

# Spatio–Temporal Transform Based Video Hashing

Baris Coskun, *Student Member, IEEE*, Bulent Sankur, *Senior Member, IEEE*, and Nasir Memon, *Member, IEEE*

**Abstract**—Identification and verification of a video clip via its fingerprint find applications in video browsing, database search and security. For this purpose, the video sequence must be collapsed into a short fingerprint using a robust hash function based on signal processing operations. We propose two robust hash algorithms for video based both on the Discrete Cosine Transform (DCT), one on the classical basis set and the other on a novel randomized basis set (RBT). The robustness and randomness properties of the proposed hash functions are investigated in detail. It is found that these hash functions are resistant to signal processing and transmission impairments, and therefore can be instrumental in building database search, broadcast monitoring and watermarking applications for video. The DCT hash is more robust, but lacks security aspect, as it is easy to find different video clips with the same hash value. The RBT based hash, being secret key based, does not allow this and is more secure at the cost of a slight loss in the receiver operating curves.

**Index Terms**—Broadcast monitoring, multimedia content authentication, robust hash, video database indexing, video hash.

## I. INTRODUCTION

**H**ASH functions are widely used in cryptography (MD5, SHA-1), where the main purpose is to check the integrity of the data. Since the resulting hash value is highly sensitive to every single bit of the input, these functions are extremely fragile and cannot be adopted for hashing multimedia data. In multimedia hashing, it is more important to be sensitive to the content rather than the exact binary representation. For instance, the raw video, its compressed version, its low-pass filtered version, its increased brightness version and its decreased contrast version should yield the same hash value since their content is essentially the same but their binary forms are very different. So, an alternate way to compute the hash is needed for multimedia applications, where the hash function results in the same output value unless the underlying content is significantly changed. Such a hash function is known as a robust or perceptual hash function. Some of the applications of perceptual video hashing include the following: 1) automatic video clip identification in a video database or in broadcasting, 2) online search in a streaming video, 3) authentication of the video content, 4) content-based watermarking. The two desired properties of hash

functions for multimedia data are robustness and uniqueness. Robustness implies that the hash function should be insensitive to perturbations, nonmalicious modifications caused by “mild” signal processing operations that in total do not change the content of the video sequence. These modifications can be enacted by the user, such as MPEG compression, contrast enhancement or can occur during storage and transmission functions, such as transcoding or packet drops. The uniqueness property implies that the hash functions are statistically independent for different content, so that any two distinct video clips result in different and apparently random hash values.

There have been many robust hash functions proposed in the literature. Fridrich [1] addresses the tamper control problem of still images by projecting image blocks onto key based random patterns and thresholding to create a robust hash. Venkatesan [2] computes an image hash for indexing and database searching from the statistics of subband wavelet coefficients. Lefèbvre [3] uses the Radon transform for a perceptual hash. Similarly Seo *et al.* [4] have used the Radon transform to obtain an affine-distortion resilient image fingerprint. Mihcak [5] iteratively selects the visually significant objects from wavelet coefficients. Caspi and Bargeron [6] obtain per frame hash by constructing a Gaussian pyramid that yields a low-resolution image, which is then median thresholded, resulting in a binary pattern for each video frame. Yang *et al.* [7] base their video identification on frame-by-frame similarity measure. They combat variations in bit rate, frame rate, compression codec and resolution via locally sensitive hashing and a nearest-neighbor search technique. In [8], Monga exploits non-negativeness of pixel values and computes hash via non-negative matrix factorization.

Although these still-image hashing methods can be extended to video signals on a frame-by-frame basis, it is our contention that a perceptual hash that encompasses the spatiotemporal content of the video in its totality would be more effective. For example, video hashes constructed from the concatenation of individual frame hashes would be very vulnerable against temporal desynchronizations, such as frame rate change or frame dropping. On the other hand, key-frame based hashes would be weak from a security point of view, since an attacker could doctor the remaining frames and obtain quite a different video sequence, and yet end up in the same hash since key frames are kept untouched.

There are relatively few algorithms in the literature that do not disregard the temporal evolution of content information. Oostveen *et al.* [9], in a seminal paper, obtain a video hash by applying  $2 \times 2$  spatiotemporal Haar filters on the randomized block means of the luminance component. In a similar vein, in view of the shortcomings of frame-by-frame hashing, we propose a technique that jointly extracts temporal features along with the spatial features. Our scheme constitutes an extension of the existing hashing technique, DCT-transformation

Manuscript received March 11, 2005; revised February 16, 2006. This work was supported by Bogazici University Research Fund Project 03A203. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jie Yang.

B. Coskun is with Electrical and Computer Engineering Department, Polytechnic University, Brooklyn, NY 11201 USA (e-mail: bcosku01@utopia.poly.edu).

B. Sankur is with Electrical Engineering Department, Bogazici University, Istanbul, Turkey (e-mail: bulent.sankur@boun.edu.tr).

N. Memon is with Computer and Information Science Department, Polytechnic University, Brooklyn, NY 11201 USA (e-mail: memon@poly.edu).

Digital Object Identifier 10.1109/TMM.2006.884614

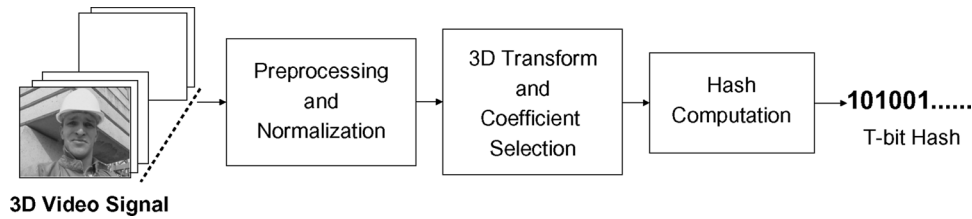


Fig. 1. Block diagram of the robust video hashing method.

followed by median quantization as in [10] to spatio-temporal data. The innovation with respect to existing video hashing methods in the literature is that it makes use of both spatial and temporal video information simultaneously via 3-D transformations. This increases robustness against possible temporal modifications, such as frame drop or frame rate change, while maintaining the robustness against spatial manipulations.

There is some analogy between robust video hashing and content-based video retrieval. However their requirements do not overlap completely. There are many proposed techniques in the literature for video retrieval based upon such features such as color histogram, texture and motion vectors for content identification [11], [12]. In content-based video retrieval, contents similar in terms of color, texture, motion distribution etc. generate similar fingerprints, which are used for database search. In robust hashing, on the other hand, authentication and security aspects play a more important role. For example, we want to prevent an attacker to easily switch between similar contents without disturbing the hash. A case in point would be two different newsreaders dressed similarly and speaking in front of the same background. While content-based video retrieval would generate the same fingerprint, the robust hash should differentiate the two newsreader scenes.

The rest of the paper is organized as follows. In Section II, the proposed hash technique is introduced along with the two different transforms. In Section III, useful properties and statistical characteristics of the resulting hash sequences are presented. Section IV contains detailed information about the experimental setup with discussions on the tools and techniques used in the experiments. The results of the experiments are detailed in Section V, where we address not only identification and verification performance under adversarial conditions but also we present broadcast monitoring results and performance comparisons with Oostveen’s algorithm [9]. Finally in Section VI, conclusions are drawn and future studies are explored.

## II. ROBUST HASH COMPUTATION

The proposed hash function is based on the intrinsic attributes of a video that capture its spatio-temporal essence. These attributes are based on the low-pass coefficients of 3-D transformations of the luminance component of a video sequence. The transforms we considered are the 3-D DCT (3D-DCT) and the 3-D 1 Random Bases Transform (3D-RBT), although several other transforms such as Discrete Wavelet Transforms, etc., would also be possible. Due to their low-pass nature, these features are insensitive to minor spatial and/or temporal perturbations. However, since the predominant portion of the energy resides in these coefficients, they possess sufficient discriminative

information about a video sequence. The final hash string is generated from the relative magnitude relationship among selected coefficients.

The video clip to be hashed can exist in various spatial dimensions and frame rates. Since our scheme targets a constant number of hash bits, we need to standardize the input video clips in order to keep the size of the hash sequence constant for all eventual inputs. This means that all the resulting hash values have equal significance and represent their ancestor video clips at equal degree of summarization. One can think of hashing as a clustering operation, where all the video sequences, corresponding to essentially the same content but that may have different representations, are all mapped to the same hash sequences. The first step in clustering is the preprocessing operation, where videos under different formats, frame rates and sizes, and sequence lengths are standardized to a reference size and length. The preprocessed video, albeit with the same content, can still have an infinite variety of appearances due to innocent or malicious modifications, which range from MPEG compression to frame drops in the transmission. Some 15 varieties of modifications are in fact described in Sections IV and V. Our hashing scheme then purports to map these sequences into the same hash value if the contents are identical, and to entirely different hash values if the contents are different.

Fig. 1 shows the basic steps of our robust hashing algorithm. The input video is first normalized to a standard format. After that, a 3-D transformation is applied on this standardized video and specific transform coefficients are selected. Finally, a hash is generated from the magnitude relationship among the selected coefficients. In the rest of this section, we explain each of the hash computation steps.

### A. Preprocessing and Normalization

The input video sequence is first converted to a standard video signal in terms of frame dimensions and of the number of frames via smoothing and subsampling. Let’s adopt a notation  $Video(w, h, f)$ , which represents some video clip with title “Video”, and where “ $w$ ” is the frame width, “ $h$ ” is the frame height and “ $f$ ” is the number of frames within the clip. An example is  $Foreman(176, 144, 400)$ , which is in the QCIF size and it contains 400 frames. In our scheme, any video signal,  $V_{original}(w, h, f)$ , is converted to a standard size,  $V_{norm}(W, H, F) = V_{norm}(32, 32, 64)$  via both spatial and temporal smoothing and subsampling. The hash function is then calculated using this standardized  $Video(32, 32, 64)$  sequence. This “standard” size was experimentally determined based upon the fact that smaller sizes risk losing semantic

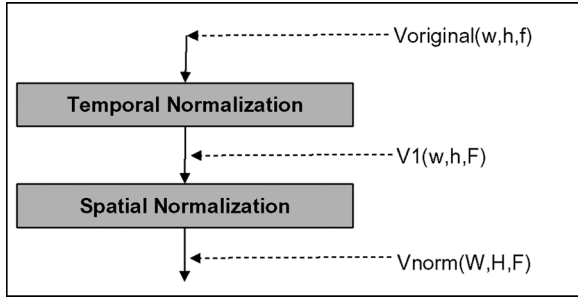


Fig. 2. Stages of preprocessing before hash extraction.

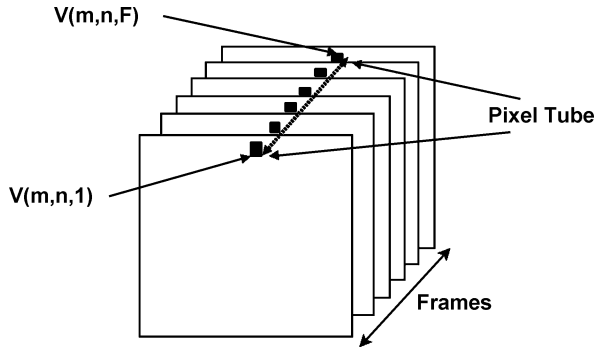


Fig. 3. Description of a pixel tube of length  $f_{total}$  and at location  $(m, n)$ .

content, hence discriminatory power, while larger dimensions have diminishing returns for uniqueness.

Fig. 2 shows how the video dimensions change in stages. The original video signal  $V_{original}(w, h, f)$  with arbitrary dimensions is temporally smoothed and subsampled to form the  $V_1(w, h, F)$  sequence. Each pixel in the video frame is filtered separately along the temporal dimension as illustrated in Fig. 3. Here, the array of pixels having the same location in successive frames is referred as a pixel tube and the  $(m, n)^{th}$  pixel tube is defined as  $\{V(m, n, f); f = 1, \dots, f_{total}\}$ , where  $f_{total}$  is the total original number of frames, and there is one such tube for each of the pixels in the frame. Temporal smoothing removes minute variations of a pixel in time and spreads large changes or object movements over several frames. Finally, the frames are spatially smoothed and subsampled to yield the goal sequence  $V_2(W, H, F)$ . All pixel tubes are filtered with a low-pass Gaussian filter with variance of  $\sigma^2 = 6$ . This kernel size was adjusted in order not to excessively smooth out the dynamic content on the one hand, and on the other hand to allow for subsampling. Obviously, with too large a kernel, all video starts looking like a single blurred image, and with too short a kernel, high-frequency motion, not essential to typify the content, will unnecessarily impact on the computed hash and its robustness. An alternative way of filtering would be a motion-compensated spatio-temporal smoothing method, where the pixel tube trajectories are not straight, but follow the object motion. Since we gained a satisfactory result from separate spatial and temporal smoothing, we did not use the computationally harder motion-compensated methods. The smoothed video signal is afterwards subsampled in time, to reduce the input clip to the target number of frames “ $F$ ”. We denote this video signal as smoothed  $V_1(m, n, F)$ .

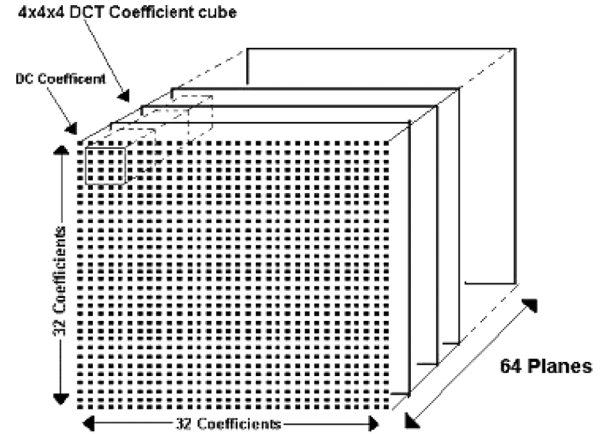


Fig. 4. The 3D-DCT array of normalized video with dimensions  $W = 32$ ,  $H = 32$  and  $F = 64$ . The cube of selected coefficients for 64-bit signature sequence is also shown in the upper-left corner.

Spatial smoothing was implemented on each frame via a 2-D Gaussian filter with variance  $\sigma^2 = 5$  and kernel  $h(m, n) = \exp(-(m^2 + n^2)/10)/\sqrt{10\pi}$ . Thus the temporally subsampled frames were convolved with the Gaussian filter  $h(m, n)$ , to yield  $V_2(m, n, F) = V_1(m, n, F) * h(m, n)$  and then subsampled to the size  $W \times H$ .

### B. 3D-Transforms and Coefficient Selection

Most 3D-transform techniques with good compacting characteristics can serve the purpose of summarizing and capturing the video content as they collect and embed in their coefficients the information concurrently from time and space. We focused on two transform types: the 3D-DCT transform due to its widespread usage in image and video processing and its easy implementation, and the 3D-RBT transform due to its key-based security aspects and its good feature extraction capability.

1) *3D-DCT Transform Case*: After applying the DCT transform to the normalized sequence  $V_{norm}(V, H, F)$ , one obtains a 3-D array of DCT coefficients  $V_{DCT}(V, H, F)$ . Typically low-frequency DCT coefficients contain the predominant part of the energy and they are also robust against most of the signal processing attacks, such as filtering and compression. To satisfy the uniqueness or discrimination property, one must judiciously enroll coefficients from mid- to high-frequency regions. In our experiments, we have found that  $T = 64$  coefficients, extracted from a  $4 \times 4 \times 4$  cube in the low-pass band, were appropriate for hash extraction.

We exclude the lowest frequency coefficients in each direction, that is DCT coefficients with addresses  $V_{DCT}(0, \dots)$ ,  $V_{DCT}(\dots, 0, \dots)$ , and  $V_{DCT}(\dots, \dots, 0)$ , to enhance uniqueness among video shots that have similar, but not identical content. In our experiments we have observed that including the lowest frequency coefficients will reduce the mean hamming distance between hash sequences by 2% (for 64 bit hash under no attack case, mean value drops from 32.01 to 31.34), which means that hash values become slightly alike for different contents. This is probably due to the fact that the lowest frequency coefficients does not reside much discriminative information. Fig. 4 shows the  $4 \times 4 \times 4$  cube containing the selected coefficients for hashing.

2) *3D-RBT Transform Case*: A hash function is secure if it is impractical for an adversary to maintain the same hash while changing the underlying video content or to obtain a significantly different hash for essentially the same content. Unless a robust hash is made secure, a pirate can beat applications that utilize it such as in broadcast monitoring, database searching or authentication [13]. In other words, the pirate can manipulate a video clip to yield hash values that are very different than the one used for monitoring and tracking. This weakness can be remedied if the computation of the hash is tied to the knowledge of a secret key. In other words, one should not be able to compute the target hash from the video clip unless one possesses the key. If the pirate does not possess the key, he/she would not know how to manipulate and engineer a clip and its hash to a desired value.

A practical way that immediately emerges would be pseudo-randomly selecting a subset of 3-D DCT coefficients and compute hash from them. However, the weakness of this method is that, the universal coefficient set whose random subset is used in hash extraction is known to an attacker. Thus, he may forge the original video without disturbing the rank order of those coefficients. In other words, he may modify the forged video by slightly altering candidate DCT coefficients and making them to have the same rank order as the original video. Then, no matter which subset of the coefficients is used, the extracted hash would be exactly the same. Therefore, the calculation of the coefficients have to be made secret (key-based), which can be achieved by using projections on random 3-D basis functions. These basis functions can be generated in various ways, for example, by narrow-band filtering of random number sequences [10]. In our work, we opted for discrete cosine transform bases, but with randomly chosen frequencies. Thus, we expect to attain a degree of randomness sufficient to fool a pirate and yet to benefit from the robustness and uniqueness of the regular DCT. We call this transform the Random Bases Transform or RBT.

Three-dimensional random bases with dimensions of  $W \times H \times F$  are generated by using separate 1-D RBT bases in each dimension, that is, by using cosinusoidal signals with random frequencies generated with a key. In fact each 1-D base is an ordinary 1-D  $l$ -term DCT function with a randomly assigned frequency:

$$b_{RBT}[n] = \cos\left(\frac{\pi(2n+1)\theta}{2l}\right), \quad n = 1, 2, \dots, (l-1). \quad (1)$$

In this function, the frequency  $\theta$  is pseudo-randomly selected as  $\theta = (pr(b_u - b_l)) + b_l$ , where  $\{b_u, b_l\}$  is the frequency interval and  $pr$  is a uniform random number in  $[0, 1]$ . For example, based on the experience of 3D-DCT coefficient selections, we have chosen  $\{b_u = 4, b_l = 1\}$ . Also the pseudo-random quantity  $pr \in \{0, 0.1, 0.2, \dots, 0.9\}$  is discrete and assumes ten different values.

In order to generate a 3-D random bases with dimensions  $W \times H \times F$ , first a number of 1-D bases has to be generated. For  $W$  direction there has to be  $H \times F$  1-D bases with length  $l = W$ ; for  $H$  direction there has to be  $W \times F$  1-D bases with length  $l = H$ ; finally for  $F$  direction  $W \times H$  1-D bases with length  $l = F$  generated. For instance,  $32 \times 32 + 32 \times 64 + 32 \times 64 =$

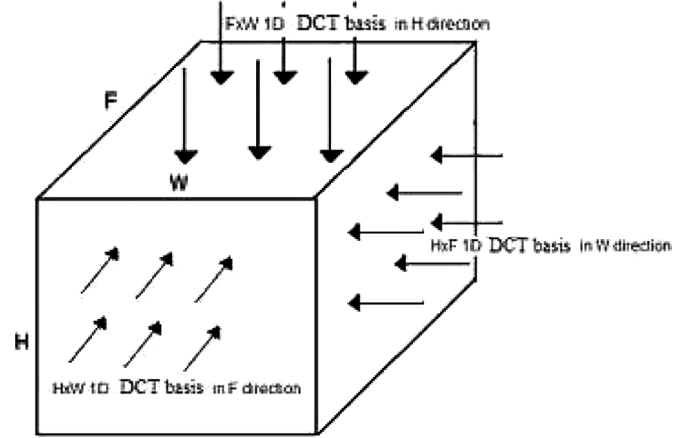


Fig. 5. Generation of a 3-D RBT basis by enrolling 1-D random-frequency DCT bases separately in each direction.

5120 1-D cosine signals are generated for a basis function of size  $32 \times 32 \times 64$ . Each of these 5120 signals has one of ten possible frequency values, which means that  $10^{5120}$  different basis functions can be generated. Therefore, an exhaustive search or brute force type attacks, which try to reveal the secret key, are not practical. These basis functions are illustrated in Fig. 5. Each arrow in this figure indicates the direction of 1-D cosine signal, that is, the data index set over which the projection operation is computed.

The hash features of a  $W \times H \times F$  video cube along each dimension are computed by a 3D-RBT basis set containing  $W \times H$ ,  $W \times F$  and  $H \times F$  random frequencies. By restricting the frequencies to lower bands one can inherit most of robustness and uniqueness properties of 3D-DCT bases. The overall 3D-RBT basis with dimensions  $W \times H \times F$  can be formulated as

$$B_{RBT}^i(w, h, f) = \beta \times \cos\left(\frac{\pi(2w+1)\theta_{(h,f)}}{2W}\right) \times \cos\left(\frac{\pi(2h+1)\theta_{(w,f)}}{2H}\right) \times \cos\left(\frac{\pi(2f+1)\theta_{(w,h)}}{2F}\right). \quad (2)$$

In this expression one has  $w = 1, \dots, W$ ;  $h = 1, \dots, H$ ;  $f = 1, \dots, F$ ,  $\beta$  is the normalization term, and  $\theta_{(h,f)}$  are pseudo-randomly generated frequencies to probe along the  $w$  dimension and similarly for the  $\theta_{(w,f)}$  and  $\theta_{(h,w)}$ .

In the DCT transform, the rows of the transform matrix follow a pattern of increasing frequencies; consequently, the columns of the matrix grow from low to high frequencies. On the other hand, in the RBT transform each row is assigned a random frequency, so that the juxtaposition of these rows may generate all high frequency patterns along the columns. Fig. 6(a) displays the 2-D signal consisting of random cosines and Fig. 6(b) one of its columns, where the high-frequency waveform is obvious. These high-frequency random basis functions unnecessarily reduce the robustness of the hash algorithm. To mitigate this loss of robustness, we low-pass filter the generated 3D-RBT bases in all three dimensions. Thus we average the random cosinusoids in the  $W \times H$  directions with a  $5 \times 5$  box filter, and those along the  $F$  direction with five-term box filter. Finally the RBT bases are normalized to have zero mean and unit length. The

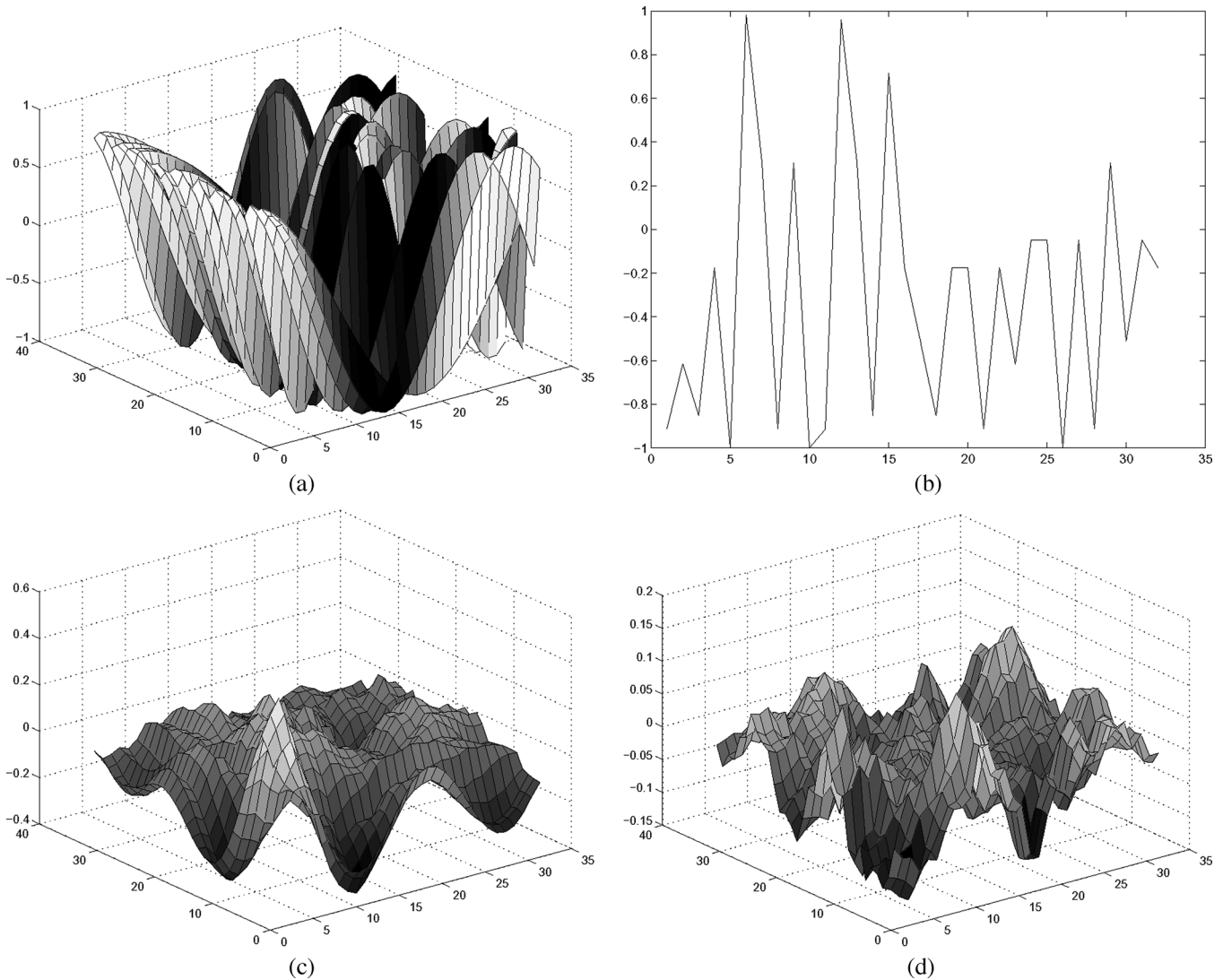


Fig. 6. (a) Display of a matrix whose rows consist of random sinusoids; (b) The 16th column of the 2-D signal in (a); (c, d) The first planes of 2 different 3-D RBT bases after low-pass filtering.

first planes of two different RBT bases after being filtered are shown in Fig. 6(c) and 6(d).

The RBT is obtained via the projection of the 3-D video data onto these 3-D basis functions. We have used dimensions similar to the 3D-DCT case, that is, using size (32, 32, 64) transform bases. The transformation is followed by the selection of the 64 components obtained from projection of signal onto 64 different basis functions.

### C. Hash Computation

Once the 3D-transform is applied and the specific coefficients are selected, the hash computation procedure is the same, regardless of which transformation is used. The selected  $T$  transform coefficients are binarized using the median of the rank-ordered coefficients. If the subset of rank-ordered coefficients is denoted as  $C_{(i)}$ ,  $i = 1, \dots, T$ , for some video sequence, then their median  $\mu$  is found as  $\mu = (C_{(T/2)} + C_{((T+1)/2)})/2$ . Once

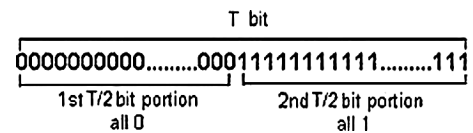


Fig. 7. Reference hash sequence used in hamming distance statistics.

$\mu$  is determined, then quantization of the selected coefficients of that video  $V$  is done as follows:

$$h_i = \begin{cases} 1 & C_{(i)} \geq \mu \\ 0 & C_{(i)} < \mu \end{cases}, \quad i = 1, \dots, T. \quad (3)$$

The quantization operation makes the hash more robust against minor changes in the video sequence, since we only preserve the information of the coefficient value being greater or smaller than the coefficient median. Furthermore, with this particular quantization we guarantee that there is always an equal number of 1's and 0's, with some interesting consequences as detailed in the next section.

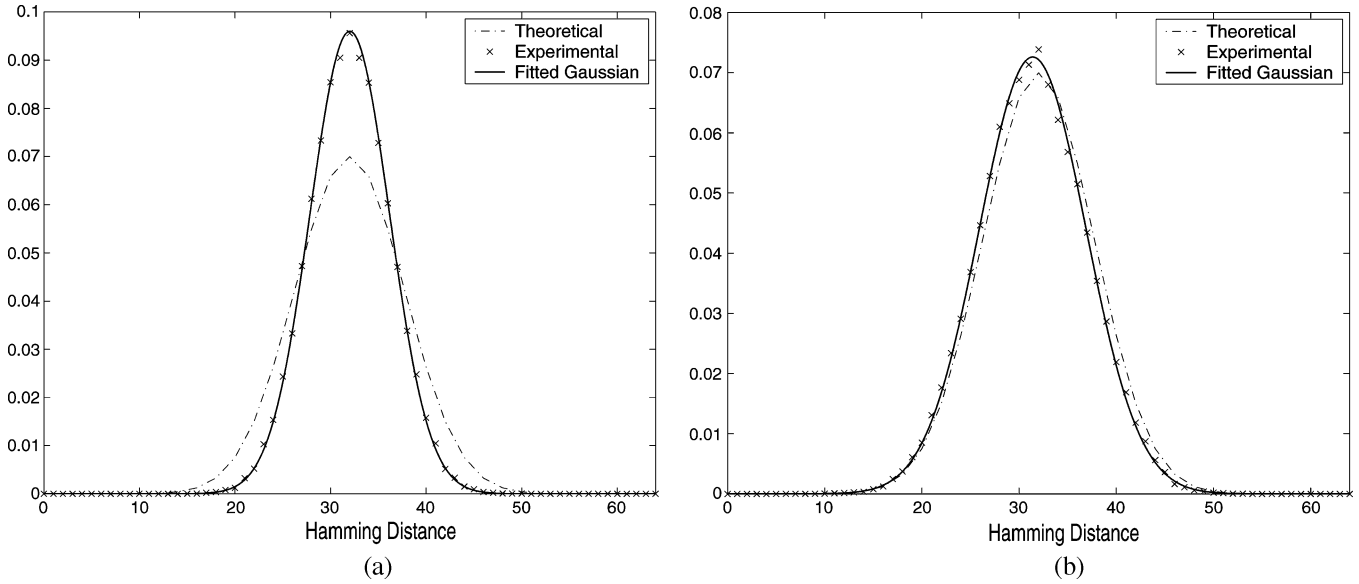


Fig. 8. Distribution of Hamming distances. Dashed curve is the theoretical probability density function [(7)] with  $T = 64$ ,  $\mu = 32$  and  $\sigma^2 = 32$ . The solid curve is the Gaussian density fitted to experimental data, where data points are marked with 'x'. a) DCT-based hash distances where  $\mu = 32.01$  and  $\sigma^2 = 17.25$ . b) RBT-based hash distances where  $\mu = 31.29$  and  $\sigma^2 = 30.33$ .

### III. PROPERTIES OF THE HASH SEQUENCE

Recall that the two critical properties of a perceptual hash function were uniqueness and robustness. Uniqueness implies that the hashes from any two different video contents should be radically different from each other with very high probability. In an ideal hash based on median quantization, we have assumed that each possible hash sequence occurs with equal probability guaranteeing the best uniqueness property. We test our hash functions whether obey this assumption by deriving theoretically the probability distribution of the distance between hashes arbitrarily selected videos and compare it against empirical densities via histograms.

The median-based quantization used in the generation of our robust hash function guarantees that there are exactly  $T/2$  1's and  $T/2$  0's in each hash sequence. The sequences possessing equal number of 1's and 0's are called "permissible sequences". In particular, the total number of possible hash values, denoted as  $N$ , can be calculated as, for  $T = 64$ :

$$N = \binom{T}{T/2} = \binom{64}{32} \approx 1.8326 \times 10^{18}. \quad (4)$$

Furthermore, we assume that all possible hash sequences occur with equal probability for the ideal case and then determine the distribution of hamming distances between any two arbitrary selected hashes. Under this condition, one would get the same probability density of hamming distances between any arbitrarily selected reference hash and all other remaining hashes. Thus, without loss of generality, we select a special hash sequence constituted of all zeroes in the first half and of all ones in the second half portion, as depicted in Fig. 7.

The hamming distance,  $\delta_0$ , between this special sequence and any other arbitrary hash sequence is determined by the number of differing digit positions, which is given by the number of 1's in the first half and the number of 0's in the second half of the  $T$ -bit sequence. More specifically, using the number of 1's and

0's in the  $k^{\text{th}}$  portion ( $k = 1$  or  $2$ ), respectively, as  $n1_k$  and  $n0_k$ , we would obviously have, for the reference sequence in Fig. 7,  $n0_2 = n1_1 = 0$  and  $n0_1 = n1_2 = T/2$ . For any other arbitrary and permissible sequence, we have the following relation:

$$\delta_0 = n1_1 + n0_2. \quad (5)$$

Also, the following equalities hold true: i)  $n1_1 + n1_2 = T/2$ , ii)  $n0_1 + n0_2 = T/2$ , iii)  $n1_1 + n0_1 = T/2$ , and iv)  $n1_2 + n0_2 = T/2$ . Using these equalities, we obtain

$$\delta_0 = n1_1 + T/2 - n0_1 = 2 \times n1_1. \quad (6)$$

Equation (6) states that the hamming distance between two arbitrary hash values is always an even number, since a differing 1 must always be compensated by a differing 0 in some other position to maintain equality of ones and zeros. Furthermore, the probability of a hamming distance  $\delta_0 = 2 \times n1_1$  is equal to the occurrence probability of  $n1_1$  ones in the first portion of the hash. Since ones and zeros occur with equal probability, the probability distribution of hamming distances is given by

$$\begin{aligned} P(\delta_0) &= P(n1_1) = \binom{T/2}{n1_1} \times 0.5^{n1_1} \times 0.5^{(T/2-n1_1)} \\ &= \binom{T/2}{n1_1} \times 0.5^{T/2} \end{aligned} \quad (7)$$

where one must have  $0 \leq n1_1 \leq T/2$  and  $\delta_0 = 2 \times n1_1$ . This probability density of hamming distances is plotted in Fig. 8. Since  $P(n1_1)$  is the binomial probability function, we have the following mean and variance values:  $E\{\delta_0\} = 2 \times E\{n1_1\} = T/2$  and  $\sigma_{\delta_0}^2 = 4 \times \sigma_{n1_1}^2 = T/2$ .

In Fig. 8, we have superposed the experimental distribution of hamming distances on the theoretical distribution in (7) under uniform distribution assumption of hashes. The experimental distribution is obtained by calculating hamming distances between 244 test video clips, that is using  $244 \times 243/2$  distance

computations overall. Also in Fig. 8, we plotted the Gaussian fitting to the observed hamming distances. We used the Expectation-Maximization (EM) fitting procedure, which also returns a figure of merit for being truly Gaussian. Also the experimental hamming distances were subjected to the Bera–Jarque gaussianness test. For the DCT-based hash, it was found that this test accepts the hypothesis of normal distribution up to significance level of 0.41. However, it is observed that the variance of the experimental distribution is lower than the theoretical distribution. This result may be primarily due to the well-structured nature of the DCT basis functions unlike the irregular and random structure of RBT basis functions. Hence, although any type of signal can be represented by DCT coefficients, the DCT coefficients of natural video signals may have some regularity, which can prevent the DCT coefficients appear as randomly picked numbers. Therefore, the resulting hashes of video clips cannot span the entire hash space uniformly. Consequently, the empirical and theoretical results does not match for DCT based hash. The goodness of fit of the Gaussian approximation to the binomial model was found to be 0.1026 using the Kolmogorov–Smirnov test.

On the other hand, for the RBT-based hash, the mean and variance of experimental distances are found as  $\mu = 31.29$  and  $\sigma^2 = 30.33$ , both very close to their theoretical values. One can observe that the solid Gaussian curve fits almost perfectly to the theoretical curve. The Bera–Jarque gaussianness test accepts the normality hypothesis with a significance level of 0.51 and the Kolmogorov–Smirnov distance between the empirical and theoretical distributions is 0.0886. The better fit between empirical distribution of hamming distances and the theoretical distribution can be due to the randomness and irregularity of the RBT patterns which result in transform coefficients and hash bits appearing as if they were randomly picked. However, since there has to be exactly 32 ones and 32 zeros in the hash function, the hash bits inherently cannot be independent from each other. For instance if first bit is 1, the probability of second bit to be 1 is  $31/63$  where the probability to be 0 is  $32/64$ . But as long as the 0's and 1's are uniformly spread through 64 positions, which we mostly observe in our experiments, the bits can be considered as almost independent. This can possibly explain the reason why the ideal and the empirical distributions differ slightly.

Another desirable property of the key-generated RBT hash is that it should be very difficult to obtain the same hash under different keys, in other words, the hashes for different keys should be totally unpredictable. If the hash values are statistically independent, then their distances should have a Gaussian distribution around the mean value of  $T/2 = 32$ , which indeed turns out to be the case. Under 100 different key values, the mean value of inter hamming distances is observed as 32.01. Also we investigated the independence of each bit in the hash. Again under 100 different key values, we calculated the marginal and conditional probabilities of each bit. We observed that each bit of the hash is almost statistically independent of other bits  $P\{h_1 = 1\} \cong P\{h_1 = 1/h_n = 1\} \cong 0.5$  where  $h_n$  represents bit locations in the hash and  $n = 1, 2, \dots, 64$ . Thus, the hash values of the same video for different keys can be regarded as statistically independent as required from a key based hash function.

A remark on the security of the RBT-based hashes: Radhakrishnan *et al.* in [13] propose a boosting technique to attack random-basis hashing algorithms. They show that, with a given image and its hash bits, the statistical characteristics of the random-bases can be learned with Adaboosting and the resulting hash bits can be eventually estimated for any given input image. The RBT-based video hash is immune to the boosting attack. The reason is that, different from vulnerable techniques pointed out in [13], each bit of our hash generated from median quantization of projections depends on the entire video sequence instead of on a single small block. More precisely, in the algorithms targeted by [13], the image (or frame) is first divided into blocks and then each block or its DCT coefficients are projected onto a random basis. The corresponding hash bit is obtained from the 1 bit quantization of that projection result with respect to a given threshold. Hence, once the attacker estimates the behavior of random bases, he could replace any block with another block without changing the hash bit. In order to make adaboosting attack impractical, it is proposed in [13] that the computed hash bits should depend also on the neighboring blocks instead of just one block. In this respect, our algorithm provides the neighborhood dependence par excellence, since each bit results from the projection of the entire video on some basis function. In other words, each bit depends on all the possible blocks into which the video can be partitioned. Moreover, unlike the vulnerable algorithms, in our method the threshold value for coefficient binarization is not predefined but calculated dynamically (median value of all projections). Thus the adaboosting attack is hampered both by dynamic thresholding and holistic (nonlocal) processing of data.

#### IV. EXPERIMENTAL SETUP

##### A. Types of Video Sequences

The video clips used in our tests were selected from four different genres of content classes widely available in video media. Our samples were extracted from TRT (Turkish Radio and Television) programs stored in high-quality MPEG-2 format. The content classes were as follows.

- *Movie*: Video from this class contains several scenes from the movie called *Soccer Dog*.
- *Documentary*: This class contains a documentary program showing the life of the Panda.
- *News*: The videos in this class consist of a speaker reading news, sports news and weather forecast refurbished with complementary scenes.
- *Sports*: This class contains scenes from a soccer game including both close and distance shots.

We selected 61 video clips from each of the content classes regardless of the scene changes or any border descriptor. So each clip may either contain a complete scene or concatenation of series of scenes. Each clip contained 350 frames, which corresponds to  $350/25 = 14$  s of test video.

##### B. Distance Metrics

In order to compare hash values, a specific distance metric has to be defined. A preferable distance metric should generate close

distances between hash values of video clips having same content and large distances between hash values of different contents.

Several proximity indices for binary vectors, such as the Jaccard index or simple matching coefficient, find applications in binary pattern comparisons [14]. We have simply used the Hamming distance between two hash values, labeled  $a$  and  $b$ , formulated as  $\delta = (1/2) \sum_{i=1}^T [1 - (h_i^a \times h_i^b)]$ , where the respective hash sequences  $\{h_i^a\}_{i=1}^T$  and  $\{h_i^b\}_{i=1}^T$  take  $-1$  and  $1$  values. We also wanted to correlate the Hamming distances with the quality differences between two video sequences, especially after the attacks, such as compression or frame drops. One possible way to measure the quality difference between two sequences (say, between original and attacked versions) is the Structural Similarity Index (SSIM) of Bovik [15], [16]. SSIM looks beyond simple pixel similarity, for a higher-level interpretation of distortion and thus attempts to provide an objective measurement of subjective quality assessment. SSIM figures range from 0 to 1, and as the two videos become more and more similar the score tends to 1. We stretched the use of the SSIM index beyond quality measurement as an indicator of content difference.

We also extracted two performance statistics from hashes using the 244 video clips: Inter-hash distances and intra-hash distances.

- *Intra-Hash Statistics* are based on the Hamming distances of hash functions of the same video content under various modifications. They represent the robustness performance of video hash function. In fact, we expect the Hamming distance to remain small between the hash of the original and the hashes of the video subjected to the modifications as in Table I.
- *Inter-Hash Statistics* are based on the Hamming distances of hash functions of different video sequences, whether in original form or in modified forms. They represent the uniqueness, that is, discrimination performance of the video hash algorithm. For any modification type or strength, we compute  $244 \times 243/2 = 29646$  Hamming scores, when we compare every clip to every other clip. If the contents are different, one expects the inter-distances to be clustered around the maximum average distance of  $T/2$ .
- *Intra-Hash Statistics with a Key* are based on Hamming distances of hash functions of the same video content under various keys. They represent the unpredictability aspects of the RBT video hash. We expect these statistics to behave as the inter-hash statistics case.

### C. Description of Attacks Types Used in the Experiments

The robustness characteristic of video hash is tested by using several different versions of the video clips obtained under various types of modifications. The types and properties of applied modifications, whether originating from signal processing operations or from losses during transmission, are listed in Table I. These form the major types of modifications on video. Also, the class of modifications can be further extended, two examples being histogram equalization and gamma correction. However, these modifications form a subset of brightness and contrast manipulations. Samples from modified video frames and

the concomitant SSIM and PSNR figures are also presented. The modification strengths used are exaggerated in order to apply “stress testing” to our video clip identifier. In other words, as evident from PSNR figures and sample frames, video quality falls often below acceptable level and yet, we expect the hash to identify and differentiate content correctly. In other words, despite severity of modifications, the hashes of the original and modified versions of the video must be close to each other.

Video signals are most likely to encounter losses during streaming over narrowband wireless channels. For a realistic measurement of the performance under such lossy channels, we have an experimental setup as illustrated in Fig. 9. The Xvid MPEG4 codec, used in lossy channel experiments, pastes the macroblocks from previous frames in lieu of lost macroblocks.

## V. EXPERIMENTAL RESULTS

In this section we report first the intra-hash and inter-hash distance statistics, and the consequent identification and verification performances. We next investigate the effects of coefficient selection patterns, sampling rate change, reverse play and discuss a broadcast monitoring application.


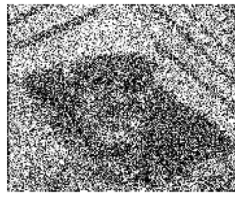










### A. Intra- and Inter-Hash Statistics

Table II and Fig. 12 give comparatively the robustness performance of the DCT-based and the RBT-based hashes. Recall that low deviation scores of intra- Hamming distances indicate robustness of the hash. It can be observed that the mean intra-Hamming values of RBT-based hashes are slightly higher than those of DCT-based hashes. This may be due to the more irregular patterns of RBT bases, which accommodate slightly higher frequency components, and hence are more sensitive to perturbations or attacks. Some comments on these results are in order.

- 1) *Blurring*: Since even heavy blurring does not much affect the low-frequency coefficients, the hash function remains very robust. Interestingly, for scenes with plain background and for almost static video sequences relatively higher Hamming distances result due to the fact most of the coefficients are close to zero, hence susceptible to sign changes with small perturbations.
- 2) *AWGN*: The high-frequency perturbation superposed on the video virtually goes unnoticed by the hash function.
- 3) *Contrast manipulation*: This manipulation modifies the range of the pixel values but without changing their mutual dynamic relationship. However, extreme contrast increase/decrease results in pixel saturation to 255 and clipping to 0, which forms low frequency plain regions and consequently distorts the hash outcome.
- 4) *Brightness manipulation*: Though the hash function is quite robust to this modification, whenever the brightness manipulation is taken to the extreme of saturation (too dark, clipped to 0 or too bright, saturated to 255), the hash function suffers. This is because the saturated pixels form uniform regions, which in turn distort the low-pass coefficients. Notice however this level of manipulation is not very realistic, as the video has lost most of its value.
- 5) *MPEG-4 compression*: Compression basically removes the high-frequency irrelevancy and so has very little effect on



TABLE I  
DESCRIPTION OF MODIFICATIONS, THEIR PARAMETERS AND ILLUSTRATIVE FRAMES

Modif. Type	Description and Param.	Sample Frame	Modif. Type	Description and Param.	Sample Frame
Blurring	Gaussian filtering of each frame, with size $\sigma_f$ of Gaussian filter kernel. Parameters: $\sigma_f^2 : 55$ , $SSIM : 0.51$ , $PSNR : 20.81$ .		AWGN	Additive white Gaussian noise with standard deviation $\sigma_n$ . Parameters: $\sigma_n : 110$ , $SSIM : 0.07$ , $PSNR : 9.72$ .	
Brightness Increase	Brightness increased by adding $r$ percent of the frame mean luminance value to each pixel: $(r \times m_{frame})$ . Parameters: $r : 80\%$ , $SSIM : 0.75$ , $PSNR : 9.91$ .		Brightness Decrease	Brightness decreased by subtracting $r$ percent of the frame mean luminance value to each pixel: Parameters: $r : 80\%$ , $SSIM : 0.38$ , $PSNR : 9.59$ .	
Contrast Increase	Linearly mapping luminance values in the interval $[v_l, v_h]$ to the interval $[0, 255]$ . The luminance values below $v_l$ and above $v_h$ are saturated to 0 and 255 respectively. Parameters: $v_h : 168$ , $v_l : 88$ , $SSIM : 0.48$ , $PSNR : 13.09$ .		Contrast Decrease	Linearly mapping luminance values in the interval $[0, 255]$ to the interval $[v_l, v_h]$ . Parameters: $v_h : 168$ , $v_l : 88$ , $SSIM : 0.73$ , $PSNR : 17.79$ .	
MPEG4 Compression	Video is compressed to the target bit rate of $b$ kbps with divX MPEG4 codec. Parameters: $b : 5kpbs$ , $SSIM : 0.73$ , $PSNR : 26.84$ .		Clipping in Time	Some percentage, $(l)$ , of the total number of frames are clipped from both beginning and end of video. One of the missing frames is shown on the right. Parameter: $l : 8\%$ .	
Fade-over	The video sequence, on one extremity, fades in from a different video, and on the other extremity, it fades out into another video sequence. The fading effects take place over $l$ percent of the number of frames. (See also Figure 10(a))Parameter: Fadeover percentage: 8%.		Lossy Channel	Video, first compressed to 30 kbps with Xvid MPEG4 codec, is streamed over lossy channel with packet drop rate of $r$ . The hash comparison takes place between the received streaming video and the compressed video (not the uncompressed original), as in Figure 9. Parameter: Channel's IP packet drop rate: 1%	
Frame rotations	Each frame of the video is rotated separately. Parameter: Rotation degrees by 1, 3, 5 and 7 degrees.		Circular Frame Shift	Each frame of the video is circularly shifted both in row and column senses. Parameter: Shift amounts by 1%, 3%, 5% and 7% percent of frame dimensions.	
Substitution attack	1-second (25 frames) portions of a video sequence were replaced by another content, not necessarily of the same genre. Parameter: 1-second long substitutions at random locations. See illustration in Figure 10(b)		Frame drop	This attack simulates a lossy channel in its extreme when the damaged packets coincide with the frame headers. Lost frames are recuperated via linear interpolation. Parameter: Drop rate varies over 30%, 50%, 70% and 90% of randomly chosen frames.	
Frame rate change	The frame rate of the video is decreased via temporal filtering and subsampling or increased via interpolation. Parameter: Frame rate change by factors of 0.25, 0.33, 0.50, 2, 3, 4.				

perceptual hash. Notice that the original QCIF video is compressed by a factor of 1520.

6) *Lossy Channel*: Under severe packet drops three types of distortion occur: the whole frame can be dropped,

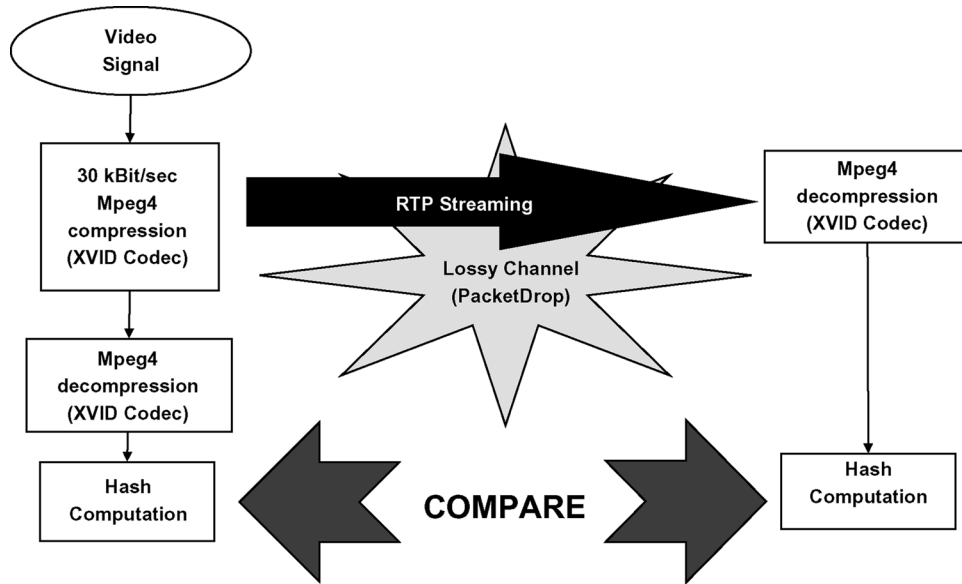


Fig. 9. Setup for lossy (packet drop) channel experiments.

TABLE II  
INTRA-HASH AND INTER-HASH AVERAGE  $\delta$  SCORES (HAMMING DISTANCES)  
UNDER MODIFICATIONS

Modif. Type	Mean Intra-Hamming Distances		Mean Inter-Hamming Distances	
	DCT	RBT	DCT	RBT
Blurring	1.33	2.69	32.01	31.12
AWGN	1.86	1.13	32.02	31.33
Brightness Increase	6.35	7.81	32.02	31.1
Brightness Decrease	4.72	7.51	32.02	31.59
Contrast Increase	7.19	7.12	32.04	31.64
Contrast Decrease	0.29	0.18	32.01	31.29
MPEG4 Comp.	1.12	0.7	32.01	31.31
Lossy Channel	2.2	2.3	32.00	31.43
Clipping in Time	10.79	11.29	31.98	31.30
Fadeover	7.71	6.45	32.00	31.23
Frame rotation (3°)	8.1	13.7	32.00	31.09
Frame shift (3%)	9.9	9.2	31.95	30.90
Frame drop (70%)	2.7	3.7	32.00	31.66
Framerate chge (1/4)	2.75	1.95	32.05	31.53
Framerate chge (4/1)	1.75	1.05	32.04	31.59

which causes deformation in the temporal waveform; second, spatial data losses within a frame are padded with black patches causing distortion in the spatial low-pass coefficients; third, lost spatial data is recuperated with a concealment technique causing slighter content perturbation. Notice that most networks would provide throughputs significantly better than the 99% rate. Notice that in the illustration in Table I, in the left image the codec has tried to conceal the error by replenishing the missing blocks around the nose of Foreman from previous frames. In the right image, the codec has failed in concealing the error and has padded the missing blocks with black regions.

7) *Fade-over*: Fade-over attack is illustrated in Fig. 10(a). Under reasonable levels of fade-over attack, most of the content information is still preserved in the video clip.

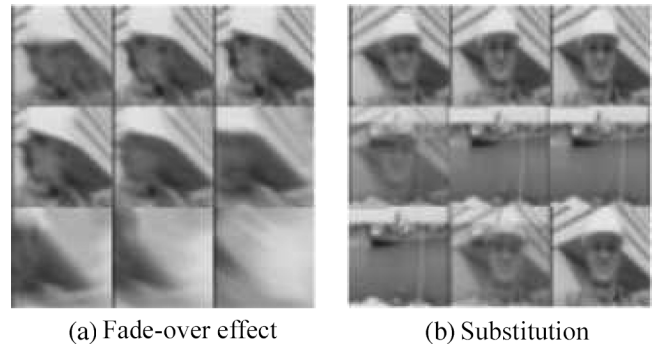


Fig. 10. Normalized video illustrations of selected modifications: (a) Fade-over effect in the Foreman sequence. (b) Substitution: Frames 5,6 and 7 are substituted with a scene from the Container sequence.

- 8) *Clipping in time*: This can be considered to be a more severe version of the fade-over attack where the scene that fades in is all blank/black. In the fade-over phase, there are still remnants of the original video; hence as expected, the performance under clipping is inferior to the fade-over case.
- 9) *Frame rotation*: We consider two geometric modifications, namely frame rotation and frame circular shift. The third possible geometric modification, that of scaling is excluded because, as stated in Section II-A, each video is first converted to QCIF size and further to size of  $32 \times 32 \times 64$  in preprocessing and normalization phase, which wipes out any kind of scaling from the video. The averaged Hamming distances, averaged over 40 video sequences, is presented in Fig. 11(a) for each rotation degree. RBT suffers more from frame rotations as compared to DCT. In any case, it can be roughly said that DCT based hash and RBT based hash can work safely under up to 7 degrees and 3 degrees of rotation, respectively.
- 10) *Frame circular shift*: We effect shifts by 1 to 7% of the frame size: for example, the 5% shift for the QCIF frame

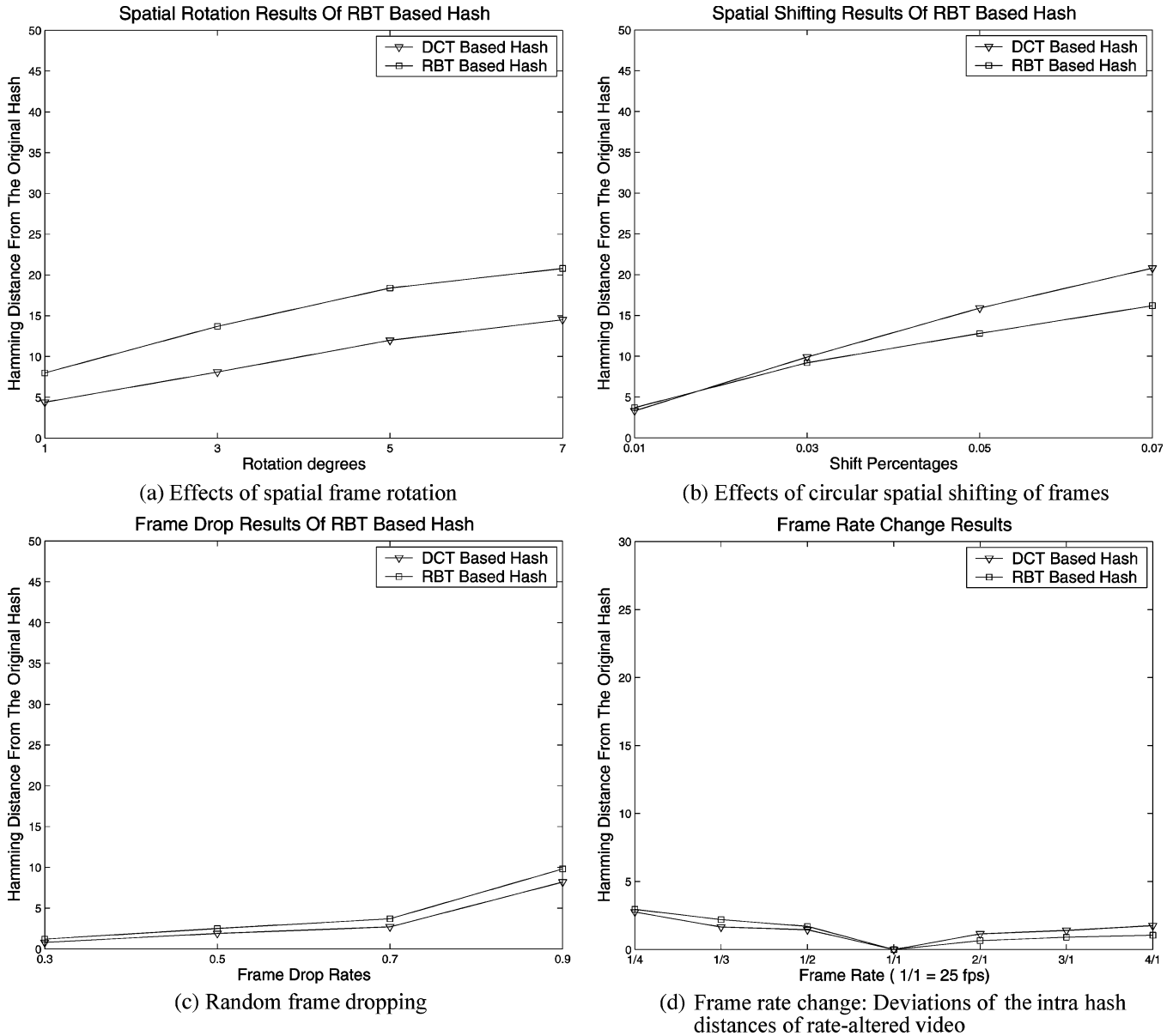


Fig. 11. Average intra hamming distance under the modifications of (a) frame rotation; (b) frame circular shift; (c) random frame dropping; (d) frame rate change.

corresponds, respectively, to row-wise 9 and column-wise 7 lines. The averaged Hamming distances are given in Fig. 11(b). It is observed that both DCT-based and RBT-based hashes digress linearly as shifting percentage is increased. Accordingly, the identification and verification performances remain at a satisfactory level up to 5% circular shifting.

- 11) *Random frame dropping*: This modification corresponds to a lossy channel when the damaged packets always coincide with the frame headers. After frame drops at randomly chosen locations, the gaps left by dropped frames are filled by linear interpolation from nearest surviving future and past frames in order to preserve the sequence length. The Hamming distances, averaged separately at each frame drop rate, are presented in Fig. 11(c). It is observed that, both the RBT-based and the DCT-based hashes survive for frame drop rates as high as 90%. That is be-

cause, the undropped frames are spread all over the video and thus the hash function is able to extract sufficient temporal information. Due to random and less robust nature of RBT based hash, its performance is slightly worse than DCT based hash.

- 12) *Frame rate change*: In this experiment, we test the performance of our hash when the video frame rate is altered but the content is preserved. Rate reduction is realized by anti-aliasing filter and subsampling the frames, while rate increase is achieved via an interpolation filter. Since the original frame rate is 25 fps, we obtain the frame rates 6.25, 8.33, 12.5, 25, 50, 75, and 100 fps or alternatively, we have 75, 100, 150, 600, 900, and 1200 frames instead of the original 300 frames, though the durations are the same and content is preserved to the extent allowed by the respective interpolation and decimation operations. Fig. 11(d) plots the hamming distance between hashes of the frame-rate-al-

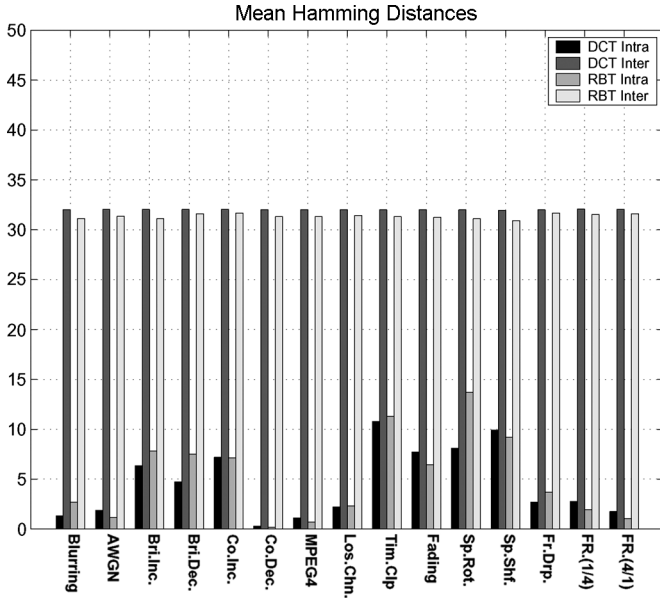


Fig. 12. The mean values of intra-hash and inter-hash statistics of RBT- and DCT-based hashes.

tered video and the hash of the original video. It is interesting to observe that the hashes of rate-altered videos do not deviate significantly from their original.

Table II and Fig. 12 summarize the inter-hash statistics. Notice that the averages stay very close to the theoretical value, under independence assumption, of  $T/2 = 32$ . As already remarked in Section III, Fig. 8, the variances of DCT-based hash differ somewhat from theoretical value. This is because DCT coefficients in adjacent bands may not be after all independent; in other words close-by frequencies tend to have similar magnitudes and signs.

**B. Identification and Verification Performance**

The identification problem is defined as the ability to recognize a video clip in a database of several other video clips. For example, given a 12-s or 300-frame video clip, the algorithm must identify that clip within a database of hundreds or thousands of other video clips, or, browse through and spot it in long video sequences (e.g., a two-hour movie). The identification or detection performance can be measured in terms of the percentage of correct recalls.

Thus to identify a video clip, which may be given in the original or in any modified form, its hash is compared against the hashes of all original video clips. The video clip, whose hash has the smallest Hamming distance is declared as the identified video. The ratio of correctly identified video clips to the number of all video clips tested in the database (244 clips in our experiments) determines the identification performance. The no-modification and modified identification performances are presented in Table III. While the overall performance is very satisfactory, the RBT-based hash performs slightly inferior to the DCT-based one.

The verification problem, on the other hand, is defined as the effort to prove or disprove that a video clip is indeed what it is claimed to be. In a verification experiment, one must test both

TABLE III  
IDENTIFICATION PERFORMANCES OF THE DCT AND RBT-BASED HASHES IN PERCENTAGES (244 VIDEO CLIPS) (ALL THE MODIFICATION PARAMETERS ARE AS IN TABLE I)

Methods ⇒ Modifications ↓	DCT Based Hash	RBT Based Hash
No Modification	100	100
Blurring	100	100
AWGN	100	100
Brightness Increase	98.77	92.62
Brightness Decrease	100	99.60
Contrast Increase	99.59	98.77
Contrast Decrease	100	100
MPEG4 Compression	100	100
Lossy Channel	100	97.13
Fade-over	99.18	97.95
Time Clipping	98.36	83.62
Frame rotation (3°)	100	97.5
Frame shift (3%)	100	98.75
Frame drop (70%)	100	100
Frame rate change (1/4)	100	100
Frame rate change (4/1)	100	100

TABLE IV  
VERIFICATION PERFORMANCE

Methods ⇒ Modifications ↓	FAR=FRR Performance		FAR=1% Performance	
	DCT	RBT	DCT	RBT
No Modification	100	100	100	100
Blurring	100	100	100	100
AWGN	100	100	100	100
Brightness Increase	99.59	97.95	99.59	97.95
Brightness Decrease	100	99.59	100	100
Contrast Increase	99.59	99.59	99.59	99.59
Contrast Decrease	100	100	100	100
MPEG4 Compression	100	100	100	100
Lossy Channel	100	97.54	100	96.72
Fade-over	99.59	99.59	99.59	99.59
Time Clipping	99.59	95.08	98.77	90.57
Frame rotation (3°)	100	100	95.00	100
Frame shift (3%)	100	100	100	100
Frame drop (70%)	100	100	100	100
Frame rate change (1/4)	100	100	100	100
Frame rate change (4/1)	100	100	100	100

the “genuine record” as well as all the other “impostor” records in their various altered versions, possibly transfigured by the modifications described in Section IV-C. Verification performance is determined by comparing the hash of each video clip to the hashes of all clips in the database. If the Hamming distance between the test hash and a database hash is below the predefined threshold, then the test video is accepted as genuine; otherwise it is rejected. If the accepted case does indeed correspond to the correct content, then one obtains a correct detection, otherwise it becomes a false alarm (false positive). If the correct content is rejected because its hash distance remains above the decision threshold, then we have a false reject case (false negative). If an “impostor” video clip (a video clip with a different content due, for example, to tampering) is rejected then we have a true negative. The verification performance is measured as the ratio of correct detections to total tests.

The verification performance is computed for both the cases of EER: Equal Error (i.e., (FAR = FRR), and also for the

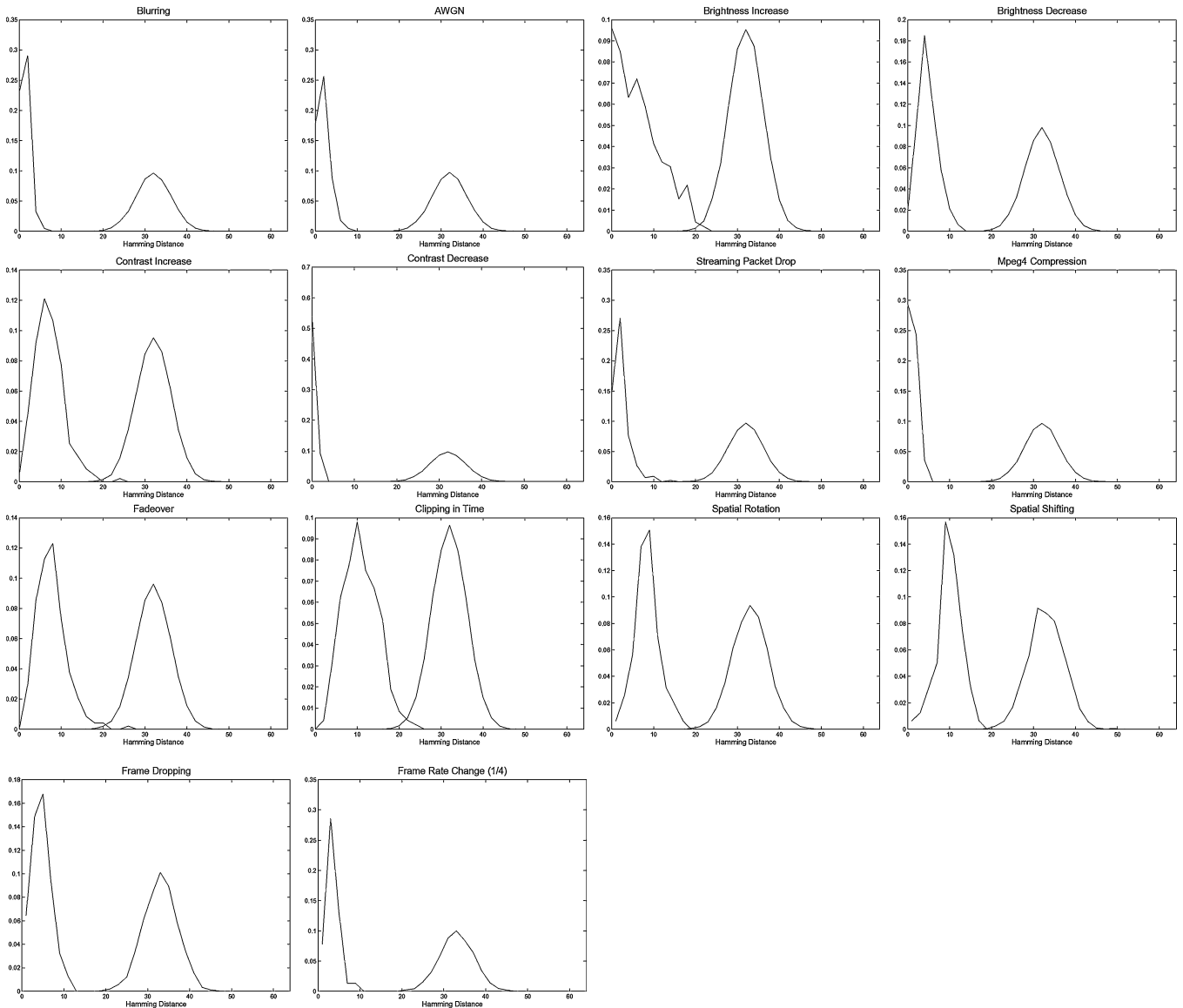


Fig. 13. Inter- and intra-Hamming distance histograms for DCT-based hash. Note that the histogram range is different in each graphic, though the area underneath always adds up to one. The modification parameters are  $3^\circ$ , 3%, 70%, and 1/4 for spatial rotation, spatial shifting, frame dropping and frame rate change respectively. The parameters for other modifications are as stated in Table I.

false alarm rate of 1%. These scores are given in Table IV and the inter- and intra-Hamming distance histograms are plotted in Fig. 13 for DCT-based hashes. The RBT-based hash distances (not shown) plot in a very similar way. Similar to the case of clip identification, DCT-hashes perform slightly better than the RBT-hashes. In either case the deep valley between the two probability humps suggests that thresholds can be easily selected to achieve various false negative—false positive compromises. The main observation is that the proposed hashes, in particular the DCT variety fares very well in verification tests. The only modification under which the verification performance drops substantially (5%) is the 8% time-clipping.

### C. Effects of Different Coefficient Selection Patterns

It is possible to consider other DCT coefficient selection patterns. We have explored with four different patterns, as detailed

in the sequel. Pattern 1 is the  $4 \times 4 \times 4$  pattern, used throughout the paper and shown in Fig. 4, that is,  $\{V_{DCT}(i, j, k); i, j, k = 1, \dots, 4\}$ . Pattern 2, also consisting of 64 coefficients, includes the lowest frequency coefficients to the exclusion of higher frequency terms, that is,  $\{V_{DCT}(i, j, k); i, j, k = 0, \dots, 3\}$ . Pattern 3 contains 128 coefficients from rectangular prism, that is,  $\{V_{DCT}(i, j, k); i, j = 1, \dots, 8 \text{ and } k = 1, 2\}$ . Obviously this scheme contains more of the higher frequency spatial information to the detriment of temporal details. Pattern 4 is the counterpart of Pattern 3 as it picks more coefficients from temporal frequencies. The 128 coefficients within the rectangular prism of  $4 \times 4 \times 8$  consist of  $\{V_{DCT}(i, j, k); i, j = 1, \dots, 4 \text{ and } k = 1, \dots, 8\}$ . The inter- and intra-hash average normalized hamming distances are shown in Fig. 14. It is interesting to observe that the performance does not depend critically of the coefficient selection pattern. Nevertheless, there are minute differences. For example, Pattern 2 is slightly more robust (due to its

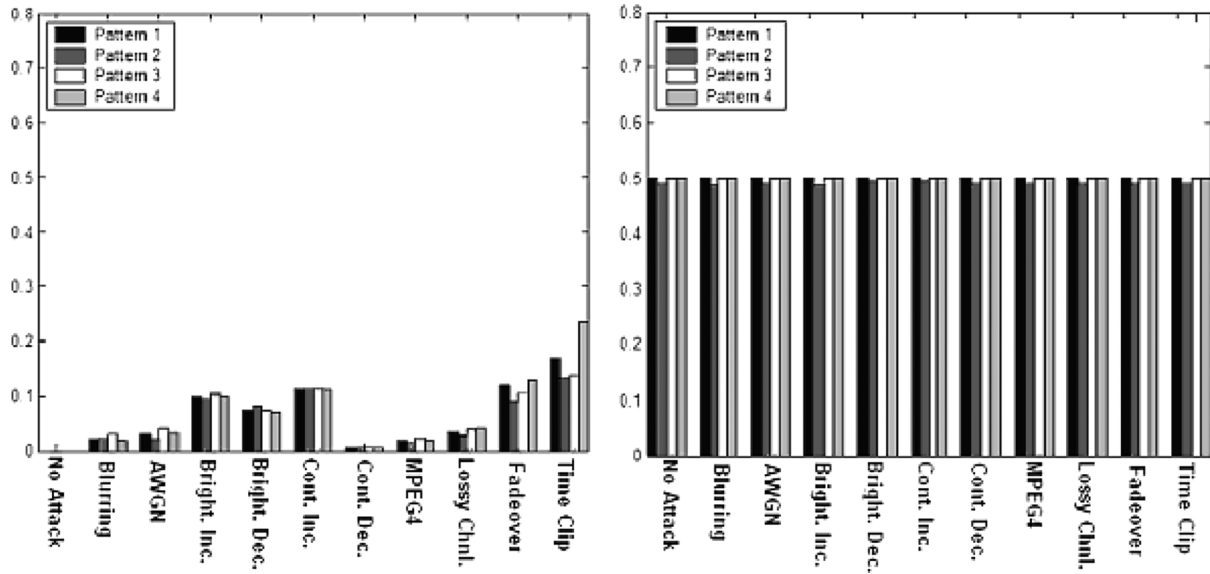


Fig. 14. Different coefficient selection: On the left side, the average intra-hash normalized Hamming distances are plotted, while on the right side, average scores for inter-hash case are given for each type of modification and for each coefficient selection pattern.

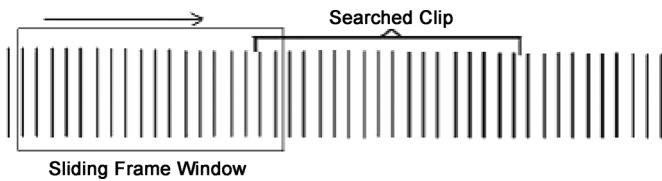


Fig. 15. Illustration of clip search via a sliding window.

lower frequency content) to the detriment of its clip discrimination ability, and Pattern 3, containing lower temporal frequencies, is somewhat more resistant to fade-over and time clip modifications.

**D. Broadcast Monitoring**

In this experiment, we search a given video clip in a longer video data, where it may be possibly embedded. The search is enabled by sliding a frame window, the same size as the estimated clip length, throughout the longer sequence (Fig. 15). We assume that, the original clip length is known and the length of the sliding window is chosen accordingly. For every step, the hash of the sequence covered by the window is calculated and compared with the reference hash. For example, the reference hash can be that of a TV advertisement spot, and the task is monitoring and automatically counting the time instances at which the spot is broadcast. Since previous experiments have shown that hashes are resistant for fade-over effects up to 8% or less (28 frames out of 350), we slid the window by steps of ten frames.

Fig. 16 illustrates the DCT- and RBT-based hash searches. Clips of length 300 frames from four video genres are used to test the hash-based search. The target clips are placed in 800-frames long videos, taking place between frame numbers 350 to 650. Clips are always embedded in their own genre, for example, sport clips within sport videos.

In the plots in Fig. 16, the *x*-axis shows the frame position in steps of 10 while the *y*-axis indicates the Hamming distance between reference hash and the hash of the sequence under the running window. One can notice that the Hamming distance drops dramatically at the correct spot of the clip. Although we assume that the clip length is previously known, even if we do not know exactly the length of the video clip, the extensive experiments on time clipping and fade-over attacks give enough idea about the robustness of the scheme against clip length uncertainty. Regarding the result of those attacks presented in Sections V-A and V-B, we can safely detect the desired video if the length is estimated with maximum error of 8% of the original length (This corresponds to a difference of 24 frames within a clip of 300 frames). In fact, the Hamming distance starts dropping gradually as the window approaches to the correct position, which suggests that, even if the window length is not exactly the same as the clip length, the algorithm can still give an indication of the desired clip. In one case of RBT-based experiments (Fig. 16, upper right plot) the Hamming distances remained at relatively lower levels (Hamming distances around 12) in the first part of the browsed video (frames 0 to 350). This was due to the fact that the content was quite similar to that of the target video, where frames 0–650 all covered the talk of a politician.

**E. Reverse Play**

In this experiment the video is reverse played, that is, the frame indices are reversed. As expected, the even temporal frequencies (Planes 2 and 4 in the  $4 \times 4 \times 4$  DCT cube in Fig. 4) are insensitive to the time reversal while the odd frequency terms (Planes 1 and 3) are affected, in that they are themselves reversed. Fig. 17 illustrates the case, where the hash bits resulting from thresholding of  $4 \times 4$  spatial DCT coefficients per temporal DCT plane are displayed in lexicographic order. In this figure, the original hash, the hash of the time-reversed video, and the positions in which the bits differ are shown. In this respect, we can scan the video either way, from head to end or from

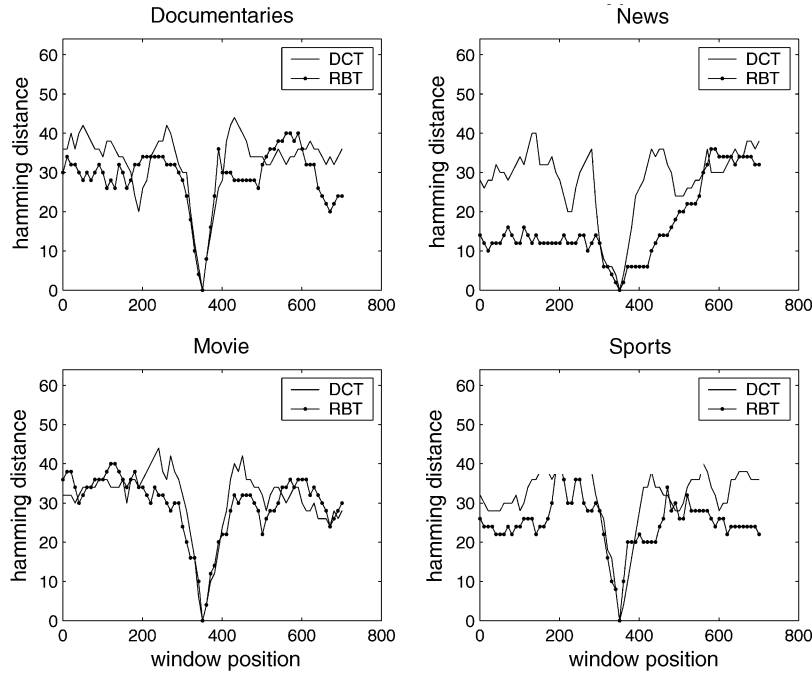


Fig. 16. Broadcast monitoring results.

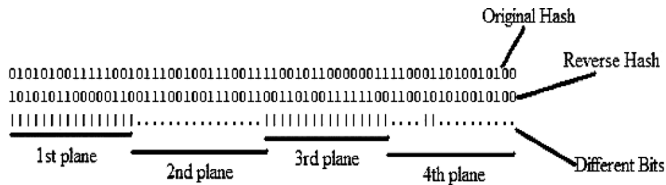


Fig. 17. Differing bit locations of DCT based hashes of a video and its reversed played version.

end to head. One use of this property could be the enabling of double check on a suspected video. In contrast there is no such symmetric property for RBT transform, so that the video cannot be browsed bidirectionally.

#### F. Effect of Video Clip Length

We have up to this point used as the “video quantum” or video clip sizes extending over 10–15 s or 300–400 frames at 25 fps. It is intriguing to investigate the identification/verification performance when the time extent of the video quantum is altered, say, down to a few seconds or up to a few minutes. Recall that whatever the length of the input video, it is always normalized to a fixed number of frames (which is 64 frames in our experiments). Hence, the longer the video, the more will be the temporal compression, hence the risk increases of glossing over some video details in time, or of not being able to detect content replacement, content deletion, etc., types of malicious attacks.

We ran experiments with different clip lengths in number of frames (at 25 fps) and subjected them to several signal-processing and channel impairment modifications described in Table I. 25 video sequences having 100, 150, 250, 400, and 700 frames are selected from each 4 genre, that is, overall 100 clips (For each of 4 genre, 5 clips for each of 5 video

length possibilities). The clip lengths in time units varied from 4 s to 28 s. After applying modifications to these clips, the Hamming distances between the hashes of original sequences and those of modified sequences are calculated. This exercise was repeated separately for each of the five chosen clip lengths. Hamming distances averaged over the 20 clips for each length are plotted in Fig. 18(a) as a function of clip length. In this limited experiment, no significant trends were observed.

Although no significant effect of frame length is observed on the performance of our hash functions under the attacks specified in Table I, it wouldn’t be a nonsense expectation that the sensitivity of the hash towards frame substitution attack would decrease as the clip length was increased. That is because, since the substituted clip length is kept fixed, the ratio of the substituted clip length to whole clip length and thus the relative amount of changed information is decreased as the original clip length increased. In other words, same amount of substitution yields to less alteration in the normalized video as the clip length increases since every video clip is normalized to a fixed number of frames regardless of its original length. Consequently the hash of the attacked video clip would be more alike to the original hash as the clip length increases. We chose the substitution length of 1-s as the minimum duration since intuitively scenes lasting less than 1 second may not be visually impacting. Since the aim of this experiment is to show the decrease in the sensitivity of the hash to substitution attacks as the video length is increased, we chose to embed 25-frame substitutions from different genres at random positions within sequences of increasing length of 100, 150, 250, 400, or 700 frames. Such a substitution is illustrated in Fig. 10(b). The sensitivity of the hash function to substitution attack is reduced as video length increases since the substituted part plays an increasingly minor role. Fig. 18(b) reveals that substitution attacks, altering up to 8% of the video sequence (1 s substitution in a 14 s clip), can

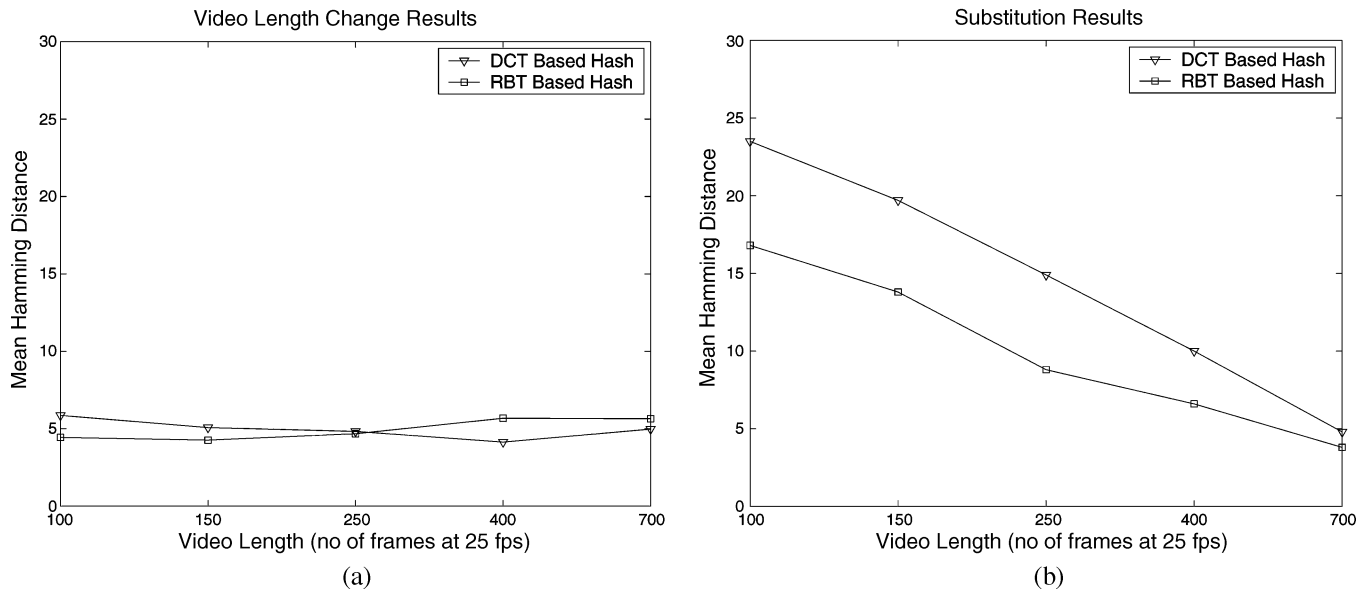


Fig. 18. Effect of Clip Duration: (a) The mean of Hamming distances between clipped and original video sequences for different video lengths for RBT-based hash. DCT-based hash performs almost exactly the same. (b) The mean of hamming distance between the hashes of video clips subjected to substitution attack and the original video clips. The distance gets closer as the video length increases.

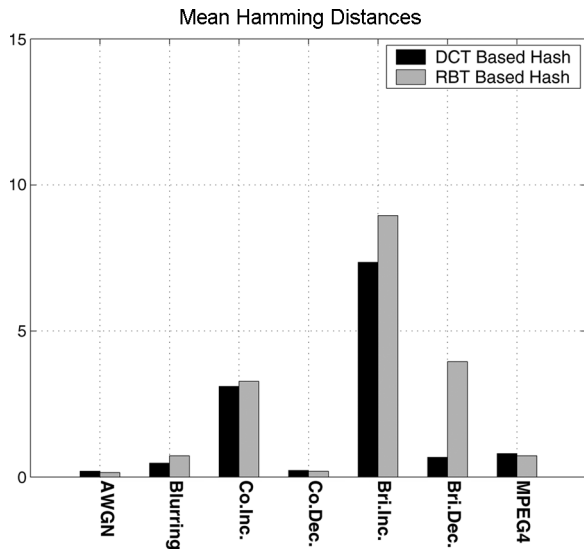


Fig. 19. Mean intra Hamming distances for attacks gauged at equal perceptual quality SSIM.

be caught. For longer sequences, short substitutions can be detected by analyzing the sequence as a concatenation of shorter video subsequences.

### G. Iso-Distortion Performance

In Sections V-A and V-B, the performance of the algorithms was observed under “stress testing” with most severe level of modifications (Section IV-C), where the video quality was often reduced to a low and commercially unacceptable level. The purpose here was to show under what severity of modifications the hash function could still perform reasonably well. As an alternative testing scheme, we investigated the performance of the hash functions at a fixed modification severity. The modification level

was quantified in terms of the quality metric SSIM: Structural Similarity Index. We fixed the SSIM at 0.7, and measured also the Hamming distance between the original and attacked video hashes. We repeated this experiment over 80 video sequences from four genre (20 for each genre). The resulting Hamming distances are averaged for each modification and presented in Fig. 19. Although, all modified video sequences are equally distorted in terms of SSIM, we observed that the both RBT and DCT based hashes have their worst performance against the modifications where pixel saturation to 255 and/or clipping to 0 occurs. The brightness increase and decrease make lighter regions of the frames totally white (saturation) and darker regions totally black (clipping to 0), respectively. Similarly, the contrast increase makes both lighter regions totally white and darker regions totally black at the same time. This causes the elimination of signal dynamic in these regions and errors in resulting hash values. For the remaining modifications, the signal distortion occurs in higher frequency components and thus has no significant effect on hash values. In conclusion at SSIM video quality of 0.7, the hashes pass the test under all attack types.

### H. Performance Comparison of the Proposed Hash

For a fair assessment, we have chosen to compare our algorithm with that of Oostveen *et al.* [9], which is based on  $2 \times 2$  spatio-temporal Haar filters. Other video hash algorithms proposed in the literature compute the hash on a frame-by-frame basis and then concatenate these hash sequences. These algorithms are very vulnerable to temporal desynchronization modifications, such as frame dropping, hence would be unfair to put them in performance comparison with ours.

The compared algorithm divides each frame into nonoverlapping blocks and computes hash from the difference between mean values of consecutive blocks both along spatial and temporal axes. Thus each bit of the hash depends on the two neighboring blocks in the current frame as well as the corresponding



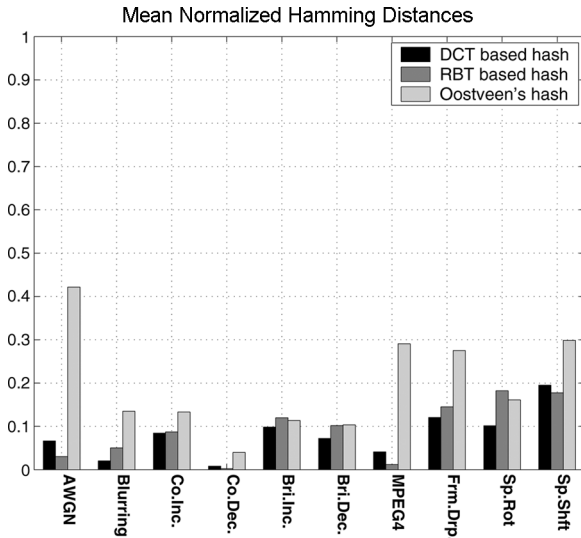


Fig. 20. Comparison of DCT/RBT-based algorithms and the Oostveen's algorithm [9].

neighbors in the previous frame. Since the hash length is different for each method, we normalized Hamming distance by dividing it by the length of the hash. For example, when normalized Hamming distance is 0.5, this means that the compared hashes are maximally different whatever method is used.

Since the hash length of Oostveen's method depends on number of frames we could only include those modifications where the video length remains constant. This leaves us with the following modifications: AWGN, blurring, contrast increase and decrease, brightness increase and decrease, MPEG4 compression, frame dropping followed by interpolation, spatial rotation and spatial shifting. The parameters of the first seven modifications are chosen the same as in Table I. Frame dropping is run at 70% loss, the spatial rotation and shifting, are realized with parameters 3 and 4%, respectively.

It is observed in Fig. 20 that our methods perform uniformly equal or better than Oostveen's method for intra-hash statistics. Since Oostveen's method uses more limited spatio-temporal information (only two consecutive frames are considered), under frame dropping it performs significantly worse than our method, which embraces the entire video sequence. The compared method has poorer performance under the modifications that lead to saturation to 0 or 255 gray values, as for example, in brightness manipulation case or high AWGN case. As for inter-hash statistics Oostveen's method performs as well as ours.

### I. Analysis of Forgery

It was remarked earlier that robust hashing and content-based retrieval differ in that, unlike a content-based retrieval system, a robust hash function should generate unique hash values whenever the content is perceived as different, despite the remnant commonality of the original and the doctored versions. Thus the perceptual hashing algorithms must walk the thin line between efficient retrieval and authentication of the genuineness of the content. In this context, the robust hashing must meet the challenge of authenticating forged video documents.



Fig. 21. Foreman is forged into background of Akiyo.

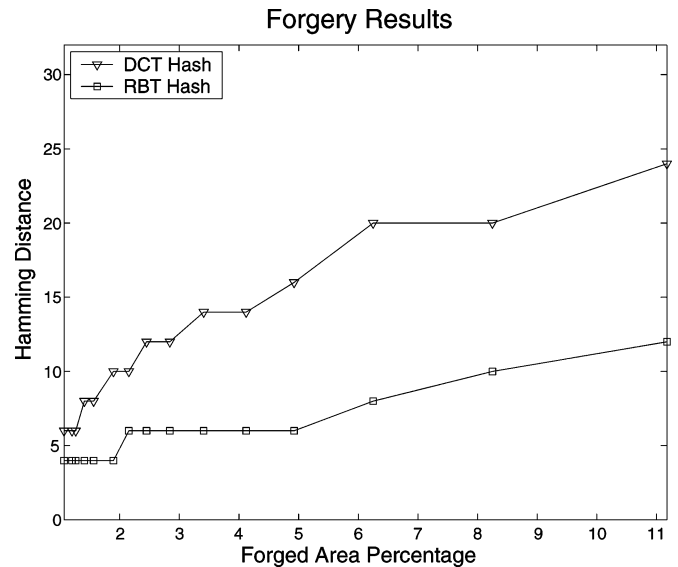


Fig. 22. Hamming distance between the hashes of the original and its region as a function of forged area percentage.

Doctoring video content and creating forged versions must be done cleverly and it is a standalone area of expertise. We used, therefore, a simple method to create forged video clips in order to test our two algorithms against forgery. We forged Foreman sequence into the static background of Akiyo sequence, as illustrated in Fig. 21, where approximately 5% of the frame area has been manipulated. We increase the forged area in small steps in a series of experiments and compute its hash difference from the original one. The Hamming distances between hash functions of the forged and original are plotted in Fig. 22, where the abscissa indicates the percentage of the forged area in a frame. We observe that DCT-based hash is significantly more sensitive to forgery than the RBT-based hash due to its well-ordered structure and properties described in Section III. However, the RBT-based hash has also an acceptable operation region when the iso-distortion results in Section V-G are considered. For example when the threshold is set to 5, innocent manipulations can be discriminated from the content altering forgeries. On the other hand, if higher sensitivity to forgery attacks is desired, the

frames can be divided into blocks and each block can be separately hashed. This would on the one hand serve to localize the content manipulation and on the other hand would enable decision fusion schemes. If one of the block hashes differs from the its counterpart above a threshold, then that subclip would be declared as forged, and the whole clip might be selected as inauthentic depending on the application.

## VI. CONCLUSION AND FUTURE WORK

We have presented a method for computing a robust hash from video clips for the purposes of identification and verification. The proposed hash function is shown to be remarkably robust against signal-processing modifications and channel transmission impairments. These modifications may cause severe perturbations on the video signal, but apparently do not significantly modify the content information as captured by the hash algorithm. Recall that some of the modifications described in Table I are “exaggerated” for stress testing of the algorithm, resulting even in unacceptable viewing quality. On the other hand, hashes of different video contents yield widely differing hash sequences. Thus, based on the proven uniqueness (randomness) and robustness, the proposed hash algorithm can be used in applications of broadcast monitoring, video database searching, watermarking for tamper-proofing, etc.

The experiments show that DCT-based hash is slightly better in identification and verification tests of video clips, as seen in performance Tables III and IV in that it more robust against most of the modifications as illustrated in Fig. 12. Moreover, DCT based hash is much easier to compute since DCT is a well known transformation and processors are most likely to be optimized for it. So, DCT-based hash would be the better choice in a non-adversarial scenario, where one does not expect malicious manipulation of the video material, such as, the own archive of a video distributor, or identifying video clips within a long broadcast. On the other hand, the RBT-based hash, which has proven to be slightly weaker, would be used in an adversarial scheme, where a malicious party might attempt to modify the video content without causing changes in the hash value or conversely minimally change the content to engender a different hash. In the first instance the pirate claims a legal hash for an illegal content while in the second instance the pirate can exploit a copyrighted material since the hash does not collide with any existing one.

The comparison of our RBT- or DCT-hashes with its nearest competitor, the Haar-based video hashing of Ostveen *et al.* [9] shows that our schemes outperform it under several attack scenarios. Finally, our experiments on forgery attack indicates that by judicious setting the authentication threshold, most of the innocent attacks can be authenticated whereas the forgeries are rejected.

There are several avenues to be explored for future work. First, alternative transform schemes, such as discrete Fourier transform, discrete wavelet transform or overcomplete basis sets can be used. Second ancillary information, such as hash from audio and/or hash from color components can be included. The audio hash can form useful complementary information in schemes where video clips are captured with different cameras, as could occur in security applications. Third, implementation

issues must be addressed for efficiency in broadcast monitoring or in security cameras. For example, instead of having the 3-D spatio-temporal video data cube to march in time, one can first 2-D-transform each frame, and then selectively apply the 1-D transform over temporal axis. Thus as the window slides past, the 2-D-transforms of the frames can be re-used so long as they are subtended by the window. Thus, for each hash instance, only the newly covered frames will have to be 2-D spatially transformed. Fourth, an intriguing alternative would be to implement the hash computation in the compressed domain, for example, between I-frames. Longer video segments, let’s say a whole movie itself, can be thought of as a concatenation of video clips, and hence the hash of the whole as a concatenation of fingerprints. The merging and compression of the concatenated hashes via error correcting decoders is an interesting alternative. Finally, the security issue is not solved thoroughly as the random frequencies employed in the RBT hash provide a limited search space and a scheme in the vein of [13] must be assessed.

## REFERENCES

- [1] J. Fridrich and M. Goljan, “Robust hash functions for digital watermarking,” in *ITCC’00: Proc. Int. Conf. Information Technology: Coding and Computing*, 2000, p. 178.
- [2] R. Venkatesan, S. Koon, M. Jakubowski, and P. Moulin, “Robust image hashing,” in *Proc. IEEE Int. Conf. Image Processing*, 2000, vol. 3, pp. 664–666.
- [3] F. Lefebvre, B. Macq, and J. Czyz, “A robust soft hash algorithm for digital image fingerprint,” in *IEEE ICIP*, Barcelona, Spain, Sep. 2003.
- [4] J. S. Seo, J. Haitsma, T. Kalker, and C. D. Yoo, “A robust image fingerprinting system using the radon transform,” *Signal Process.: Image Commun.*, vol. 19, no. 4, pp. 325–339, 2004.
- [5] M. K. Mihcak and R. Venkatesan, “New iterative geometric methods for robust perceptual image hashing,” in *Proc. Digital Rights Management Workshop*, Nov. 2001.
- [6] Y. Caspi and D. Barger, “Sharing video annotations, international conference on image processing,” in *ICIP’04*, Singapore, Oct. 2004.
- [7] Z. Yang, W. Oop, and Q. Sun, “Hierarchical non-uniform locally sensitive hashing and its application to video identification,” in *ICIP’04*, Singapore, Oct 2004.
- [8] V. Monga and M. K. Mihcak, “Robust image hashing via non-negative matrix factorizations,” in *ICIP’05*, Genoa, Italy, Sep. 2005.
- [9] J. Ostveen, T. Kalker, and J. Haitsma, “Visual hashing of digital video: applications and techniques,” in *SPIE Applications of Digital Image Processing XXIV*, San Diego, CA, July 2001.
- [10] J. Fridrich, “Robust digital watermarking based on key-dependent basis functions,” in *The 2nd Information Hiding Workshop*, Portland, OR, Apr. 1998.
- [11] X. Yang, Q. Tian, and E. Chang, “A color fingerprint of video shot for content identification,” in *Proc. 12th Annu. ACM Int. Conf. Multimedia*, New York, 2004, pp. 276–279.
- [12] J. Yuan, L. Duan, Q. Tian, and C. Xu, “Fast and robust short video clip search using an index structure,” in *Proc. 6th ACM SIGMM Int. Workshop on Multimedia Information Retrieval*, New York, 2004, pp. 61–68.
- [13] R. Radhakrishnan, Z. Xiong, and N. D. Memon, “On the security of visual hash function,” in *Proc. SPIE, Electronic Imaging, Security and Watermarking of Multimedia Contents V*, Santa Clara, CA, Jan. 2003, vol. 5020.
- [14] R. C. Dubes and A. K. Jain, “Clustering methodologies in exploratory data analysis,” *Adv. Comput.*, vol. 19, pp. 113–228, 1980.
- [15] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error measurement to structural similarity,” *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 601–612, 2004.
- [16] Z. Wang, L. Lu, A. C. Bovik, and J. Kouloheris, “Video quality assessment based on structural distortion measurement,” *Signal Process.: Image Commun.*, vol. 19, no. 1, 2004.
- [17] K. M. Pua, J. M. Gauch, S. E. Gauch, and J. Z. Miadowicz, “Real time repeated video sequence identification,” *Comput. Vis. Image Understand.*, vol. 93, no. 3, pp. 310–327, 2004.

- [18] M. Johnson and K. Ramchandran, "Dither-based secure image hashing using distributed coding," in *Proc. IEEE Int. Conf. Image Processing*, Barcelona, Spain, Sep. 2003.
- [19] R. B. D'Agostino and M. A. Stephens, *Goodness-of-Fit Techniques*. New York: Marcel Dekker, 1986.
- [20] C. Y. Lin and S. F. Chang, "A robust image authentication method distinguishing jpeg compression from malicious manipulation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 2, pp. 153–168, Feb. 2001.
- [21] S. Bhattacharjee and M. Kutter, "Compression tolerant image authentication," in *Proc. ICIP*, Chicago, IL, 1998, vol. 1, pp. 435–439.



**Baris Coskun** (S'06) received the B.S. and M.Sc. degrees in electrical engineering from Bogazici University, Turkey in 2001 and 2004 respectively. He is currently pursuing his Ph.D. degree in the Electrical and Computer Engineering Department, Polytechnic University, Brooklyn, NY.

He is currently a Research Assistant at Polytechnic University. His research interests are multimedia communications, signal processing, and security.

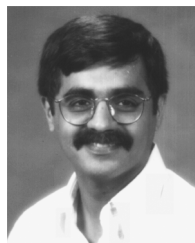


**Bulent Sankur** (M'76–SM'90) has received the B.S. degree in electrical engineering from Robert College, Istanbul, Turkey, and the M.Sc. and Ph.D. degrees from Rensselaer Polytechnic Institute, Troy, NY.

He has been teaching in the Department of Electric and Electronic Engineering, Bogazici (Bosphorus) University, Istanbul. His research interests are in the areas of digital signal processing, image and video compression, biometry, cognition, and multimedia systems. He has held visiting positions at the University of Ottawa, Technical University of Delft, and

Ecole Nationale Supérieure des Telecommunications, Paris.

Dr. Sankur was the chairman of ICT'96: International Conference on Telecommunications and EUSIPCO'05: The European Conference on Signal Processing as well as the technical chairman of ICASSP'00.



**Nasir Memon** (M'89) is a Professor in the Computer Science Department at Polytechnic University, New York. He is the director of the Information Systems and Internet Security (ISIS) lab at Polytechnic. His research interests include data compression, computer and network security, digital forensics, and multimedia data security.