \tag{2}$$

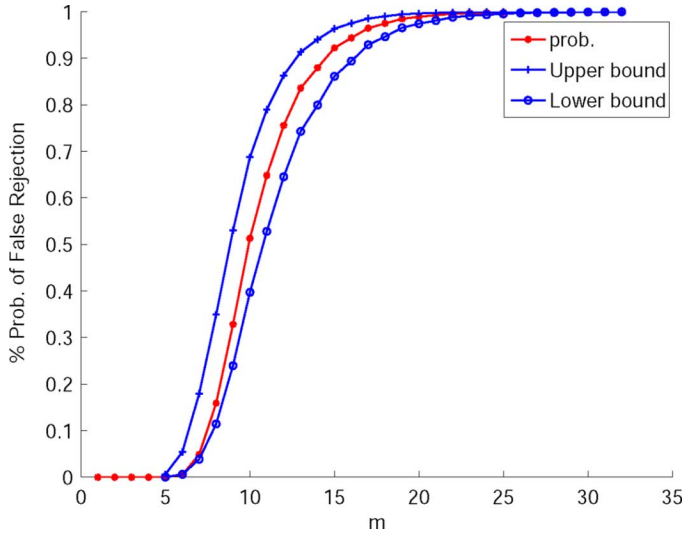

Fig. 10. Probability of false rejection for the inverted-file-based approach for different word lengths m ($L = 128$, $th = 32$).

which is upper bounded by

$$\mathrm{Ub}_{\mathrm{FR}} = \sum_{k=0}^{k_{\max}} (-1)^k \left(1 - \frac{m}{L - th + 1}\right)^{k*th} \tag{3}$$

and lower bounded by

$$\mathrm{Lb}_{\mathrm{FR}} = \sum_{k=0}^{k_{\max}} (-1)^k \left(1 - \frac{m}{L}\right)^{k*th} \tag{4}$$

where $k_{\max} = \max\{k \mid L - k*m \geq th\}$. Fig. 10 shows the false rejection (false negative) probability of the inverted-file-based algorithm, for fingerprints of length $L = 128$ bits and a detection threshold of $th = 0.25 * L$. Finally, it is worth mentioning that when the query belongs to the database, the algorithm usually returns the result after examining the first few words. The probability that the algorithm continues its search by examining the second word, i.e., the probability that the first word has an error (denoted by p_{w_1}) is

$$p_{w_1} = 1 - \overline{p_{w_1}} = 1 - \frac{\binom{L-m}{th}}{\binom{L}{th}}. \tag{5}$$

The probability that the algorithm continues its search to the third word is even lower. For a query that is closer to one of the fingerprints in the database by th, assuming that the algorithm examines only the first two words, the algorithm searches an expected number of $(N)/(2^m)(1+p_{w_1})$ queries. In Section III-B, we propose another approximate search algorithm and we compare the estimated run-time of both algorithms in terms of the expected number of queries that must be searched.

B. Cluster-Based Similarity Search

In this section, we propose another similarity search algorithm for binary fingerprints. Our main idea is to use clustering to reduce the number of queries that are examined within the database. By assigning each fingerprint to one and only one cluster (out of K clusters), the fingerprints in the database will

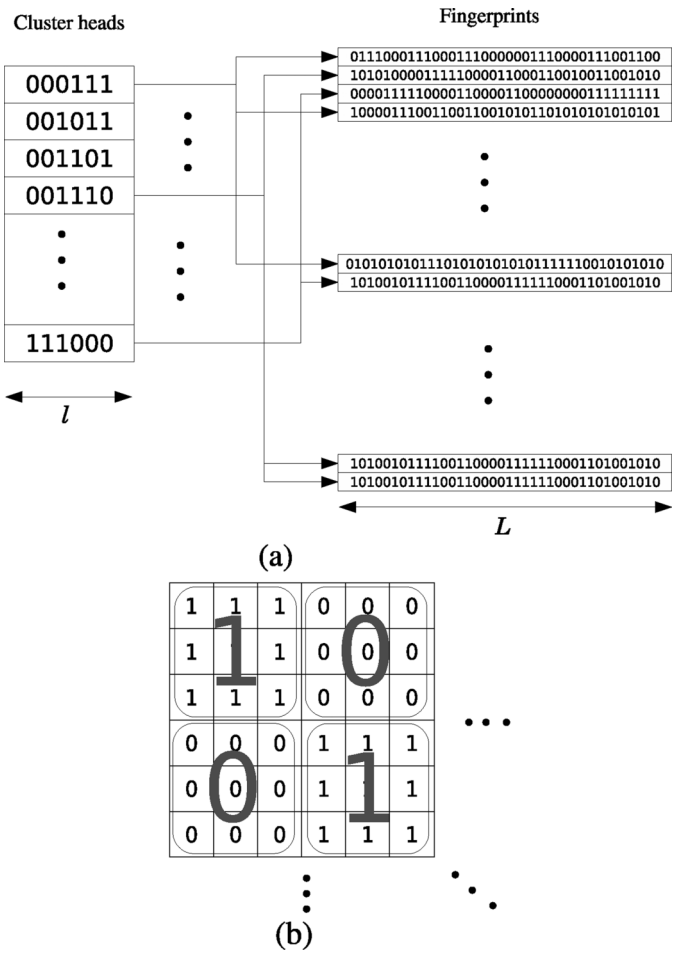

Fig. 11. (a) Clustering the fingerprints for TIRI-DCT. (b) Expanding a cluster head to compare it with a fingerprint.

be clustered into K nonoverlapping groups. To do so, a centroid is chosen for each cluster, termed the *cluster head*. A fingerprint will be assigned to cluster k if it is closest to this cluster's head [see Fig. 11(a)]. To determine if a query fingerprint matches a fingerprint in the database, the cluster head closest to the query is found. All the fingerprints (of the videos in the database) belonging to this cluster are then searched to find a match, i.e., the one which has the minimum Hamming distance (of less than a certain threshold) from the query. If a match is not found, the cluster that is the second closest to the query is examined. This process continues until a match is found or the farthest cluster is examined. In the latter case, the query is declared to be out of the database.

The cluster heads should be chosen such that a small change in the fingerprint does not result in the fingerprint being assigned to another cluster. In our general setting, we choose cluster heads (centers) as all the binary vectors with length $l \ll L$. To assign a fingerprint to a cluster, the fingerprint is first divided into segments (words) of length $m = L/l$. Each word is then represented by one bit in the l-bit cluster head, depending on the majority of word's bit values; e.g., it is represented by 1, if it has more than (or equal to) $m/2$ 1's and it is represented by 0, if it has less than $m/2$ 1's. Equivalently, each bit of the cluster head can be replicated m times and the Hamming

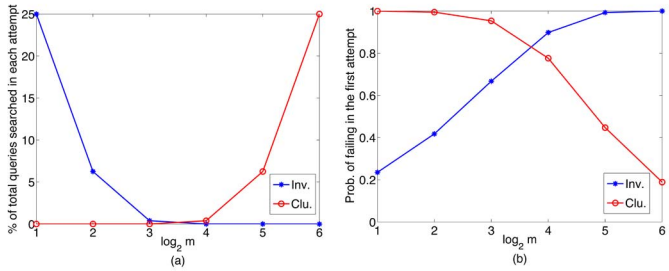

Fig. 12. Comparing the runtime of the clustering-based search method with that of the inverted-file-based method for different values of $m(L = 128,\ th = 16)$. (a) Approximate percentage of queries searched with each attempt. (b) Probability of failing the first attempt.

distance between the expanded $L = m * l$ bit version of all the cluster heads and the fingerprint is calculated. The cluster head closest to the fingerprint is then assigned to that fingerprint (at the end of this section, we will explain a version of cluster assignment developed for TIRIs).

As stated earlier, if the query is not matched to any fingerprint in a certain cluster, the algorithm continues its search over other clusters starting from the closer ones. For this algorithm, in the worst-case scenario, all the fingerprints will be searched. So unlike the case of the inverted-file-based search, the clustering-based search is guaranteed to return a match if it exists. Thus, in the worst-case scenario, this algorithm has a runtime of $O(N)$, which is not better than the exhaustive search. However, similar to the case of the inverted-file-based search, the algorithm in practice returns a match (if one exists) after examining the first few clusters, thus reducing the actual number of comparisons to $O(N/2^l)$ queries. The probability that the algorithm fails to find the fingerprint in the closest (first) cluster is upperbounded by the probability that a word in a fingerprint has approximately an equal number of 1's and 0's multiplied by the expected number of words that have errors. This probability, denoted by p_{change}, can be approximated as

$$p_{\text{change}} = 1 - \left(1 - \text{erf}\left(\frac{1}{\sqrt{2m}}\right) \cdot (1 - e^{-m.th/L})\right)^l. \quad (6)$$

The first component $\text{erf}(\frac{1}{\sqrt{2m}})$ is derived using a Gaussian approximation assuming m is 7 bits or more. The second component $(1 - e^{-m.th/L})$ is derived by approximating the spatial distance between two errors by an exponential distribution assuming that errors happen rarely and independently with an average of th/L.

The above approximation is not good for small m's and the actual probability has a complicated form. Fig. 12(a) shows the approximate number of queries that are searched in each attempt for the proposed cluster-based approach as well as the inverted-file-based approach (assuming that the fingerprints are spread randomly in the binary space). Fig. 12(b) shows the probability that each approach fails in the first attempt. The higher this probability, the more likely it is for the algorithm to continue its search through other attempts (next clusters, or table columns). It can be seen that the cluster-based approach is very likely to find the match in its first attempt when m is large (low probability of failing); however, percentage of the total queries

TABLE I
ATTACK PARAMETER VALUES USED IN THE EVALUATION OF THE PROPOSED ALGORITHMS

Attack	Effect	Min	Max
Noise (σ)	$l'_{m,n,k} = l_{m,n,k} + G(0, \sigma)$	0	100
Brightness (b)	$l'_{m,n,k} = l_{m,n,k} + b\,\mu_k$	-0.7	0.7
Contrast (c)	$l'_{m,n,k} = c\,(l_{m,n,k} - 127.5)$	0.5	2
Rotate (r)	Rotates the whole frame r°	-5	5
Time shift (δ)	Video is shifted δ seconds in time	-0.5	0.5
Spatial shift (sr, sd)	Shifts the frame $sr\%$ right and $sd\%$ down	-4	4
Frame loss (fd)	$fd\%$ of the frames are randomly dropped	0	70

that should be searched in the first cluster is very large compared to when m' is small. Please note that an exhaustive approach searches 100% of the queries. For the inverted-file-based method, this trade-off can also be seen, e.g., when the word length is small the algorithm is likely to find a match in the first attempt, but it has to search a larger number of queries.

To increase the performance of the search, an improved version of clustering for TIRI-based fingerprinting is used throughout this paper. The cluster heads are chosen as those binary vectors of even length $l \ll L$ with an equal number $(l/2)$ of 0's and 1's. This procedure will result in fingerprints being divided into $\binom{l}{l/2}$ clusters. We need to determine which cluster head is closest to a fingerprint. To compare two binary vectors of different length, we propose the following procedure adapted for fingerprints derived by TIRI-DCT. As presented, to derive an L-bit fingerprint, TIRI-DCT divides a TIRI to $L/2$ overlapping blocks. Two bits are then derived from each block (one representing a horizontal feature and one representing a vertical one). To compare a fingerprint to a cluster head, we use only one of these features for simplicity, e.g., horizontal bits. We then expand each cluster head of length l to a bit vector of length $L/2$. To do so, we first assign each n adjacent blocks to a larger block $(n = L/(2l))$, as shown in Fig. 11(b). We then assign each bit of the cluster head to one of these larger blocks. Finally, we assign all the original blocks with a bit value equal to their corresponding large block and concatenate all these bit values to form an expanded cluster head of length $L/2$. Fig. 11(b) demonstrates how this expansion is done. Following this process, the Hamming distance can be calculated to measure the closeness of the fingerprints and the cluster heads. The proposed cluster-based search reduces the search time approximately proportional to the number of clusters.

IV. SIMULATION RESULTS

In this section, we evaluate the performance of the proposed algorithms using a video database. We first compare the raw performance of TIRI-DCT with 3D-DCT in Section IV-A. For comparisons, we use an exhaustive search as it gives the most accurate results. We will show that TIRI-DCT is faster than 3D-DCT while maintaining a very good performance over the range of the studied attacks. Next, in Section IV-B we evaluate the performance of the approximate search techniques proposed in Section III. We have implemented these algorithms as the search engine for the fingerprints derived by TIRI-DCT, and compared the results to the exhaustive search. We show that the cluster-based search method outperforms the inverted-file-based method both in search time and detection performance.

To evaluate the performance of the proposed algorithms, a database of 200 videos (collected from ReefVid [35]) was created. TIRI-DCT and 3D-DCT were separately applied to each video in the database. A fingerprint database was then formed from all of the generated fingerprints for each algorithm. Next, videos in the database were attacked (distorted) to generate query videos. The attacks studied included added Gaussian noise, changes in brightness/contrast, frame loss, shift in time/space, and rotation. Table I shows the attacks and their corresponding parameter values.

Before extracting the fingerprints every video is filtered and down-sampled to a frame rate of 4 frames/s and a frame size of 144×176. Videos are then segmented to 50% overlapping J frame sequences. We used $J = 8$, equivalent to 2-s segments. This value ensures that our fingerprinting algorithm is able to detect clips as short as 2 s. Note that the copyrighted videos may be cut into smaller clips and used within other videos (called *mash-ups*). This is why a fingerprinting algorithm should be able to recognize very short copies as well. To determine the values of γ (the exponential weighting basis for generating TIRIs) and w (half of the width of the blocks in TIRIs), we ran some simulations on a small database of 14 videos, separately collected from [30]. We generated 280 videos out of these videos by applying random combined attacks from Table I to them. For different values of γ and w, the experimental probability distribution of the Hamming distance between the similar and different videos was then calculated. Fig. 13(a) shows an example for $\gamma = 0.65$ and $w = 16$. Fig. 13(b) shows the receiver operating characteristic (ROC) curve for different values of $w(4 \leq w \leq 32)$ when $\gamma = 0.65$.

The figure also quantifies the trade-off between the robustness and the discrimination ability of our copy detection system. To evaluate the system's performance and to determine the suitable parameter values, we require a single combined metric. Here, we used the F-score (F_β), defined as follows [36]:

$$F_\beta = (1+\beta^2)\frac{\text{precision.recall}}{\beta^2\text{precision} + \text{recall}}. \tag{7}$$

In (7), β defines how much weight should be given to *recall* versus *precision*. Recall is the same as the true positive rate (a measure of robustness of the system). Precision is a measure of discrimination and is defined as the percentage of correct hits within all the detected copies. F-score of a clustering system is a number between 0 and 1, with 1 representing a perfect classification system that is completely robust and completely discriminant (100% precision and 100% recall). A low F-score close to 0 represents a poor system in terms of both robustness and discrimination. With a proper choice of β, the F-score can be

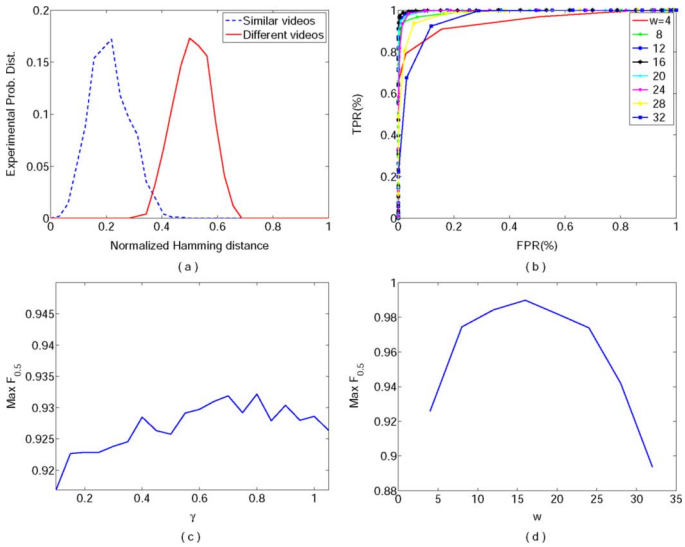

Fig. 13. (a) Distribution of the Hamming distance between fingerprints corresponding to similar (copied) and different video sequences. (b) ROC curve for TIRI-DCT for $w = 4 : 32$ and $\gamma = 0.65$. (c) Maximum $F_{0.5}$ for $0 < \gamma \leq 1$ and $w = 16$. (d) Maximum $F_{0.5}$ for $w = 4 : 4 : 32$ and $\gamma = 0.65$.

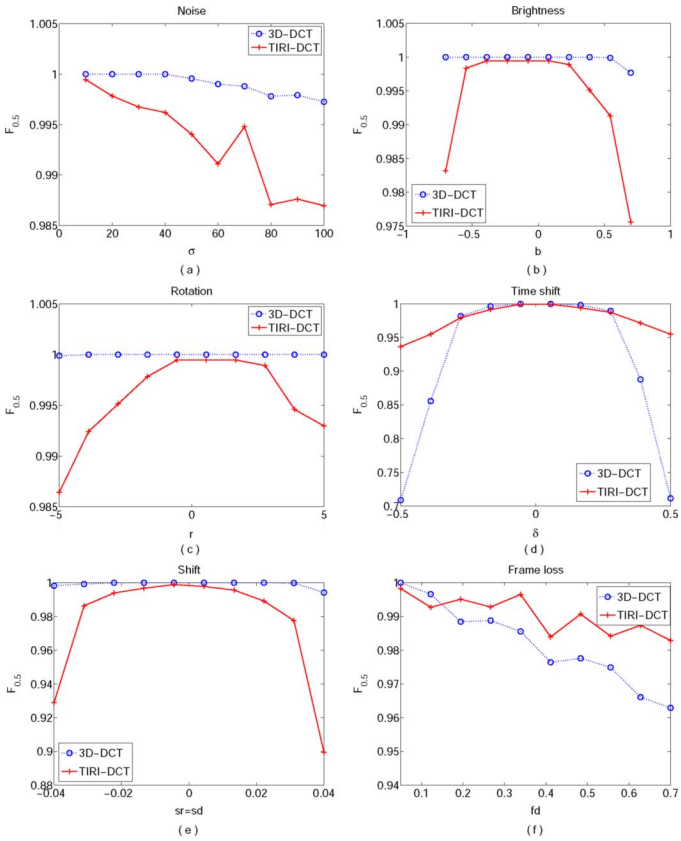

Fig. 14. F-score of 3D-DCT and TIRI DCT for different attack parameters from Table I: (a) Noise addition; (b) change in brightness; (c) rotation; (d) time shift; (e) spatial shift; (f) frame loss.

a valuable *single measure* to summarize the detection performance. A copy detection system should have high precision to minimize the amount of human interaction required. We used $\beta = 0.5$, which gives twice as much importance to precision as to recall. Fig. 13(c) and (d) shows the maximum $F_{0.5}$ for $0 < \gamma \leq 1$ at $w = 16$ and for $4 \leq w \leq 32$ at $\gamma = 0.65$. Our experiments show that $\gamma = 0.65$ and $w = 16$ give the best results, so we used these values for the TIRI-DCT algorithm. These parameter values result in a 126-bit hash for TIRI-DCT (ignoring the marginal blocks). For 3D-DCT, the first $6 \times 6 \times 4$ low frequency 3D-DCT coefficients were chosen to generate fingerprints of length 144 bits. $6 \times 6 \times 4$ is chosen through simulations so that both algorithms have approximately the same fingerprint length and at the same time the performance of the 3D-DCT is maximized. Experiments show that extracting more features (for 3D-DCT) from the temporal domain did not increase the precision and recall even for time shift attacks. This is because the temporal domain of a video is basically a low frequency domain (e.g., 3D-DCT's $F_{0.5}$ decreases by 16% if we used the first $4 \times 4 \times 8$ block instead).

To see if a query video is a pirated copy of a premium video content in the database, first the fingerprint of the query was extracted and the nearest neighbor of the extracted fingerprint was then sought in the fingerprint database. If the Hamming distance of this nearest neighbor with respect to the query fingerprint was less than some predefined threshold, the corresponding video was announced to be a match. Using the training database, the thresholds were set to 0.24 and 0.31 for TIRI-DCT and 3D-DCT respectively. Each fingerprinting algorithm was then used to locate all the attacked videos in the video database, and the detection performance of the algorithm was evaluated.

A. Performance Evaluation of the Fingerprint Extraction Algorithm

To get the most accurate results, in this section, we compare the performance of TIRI-DCT and 3D-DCT when an exhaustive search is used for searching the database. Table II shows the results of applying TIRI-DCT and 3D-DCT to the test database. Attacks were mounted independently on the videos to generate the queries. For each attack parameter, ten equally spaced values were chosen from the corresponding range in Table I. Table II reports the average true positive rate (TPR), false positive rate (FPR), and the $F_{0.5}$ over these values. As mentioned, we chose $\beta = 0.5$ to give precision twice the importance of recall. Fig. 14 shows the F-score for different attack parameters: noise addition, change in brightness, rotation, temporal/spatial shift, and frame loss.

Table II shows that both TIRI-DCT and 3D-DCT have an average F-score of 0.99. So both algorithms have a very good performance on average, but as it can be seen from the table that TIRI-DCT maintains this high performance for all the attacks, while the performance of 3D-DCT is degraded for time domain attacks. Fig. 14 shows that both algorithms are robust to noise addition, changes in brightness/contrast, and rotation with high F-scores. It can be seen from Table II and Fig. 14 that 3D-DCT is robust to geometrical and signal processing attacks, but is sensitive to time shifts and frame loss, both of which happen frequently when dealing with online video databases. Time shift can happen regularly when the beginning of a query video segment is not aligned with the beginning of the reference video segment in the video database. Frame loss is also a common type of attack that can happen with online video streams. This is because 3D-DCT relies on extracting too much detail along the time axis, which decreases the robustness of the algorithm.

TABLE II
COMPARING TIRI-DCT WITH 3D-DCT WHEN EXHAUSTIVE SEARCH IS USED

Attack	TPR (%)		FPR (%)		F-score	
	3D-DCT	TIRI-DCT	3D-DCT	TIRI-DCT	3D-DCT	TIRI-DCT
Noise	99.52	99.24	0.00	0.66	1.00	0.99
Brightness	99.90	99.40	0.01	0.60	1.00	0.99
Contrast	99.96	99.02	0.01	0.75	1.00	0.99
Rotation	99.99	99.57	0.00	0.43	1.00	1.00
Time shift	78.90	95.8	1.29	1.73	0.91	0.98
Spatial shift	99.71	96.25	0.03	1.85	1.00	0.98
Frame loss	93.21	98.39	0.42	0.78	0.98	0.99
Average	95.88	98.24	0.25	0.97	0.99	0.99

TABLE III
HASH EXTRACTION TIME (IN mS) FOR DIFFERENT SEGMENT SIZES

Method	Segment length in mS			
	1	2	3	4
3D-DCT	3.3	4.6	6.0	7.4
TIRI-DCT	1.4	1.5	1.6	1.8

TIRI-DCT demonstrates an acceptable performance for all attacks studied here. However, it should be noted that TIRI-DCT is more sensitive to spatial shifts than 3D-DCT. Again this is because 3D-DCT relies too much on temporal information and thus it is more robust to spatial attacks like shift. However, we should note that with digital transmissions spatial shift does not occur within online video databases.

The performance (for TIRI-DCT) reported above is achieved using short fingerprint lengths of 126 bits. By increasing the fingerprint length to 640 bits, we were able to decrease the FPR on average by about 70% (to a maximum of about 0.4%) without affecting the TPR, resulting in a 0.6% increase in the F-score. As mentioned in Section II-B2, the problem with the binarization scheme in 3D-DCT limits the number of coefficients and thus the fingerprint length that can be used with this method. Our results show that increasing the fingerprint length for 3D-DCT to about 600 bits has a negative effect on the TPR resulting in an approximate 7% decrease in the F-score.

Another important property of TIRI-DCT is that it is computationally less demanding than 3D-DCT. The latter applies a 3D-DCT transform requiring $O(WHJ\log(WHJ))$ computation time, whereas, TIRI-DCT applies a two-dimensional block DCT requiring $O(WH\log(WH))$ computation time. The above computational complexities are calculated based on the assumption that all the coefficients are derived. In practice, only the coefficients that are used should be computed and therefore the actual complexity depends on the number of coefficients used. For L-bit fingerprints, the computational complexity of the 3D-DCT would, therefore, be $O(\sqrt[3]{L}WHJ)$ whereas the computational complexity of TIRI-DCT would be $O(\max(\sqrt{L},J)WH)$.

Table III shows the fingerprint extraction time in mS (for a practical implementation) of both algorithms for different segment lengths. Algorithms were run on Matlab R2008b with a 64-bit PC with a 2.4-GHz CPU and 4 GB of RAM. Note that the proposed algorithms have a primary step of generating TIRIs that involves weighted averaging, which is also included in the fingerprint extraction times shown in Table III. It can thus be seen that TIRI-DCT is more than 3 times faster than 3D-DCT for 2-S segments used throughout this paper.

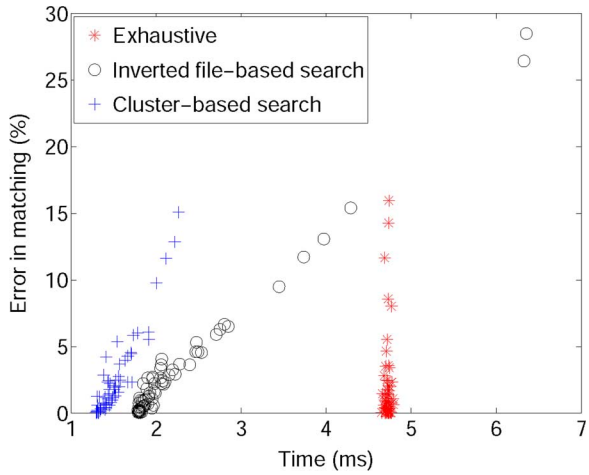

Fig. 15. Error versus search time for different search algorithms applied on fingerprints derived by TIRI-DCT.

B. Performance Evaluation of the Fingerprint Extraction Algorithm

As discussed earlier, an important property of a fingerprinting system is its ability to detect and/or reject a query video within a large database in a fast and reliable fashion. In this section, we evaluate the performance of the proposed inverted-file-based search and the cluster-based search algorithms and compare them with that of the exhaustive search method (as the reference) when applied to the fingerprints derived by TIRI-DCT. Fig. 15 shows the speed versus total error-rate (false positive rate + false negative rate) for all the three methods. Each point is the average over 1800 query fingerprints extracted from the attacked version of the videos in the database searched in a database of the same size. In the simulations, the words are 4 bits for the inverted-file-based search, and the cluster heads are 6 bits for the cluster-based search. It can be seen that both approximate search algorithms are much faster than the exhaustive search when error rates are low (the desired operation point of the system). It can also be seen that the proposed cluster-based approach is faster than the inverted-file-based search.

Table IV presents the average performance of the proposed approximate similarity search algorithms on the queries used for generating Table II results. Table IV and Fig. 15 highlight the trade-off between the system's performance and its detection speed, where decreasing the detection time by about 4 times has reduced the F-score by about 1%. Table shows that both search methods have a performance close to the exhaustive search for noise addition, brightness and contrast change, and rotation. The

TABLE IV
COMPARISON OF THE PERFORMANCE OF EXHAUSTIVE SEARCH (EXH.), INVERTED-FILE-BASED SIMILARITY SEARCH (INV.), AND CLUSTER-BASED SIMILARITY SEARCH (CLU.)

Attack	TPR			FPR			F-score		
	Exh.	Clu.	Inv.	Exh.	Clu.	Inv.	Exh.	Clu.	Inv.
Noise	99.24	98.62	98.19	0.66	1.21	1.63	0.99	0.99	0.98
Brightness	99.40	99.22	99.20	0.60	0.75	0.77	0.99	0.99	0.99
Contrast	99.02	98.83	98.81	0.75	0.78	0.80	0.99	0.99	0.99
Rotation	99.57	98.93	98.45	0.43	1.06	1.54	1.00	0.99	0.98
Time shift	95.8	93.46	93.08	1.73	2.12	2.49	0.98	0.97	0.96
Spatial shift	96.25	92.61	91.86	1.85	2.61	3.37	0.98	0.96	0.95
Frame drop	98.39	96.94	96.54	0.78	1.41	1.81	0.99	0.98	0.98
Average	98.24	96.94	96.59	0.97	1.42	1.77	0.99	0.98	0.98

TABLE V
PERFORMANCE OF THE SYSTEM AFTER ADDING THE POSTPROCESSING MODULE

Attack	TPR (%)		FPR (%)		F-score	
	Clu.	Inv.	Clu.	Inv.	Clu.	Inv.
Noise	99.12	98.98	0.80	0.94	0.99	0.99
Brightness	99.07	99.06	0.91	0.92	0.99	0.99
Contrast	98.61	98.61	0.98	0.99	0.99	0.99
Rotation	99.19	99.13	0.81	0.87	0.99	0.99
Time shift	95.00	94.88	1.19	1.32	0.98	0.98
Spatial shift	93.82	93.56	1.53	1.77	0.97	0.97
Frame loss	98.39	98.37	0.80	0.83	0.99	0.99
Average	97.60	97.50	1.00	1.09	0.99	0.99

performance of the search algorithms, however, decreases for attacks such as time shift or spatial shift.

To further increase the performance of the search algorithms, we have deployed a simple postprocessing approach. The postprocessing module exploits the existing temporal dependency between adjacent fingerprints to correct the detection results. It corrects the index of a video segment (current segment) by looking at its adjacent segments (here four segments before and four segments after the current segment). If more than 50% (here, five or more) of these segments have the same video index, the segment is assigned the same index. Using this postprocessing approach, we were able to boost the performance as shown in Table V.

It can be seen from Table V that the applied postprocessing has boosted the system's performance by both increasing the TPR and decreasing the FPR. The effect of the applied postprocessing step can specifically be seen for more severe attacks such as time shift and spatial shift, where the FPR is decreased to about half of its original value. Tables IV, V, and Fig. 15 show that the cluster-based search algorithm is faster than the inverted-file-based approach while it also has a better performance. We thus adopt the cluster-based algorithm as the search engine for our copy detection system. In Section V, we propose and evaluate a secure version of the proposed fingerprinting algorithm.

V. DISCUSSION AND ANALYSIS OF THE FINGERPRINTS SECURITY

We mentioned that security of a fingerprinting system is important for applications such as copyright protection where adversaries compromise the performance. In this section we propose a secure version of our fingerprinting algorithm, denoted as Secure TIRI-DCT. To have a secure algorithm, randomness is introduced to the algorithm via a set of two secret keys. Each key is a randomly selected vector of integers defining a certain procedure to be performed. We explained in Section II-B2, that TIRI-DCT algorithm generates fingerprints by segmenting the TIRI images into n overlapping blocks and extracting two feature (the first two non-DC DCT coefficients) from each block. In the Secure TIRI-DCT, one feature is extracted from each block which is selected from the set of first three non-DC DCT coefficients. The first key defines the coefficient to be selected from each block. The second key permutes all the selected coefficients and generates a randomly arranged feature vector. The feature vector is then binarized using the same method as TIRI-DCT. Assuming that TIRI is divided to L blocks in Secure TIRI-DCT, there would be $3^L \times L!$ different possibilities depending on the selected keys (as there are three choices for each block, 3^L is then multiplied by the number of permutations for L blocks). This randomization procedure results in a highly secure fingerprinting system, which makes it almost computationally impossible for an intruder to extract the same fingerprints from a video without knowing the keys.

To analytically evaluate the security of a fingerprinting system from the point of view of an adversary, Swaminathan *et al.* have proposed the use of entropy (or the differential entropy) of the generated fingerprints [27]. The entropy shows how hard it would be for an intruder who knows the algorithm but does not have any information about the secret keys to estimate the fingerprint. For our proposed secure TIRI algorithm, the entropy of the intermediate fingerprints (before binarization) is $L \log 3(\sum_{k=1}^{L} \log k)$, which is derived for the case when the keys are independently drawn from a uniform distribution. However, the entropy of the final fingerprint $N(h_L)$, after binarization, is much lower and is equal to $\log \binom{L/2}{L}$. This entropy can be found using a recursive formula

$$\mathcal{N}(h_L) = \mathcal{N}(h_{L-2}) + \log(L(L-1)) - 2\log(L/2). \quad (8)$$

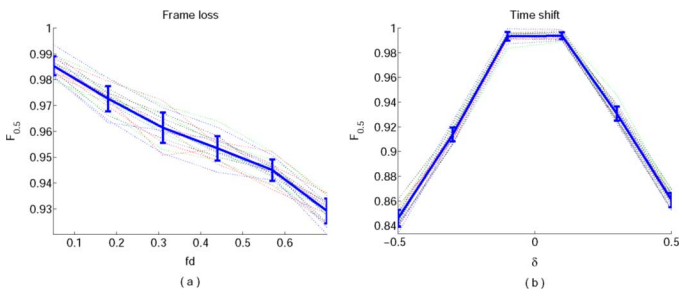

Fig. 16. F-score for different keys for different parameter values when the video is attacked by (a) frame loss and (b) time shift.

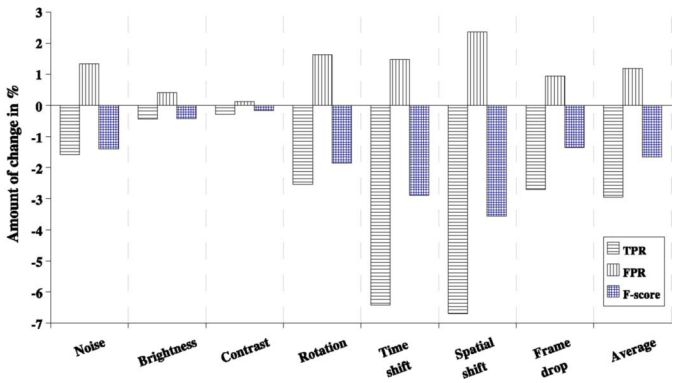

Fig. 17. Amount of change in the average TPR, FPR, and $F_{0.5}$ when the security is added to the system.

The entropy of the final stage fingerprints determines the actual amount of randomness in the generated fingerprints. This entropy is usually much lower than the entropy of the intermediate features studied in [27]. Last but not least, entropy is not a good measure, because it does not consider the prior knowledge of the adversary. In fact, it should be practically impossible for an intruder who knows the fingerprinting algorithm to estimate the fingerprints without knowing the secret key even though he has some sample fingerprints. For example, for Secure TIRI-DCT, the use of the two secret keys mentioned above decreases the amount of information revealed by the sample fingerprints about the keys. If only the permutation (first) key were used, the key could be found with a high probability, given enough sample fingerprints. When both keys are used, it is much harder for the intruder to guess them (based on the available samples) although the entropy does not change (by adding the second key). We now show that the performance of the system is independent of the selected keys. Fig. 16 shows the performance of Secure TIRI-DCT for frame losses of 5%–70% and time shifts of -0.5 to 0.5 s, when different keys are used (1000 keys were randomly selected). Each key is used both for generating the database as well as extracting the query fingerprints from the attacked videos.

Fig. 16 shows how little the F-score is affected by choosing different DCT coefficients (features) as the result of using the selection key. The bars on the plots show the variance of the F-score for different parameters and different keys. Similar observations apply to other attacks, which further demonstrates that Secure TIRI-DCT performance is not significantly affected by the selected keys.

Fig. 17 quantifies the trade-off between security and the system's performance, represented by TPR, FPR, and F-score. The bars in the chart show the amount of change in TPR, FPR, and the F-score. On average, TPR decreases by about 3%, FPR increases by about 1.2%, and F-score increases by about 1.7%. These results indicate that security is gained at the expense of a slight reduction in the system's performance. The gap in the performance can be decreased further by increasing the length of the fingerprint. The above results represent the performance of secure 64-bit fingerprints versus that of nonsecure 126-bit fingerprints. When comparing secure and nonsecure systems with fingerprints of lengths 320 and 640, we observe that, on average, the TPR of the secure system decreases by 1%, FPR increases by 0.3%, and F-score decreases by 0.5%. These results show that security can be achieved with a small detection performance loss using larger fingerprints length. It is, however, worth mentioning that larger fingerprints result in a decrease in detection speed as they require more computation in calculating the Hamming distances between the fingerprints.

VI. Conclusion

This paper proposes a fingerprinting system for video copy detection. It can be used for copyright management and indexing applications. To the best of our knowledge, this is the first paper to discuss robustness, discrimination, security, and fast search of fingerprints simultaneously. The system consists of a fingerprint extraction algorithm followed by an approximate search method. The proposed fingerprinting algorithm (TIRI-DCT) extracts robust, discriminant, and compact fingerprints from videos in a fast and reliable fashion. These fingerprints are extracted from TIRIs containing both spatial and temporal information about a video segment. We demonstrate that TIRI-DCT generally outperforms the well-established (3D-DCT) algorithm and maintains a good performance for different attacks on video signals, including noise addition, changes in brightness/contrast, rotation, spatial/temporal shift, and frame loss. It is shown experimentally that TIRI-DCT has a high average true positive rate of 98.2% and a low average false positive rate of 0.97%. We also propose two fast approximate search algorithms: the inverted-file-based method which is a generalization of an existing search method, and another method based on a novel clustering-based approach. Analytical and experimental studies demonstrates that both of the algorithms are very fast compared to an exhaustive search and maintain a good performance. The cluster-based method is experimentally shown to be superior to the inverted-file-based method in terms of the detection performance and the query retrieval time. We have thus adopted the cluster-based method as the search engine of our fingerprinting system. By applying a simple postprocessing method, the final system performance yields a high average TPR of 97.6% and a low average FPR of 1.0%.

As security of the fingerprinting system is important for some applications such as copyright protection, we also introduced a secure version of our algorithm, denoted as Secure TIRI-DCT. We showed that the use of this algorithm makes it extremely difficult for an adversary to tamper with the fingerprinting system. The increase in the security was, however, associated with a

slight decrease in the performance. Security is an area of research that still needs much attention. As part of our future work, we will conduct a detailed analytical study of the security of fingerprinting algorithms including the one proposed in this paper.

As another part of our future work, we will carry an extensive comparison study to compare our fingerprinting algorithms to other state-of-the-art algorithms. We will also evaluate our proposed fast search methods when applied to other fingerprinting methods. We also plan to study the performance of the system in the presence of some other attacks, such as cropping, and logo insertion. Another class of attacks, studied mainly in the video retrieval community, are content-changing attacks such as changing the background or picture in picture. Robustness against such attacks is not achievable via global fingerprints. Such attacks can only be handled using local fingerprints via interest point-based algorithms. Including such capabilities is beneficial for a complete copy-detection system. We also intend to improve the robustness of the global features to large geometric attacks, and further reduce the required search time, while maintaining acceptable performance.

ACKNOWLEDGMENT

The authors would like to thank C. Wilson and the anonymous reviewers for their valuable comments on the paper.

REFERENCES

[1] J. Oostveen, T. Kalker, and J. Haitsma, "Feature extraction and a database strategy for video fingerprinting," in *Proc. Int. Conf. Recent Advances in Visual Information Systems (VISUAL)*, London, U.K., 2002, pp. 117–128, Springer-Verlag.

[2] J. Lu, "Video fingerprinting for copy identification: From research to industry applications," E. J. Delp, J. Dittmann, N. D. Memon, and P. W. Wong, Eds., SPIE vol. 7254, no. 1, 2009, p. 725402 [Online]. Available: http://link.aip.org/link/?PSI/7254/725402/1

[3] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford, "Video copy detection: A comparative study," in *Proc. ACM Int. Conf. Image and Video Retrieval*, New York, NY, 2007, pp. 371–378, ACM.

[4] A. Hampapur and R. M. Bolle, Videogrep: Video copy detection using inverted file indices IBM Research Division Thomas. J. Watson Research Center, Tech. Rep., 2001.

[5] L. Chen and F. W. M. Stentiford, "Video sequence matching based on temporal ordinal measurement," *Pattern Recogn. Lett.*, vol. 29, no. 13, pp. 1824–1831, 2008.

[6] R. Radhakrishnan and C. Bauer, "Content-based video signatures based on projections of difference images," in *Proc. MMSP*, Oct. 2007, pp. 341–344.

[7] C. De Roover, C. De Vleeschouwer, F. Lefebvre, and B. Macq, "Robust video hashing based on radial projections of key frames," *IEEE Trans. Signal Process.*, vol. 53, no. 10, pp. 4020–4037, Oct. 2005.

[8] S. Lee and C. Yoo, "Robust video fingerprinting for content-based video identification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 983–988, Jul. 2008.

[9] X. Su, T. Huang, and W. Gao, "Robust video fingerprinting based on visual attention regions," in *Proc. ICASSP*, Washington, DC, 2009, pp. 1525–1528, IEEE Computer Society.

[10] A. Joly, O. Buisson, and C. Frelicot, "Content-based copy retrieval using distortion-based probabilistic similarity search," *IEEE Trans. Multimedia*, vol. 9, no. 2, pp. 293–306, Feb. 2007.

[11] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, Washington, DC, 2003, p. 1470, IEEE Computer Society.

[12] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. Eur. Conf. Computer Vision (ECCV)*, Berlin, Heidelberg, Germany, 2008, pp. 304–317, Springer-Verlag.

[13] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. Int. Conf. Very Large Data Bases (VLDB)*, San Francisco, CA, 1999, pp. 518–529, Morgan Kaufmann Publishers Inc..

[14] Z. Liu, B. Shahraray, and T. Liu, AT&T research at TRECVID 2009 content-based copy detection [Online]. Available: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009

[15] M. Hritier, V. Gupta, L. Gagnon, G. Boulianne, S. Foucher, and P. Cardinal, CRIMs content-based copy detection system for TRECVID [Online]. Available: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009

[16] A. Natsev, M. Hill, J. R. Smith, L. Xie, R. Yan, S. Bao, D. Wang, M. Merler, and Y. Zhang, IBM research TRECVID-2009 video retrieval system [Online]. Available: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009

[17] U. Cao, Y.-D. Zhang, B.-L. Feng, L. Bao, L. Pang, J.-T. Li, K. Gao, X. Wu, H.-T. Xie, W. Zhang, and Z.-D. Mao, TRECVID 2009 of MCGICT- CAS [Online]. Available: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009

[18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[19] Personal Communication With Prof. David Lowe Aug. 2010.

[20] B. Coskun, B. Sankur, and N. Memon, "Spatiotemporal transform based video hashing," *IEEE Trans. Multimedia*, vol. 8, no. 6, pp. 1190–1208, Dec. 2006.

[21] R. Radhakrishnan and C. Bauer, "Robust video fingerprints based on subspace embedding," in *Proc. ICASSP*, Apr. 2008, pp. 2245–2248.

[22] G. Willems, T. Tuytelaars, and L. Van Gool, "Spatio-temporal features for robust content-based video copy detection," in *Proc. ACM Int. Conf. Multimedia Information Retrieval*, New York, NY, 2008, pp. 283–290, ACM.

[23] A. Joly, O. Buisson, and C. Frelicot, "Content-based copy retrieval using distortion-based probabilistic similarity search," *IEEE Trans. Multimedia*, vol. 9, no. 2, pp. 293–306, Feb. 2007.

[24] C. Kim and B. Vasudev, "Spatiotemporal sequence matching for efficient video copy detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 1, pp. 127–132, Jan. 2005.

[25] M. Malekesmaeili, M. Fatourechi, and R. K. Ward, "Video copy detection using temporally informative representative images," in *Proc. Int. Conf. Machine Learning and Applications*, Dec. 2009, pp. 69–74.

[26] M. Malekesmaeili and R. K. Ward, "Robust video hashing based on temporally informative representative images," in *Proc. IEEE Int. Conf. Consumer Electronics*, Jan. 2010, pp. 179–180.

[27] A. Swaminathan, Y. Mao, and M. Wu, "Robust and secure image hashing," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 2, pp. 215–230, Jun. 2006.

[28] A. L. Varna and M. Wu, "Modeling and analysis of content identification," in *Proc. ICME*, 2009, pp. 1528–1531.

[29] B. Coskun and N. Memon, "Confusion/diffusion capabilities of some robust hash functions," in *Proc. Conf. Information Sciences and Systems (CISS)*, 2006, pp. 1188–1193.

[30] Video Traces Research Group Arizona State University [Online]. Available: http://trace.eas.asu.edu/index.html

[31] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. Int. Conf. Computer Vision Theory and Applications*, 2009, pp. 331–340.

[32] J. A. Haitsma, A. A. C. M. Kalker, C. P. M. J. Baggen, and J. C. Oostveen, Generating and matching hashes of multimedia content US 2002/0178410, Nov. 2002.

[33] M. L. Miller, "Audio fingerprinting: Nearest neighbour search in high dimensional binary spaces," in *IEEE Workshop on 2002, Multimedia Signal Processing, 2002*, 2002, pp. 182–185.

[34] B. Coskun and N. Memon, "Online sketching of network flows for realtime stepping-stone detection," in *Proc. Annual Computer Security Applications Conf.*, Los Alamitos, CA, 2009, pp. 473–483, IEEE Computer Society.

[35] ReefVid: A Resource of Free Coral Reef Video Clips for Educational Use [Online]. Available: http://www.reefvid.org

[36] C. van Rijsbergen, *Information Retrieval*. London, U.K.: Butterworth, 1979.

Mani Malek Esmaeili received the B.Sc. and M.Sc. degrees in electrical and computer engineering from the University of Tehran, Iran. He is currently working toward the Ph.D. degree in the Electrical and Computer Engineering Department, University of British Columbia, Vancouver, Canada.

His research interests include signal processing, machine learning, game theory, and optimizations. He is working on scalable video and audio fingerprinting algorithms for online applications.

Mehrdad Fatourechi was born in Tehran, Iran, in 1976. He received the M.Sc. degree from the University of Tehran in 2001 (*summa cum laude*) and the Ph.D. degree from the University of British Columbia, Vancouver, Canada, in 2008.

He is currently a research associate in the Department of Electrical and Computer Engineering, University of British Columbia. His research interests include image and video processing, pattern recognition, machine learning, statistical signal processing, and digital signal processing.

Rabab Kreidieh Ward (S'71–M'72–SM'85–F'99) is a Professor in the Electrical and Computer Engineering Department, University of British Columbia, Vancouver, BC, Canada. She is presently with the Office of the Vice President Research and International as the natural sciences and engineering research coordinator. Her research interests are in signal, image, and video processing. She has made contributions in the areas of signal detection, image encoding, compression, recognition, restoration and enhancement, and their applications to infant cry signals, cable TV, HDTV, medical images, and astronomical images. She has published over 380 journal and conference papers and holds six patents related to cable television picture monitoring, measurement, and noise reduction. Applications of her work have been transferred to U.S. and Canadian industries.

She was a Vice President of the IEEE Signal Processing Society, 2003–2005, the General Chair of the IEEE ICIP 2000, the IEEE ISSPIT 2006, and the Vice Chair of the IEEE International Symposium on Circuits and Systems 2004. She is a Fellow of the Royal Society of Canada, Canadian Academy of Engineers and Engineering Institute of Canada, and a recipient of the UBC 1997 Killam Research Prize, YWCA Woman of Distinction Award (2008), the R. A. McLachlan Memorial Award (the top award) of the Association of Professional Engineers and Geoscientists of British Columbia, and the "Society Award" of the IEEE Signal Processing Society.