The Core of Video Fingerprinting: Examples of Feature Extraction

Wei-Lun Chao

Graduate Institute of Communication Engineering, NTU

Abstract

In previous two reports, I have briefly introduced the concept and structure of video fingerprinting, and discussed the difference between it and watermarking. And now, I'll cut into the main topic of this technique, the ***Feature Extraction***: What kinds of features have been used to construct the video fingerprints? What are their characteristics?

Content

1. Introduction

Because there is still no published book for video fingerprinting (only a paper collection from Amezon.com), my understanding of it mainly comes from the papers I have read. From these papers, I found authors either create new methods or improve proposed methods then verify the performance through experimentation, so I can't not clearly claim what kind of features is suitable to construct a robust video fingerprints, but make a detail summary and classification about the methods I have known and their characteristics against transformations.

In the report ***"Introduction to Video Fingerprinting"***, there have been several classifications based on different aspects of CBCD techniques, and now I generally separate techniques into ***"global-description-based"*** and ***"local-description-based"*** then introduce them in order. More detail focuses on the techniques of my project, and the content of my partner's, Julein Dubois, master thesis is included.

2. Global-descriptor-based signature

In this section, we talk about global-descriptor-based signatures. Global-descriptor-based signature exploits features from all the pixels in a frame; while the local-descriptor-based signature exploits features only from and around the ***points of interest*** in a frame.

2.1 Introduction to image-based and sequence-based

Based on a small video clip, there are kinds of signature-storing forms. Signature extraction may exploit feature from the whole video **(*sequence-based*)** or from a set of key frames **(*image-based*)**. Image-based signatures first find a set of key frames, then extract features from each key frame as a vector (Here I just call the set of features from each frame a vector), and finally either combine all the vector as a new vector or build a vector sequence. Sequence-based signature gets vectors from every frame, and finally obtains a new vector, a combined vector sequence (several vectors are combined as a new vector), or just a clip-long vector sequence.

An interesting situation we'll find out is that the signature extraction technique on each key frame can also be used on all the frames for sequence-based signature. In the rest part of this section I'll introduce different kinds of global descriptors.

2.2 Shot boundaries

This method was proposed by Indyk et al. [6]. They use temporal fingerprints based on the shot boundaries of a video sequence, and the time distance between boundaries is its signature. The feature of a boundary image isn't used since the content of this image is unreliable, usually black or hard to be identified by human eyes. This technique needs a shot-boundary-detection algorithm, and it can be efficient for finding a full movie, but may not work well for short episodes with a few boundaries.

2.3 Color, luminance, intensity and histogram

This subsection introduces techniques using histograms or functions of intensity as their signatures.

2.3.1 Color histogram

This measure was originally proposed by Naphade et al. [7] and compared by Hampapur and Bolle [3]. ***YUV histograms*** are used as the signature of each frame in the sequence and the use of ***histogram intersection*** as a distance measure between two signatures (frames). Here I'll present the matching algorithm, let the reference signature be *R(t)* and a test signature of length L_T be *T(t)*, *t* is the frame index. The normalized histogram intersection (NHI) is given by

$$\mathbf{NHI\ (t)} = \frac{1}{L_T}\sum_{i=t-L_T/2}^{t+L_T/2} I(H_{Ri}, H_{Ti}) \qquad (2.1)$$

with

$$I(H_{Ri}, H_{Ti}) = \frac{\sum_{l=1}^{l=M} min(H_{Ri}(l), H_{Ti}(l))}{\sum_{l=1}^{l=M} H_{Ri}} \qquad (2.2)$$

where *H* means the histogram and *M* is the number of bins of it. The NHI measures the similarity between *R(t)* and *T(t)*. The maximum NHI(t) at point t_{max} is the best match.

YUV color space is not the uniquely one used in histogram methods, YCbCr, RGB, or even gray-level histogram can also be computed, and the matching algorithm can be improved or changed, too.

2.3.2 YCbCr histogram

This is one part of our project presented in [5]. Rather than computing a vector for every frame and rather than using a selected number of bins for each component of color separately (like the one in 2.3.1), this technique computes a combined vector from a bunch of frames.

For a video-scene clip with *N* frames, and the number of vectors we want is *K*, we use a window having *P=N/K* frames for histogram computation. For a window of *P* frames, we allocate five bins for each of the *Y, Cr, Cb* axes, thus making it a 125-dimensinal feature per window. So the effective size for the entire video, considering *K* key frames = *K* × 125. Figure 2.1, 2.2 shows this technique. An important improvement is that we should normalize the 125-dimensional feature to solve the frame drop problem.

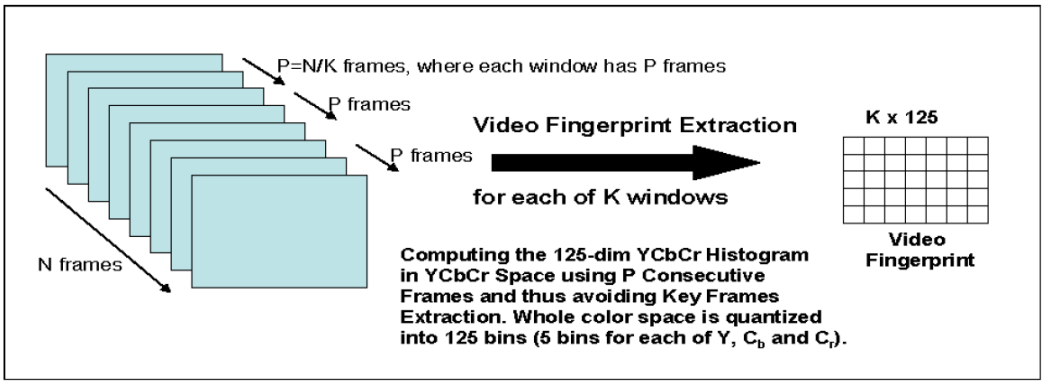


Figure 2.1: Generation of YCbCr histogram-based signature.

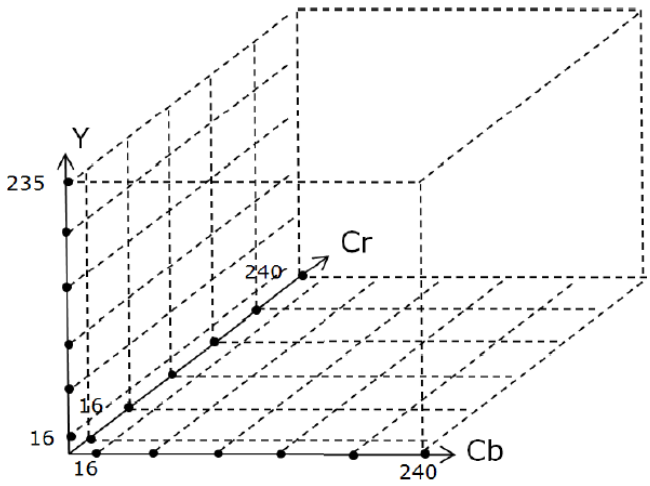


Figure 2.2: The color space of YCbCr

In our project, we use the MATLAB built-in function to transform a RGB color vector into YCbCr. The number of bins for each RGB components is 256, while the number is different for YCbCr. (Y∈[16, 235] and Cr, Cb∈[16, 240])

2.3.3 Other techniques

Lienhart et al. [8] describe a system for video clip matching using the color coherent vector to characterize key frames from the clip. Sanchez et al. [9] discuss the use of the principal components of the color histogram of key frame for copy detection.

A compact and robust fingerprint based on a novel space-time color feature was proposed by Yang et al. in [23]. Here, each shot is divided into a fixed number of equal size segments, each of which is represented by a blending image obtained by averaging of the pixel values of all the frames in a segment. Each blending image is divided into blocks of equal size, and then two color signatures are extracted per block. There're still other features such as luminance and its variants [10, 11, 12], domain color [13], etc.

2.3.4 Discussion

The color histogram signature uses only the color properties without using the spatial properties information. The performance is not good on movies like Star Wars. There are a number of shots in different parts of the movie with the same color scheme, which are distinguishable without the use of spatial information. This problem may be solved by partitioning a frame into blocks and gets histograms of each block, but from the section 2 of the ***"Introduction to Video Fingerprinting"***, we know that color properties of a frame (ex. luminance histogram and, hue, and saturation, etc.) are susceptible to color variations that are caused by different encoding devices. Methods using luminance, variants, coherent vector, principal components of the color histogram, and domain color also have this disadvantage of sensitivity.

The method proposed by Yang is more robust to kinds of image processing because it exploits both special and temporal properties of a video.

2.4 Changes in or between frames

The technique of this subsection exploits the relation between pixels in a frame (spatial distribution), change of pixel value in the same location or change of patch intensity in the same block between two consecutive frames (temporal distribution). The key factor, ***relation***, gives strong spatial and temporal information.

2.4.1 Motion direction

The motion-base signatures exploit the change information in the video [3]. Figure 2.3 shows a block diagram of the motion signature extraction process. The frames are partitioned into $N = N_x \times N_y$ blocks and motion vectors are estimated for each block. The direction of the motion vector for each block is quantized into Q directions or levels. A signature of any given frame is the number of motion vector contain in each of the Q levels. A level 0 is assigned to the block with zero amplitude of the motion vector. For example, if we are using 15x15 blocks per frame, and 4 directions for the motion vector quantization, the video signature at any time t will be:

$$\boldsymbol{S_m(t) = q_0(t), q_1(t), q_2(t), q_3(t), q_4(t)}$$
$$\boldsymbol{q_i\,(t) \in \{0, \ldots\ldots, 255\}},\ i=0,\ldots\ldots\ 4$$

with $\sum_{i=0}^{4} q_i\,(t) = 225$.

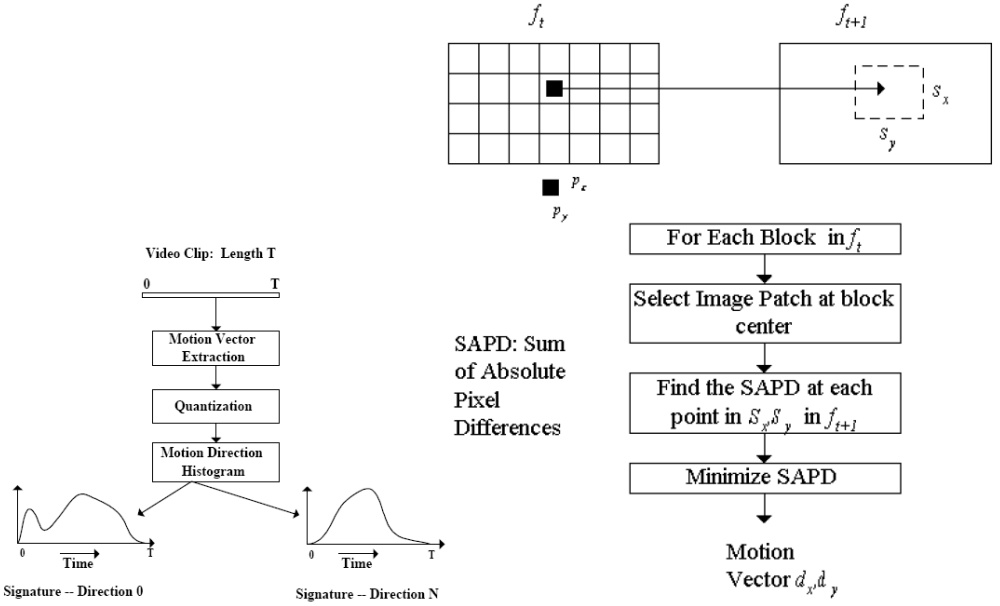


Figure 2.3: Left: Block diagram of motion signature extraction. Right: Block diagram of motion signature estimation.

To get the motion direction, a small intensity patch *P* is selected around the center of each block and a search neighborhood (S_x, S_y) is selected around the center of the corresponding block in the next frame. The intensity patch *P* is placed at all possible locations within the search neighborhood and the sum of absolute pixel difference (SAPD) is computed. The SAPD is used as a measure of patch similarity and the minimum SAPD is considered the match location and determines the motion direction. The ***normalized correlation coefficient*** is computed in the matching step and the time (frame index) with maximum quantity is selected as the best match.

2.4.2 Temporal measure

This method defines a global temporal activity *a*(*t*) depending on the intensity *I* of each pixel (N is the number of pixels for each image) [2].

$$\boldsymbol{a\,(t) = \sum_{i=1}^{N} K(i)(I(i,t) - I(i,t-1))^2} \qquad (2.3)$$

where *K(i)* is a weight function to enhance the importance of the central pixels. A signature is computed around each maxima of the temporal activity a(t). The spectral analysis by a classical FFT leads to a 16-dimensional vector based on the phase of the activity.

2.4.3 Centroids of gradient orientation (CGO)

This is also one of techniques we used in our project. It is proposed by Sunil Lee and Chang D. Yoo in [4]. ***The gradient orientation is the direction in which the directional derivative has the largest value.*** Figure 2.4 shows an overview of this method. First, an input video is resampled at a fixed frame rate F frames per second (fps) to cope with the frame rate change (typically F = 10 fps). Next, each resampled frame is converted to the gray scale and its width and height are normalized to the fixed values R_x and R_y (typically R_x = 320 and R_y = 240), respectively. These steps make the proposed fingerprints robust against variations in color characteristics and resizing. Then each resized frame is partitioned into $M = M_x \times M_y$ blocks (typically M_x= *4* and M_y= *2*), and the centroid of gradient orientation is calculated for each block. Finally, M-dimensional vector of the centroids is obtained and used as a fingerprint for the frame. For fingerprinting matching, a fingerprint sequence that consists of fingerprints extracted from K consecutive frames is sued (typically K =100 that corresponds to 10 seconds when F = 10 fps).

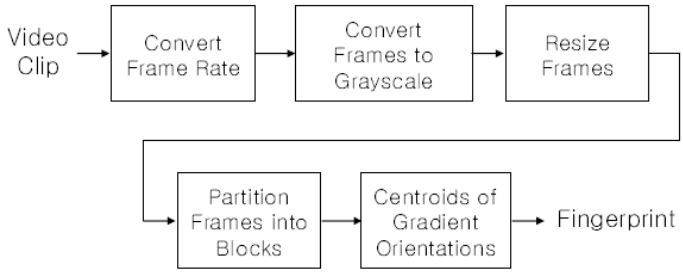


Figure 2.4: Overview of CGO.

Here I'll show the detail. For each luminance value $s(x, y)$ of a block S in a video frame, the gradient magnitude $m(x, y)$ and orientation $\theta(x, y)$ are calculated as follow:

$$\boldsymbol{m(x, y) = \sqrt{G_x^2 + G_y^2}} \qquad (2.4)$$

$$\boldsymbol{\theta(x, y) = tan^{-1}(G_y/G_x)} \qquad (2.5)$$

where the partial derivatives G_x and G_y are approximated by $G_x = s(x+1, y) - s(x-1, y)$ and $G_y = s(x, y+1) - s(x, y-1)$, be careful that G_y/G_x is likely to be 0/0 and causes an error. Then, the centroid of gradient orientations of the block S is calculated as follows:

$$\boldsymbol{c = \frac{\sum_{x=2}^{X-1}\sum_{y=2}^{Y-1}\theta(x,y)m(x,y)}{\sum_{x=2}^{X-1}\sum_{y=2}^{Y-1}m(x,y)}} \qquad (2.6)$$

where $X = R_x / M_x$ and $Y = R_y / M_y$. The value of CGO range from $-\pi/2$ to $\pi/2$ regardless of the location of the block. A vector of M CGOs is used as a fingerprint for

a frame, and a vector of *MK* CGOs extracted from *K* consecutive frames is used as a fingerprint sequence for fingerprint matching. The ***squared Euclidean distance*** is used in matching step.

2.4.4 Other techniques

Oostveen et al. in [22] presents the concept of video fingerprinting or hash function as a tool for video identification, they have proposed a spatio-temporal fingerprint based on the differential of luminance of partitioned grids in spatial and temporal regions.

2.4.5 Discussion

Motion signature captures the relative change in the intensities over time, thus it is immune to global color change; however, it discards spatial information and in [3], and the performance is worse than ordinal feature. Temporal measure is the weakest technique in subsection 2.4 since it also discards the spatial information, and the pixel change between frames is not reliable enough. CGO exploits the spatial relation between pixels and is stored in the order of block location and temporal location. It performs well against kinds of image processing other than histogram equalization.

2.5 Ordinal measure

The original measure is originally proposed by Bhat and Nayar [14] for computing image correspondences, and adapted by Mohan in [15] for video purposes.

2.5.1 Spatial-Intensity ordinal

The ordinal intensity signature [2, 3] consists in partitioning the image into *N* blocks; these blocks are stored using their average gray level and the signature *S*(*t*) use the rank r_i of each block *i*. Table 2.1 shows an example.

$$\boldsymbol{S\,(t) = (r_1, r_2, \ldots\ldots, r_N)}$$

20.3	12.9	123.2	1	0	5
250.1	72.3	199.2	8	3	6
69.3	80.2	200.0	2	4	7

Table 2.1: Left: Average gray level intensity values in a video frame divided in 9 blocks Right: Ranks of blocks based on intensity ordering.

The distance *D*(*t*) is defined for computing the similarity of two videos (a reference *R* and a test *T*) at a time code *t* when L_T is the length of the considered segment. It is computed by placing the test signature at different points along the reference and

computing the distance.

$$\boldsymbol{D\,(t) = \frac{1}{L_T}\sum_{i=t-L_T/2}^{t+L_T/2} |R(i) = T(i)|} \qquad (2.7)$$

The point t_{min} at which $D(t)$ is minimal is considered the best match between $R(t)$ and $T(t)$.

2.5.2 Ordinal histogram feature

This technique is presented in [5], and also executed in our project. The concept is similar with the one in 2.4.3 but the storing is different, only one vector for a video-scene clip. Color and spatial information is used, but no temporal information. First, a frame is partitioned into 4 blocks (can be changed), and the RGB components are separately measured. The averaged intensity is computed for each block and the order is stored.

From figure 2.5, a 4-dimensional vector is generated for each color component for each frame, and there will be 4! different rank combinations, then each color component of each frame will be classified into a 24-bin histogram. Finally, a 72-dimentional feature is generated (24 for each color component) for a video-scene clip.

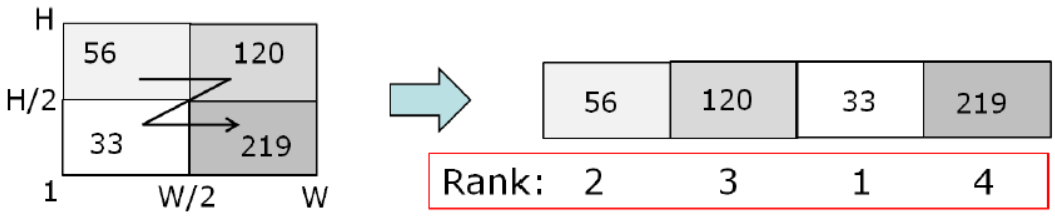


Figure 2.5: Rank vector computation for one color component

2.5.3 Temporal ordinal

Instead of using the rank of the regions in the image, the method proposed by L. Chen and F. Stentiford in [16] use the rank of regions along the time. If each frame is divided into K blocks and if λ^k is the ordinal measure of the region k in a temporal window with the length M, the dissimilarity D between a query video V_q and a reference video V_r at the time code t is:

$$\boldsymbol{D\,(V_q, V_r^p) = \frac{1}{K}\sum_{k=1}^{K} d^p(\lambda_q^k \lambda_r^k)} \qquad (2.8)$$

where:

$$\boldsymbol{d^p(\lambda_q^k,\, \lambda_r^k) = \frac{1}{C_M}\sum_{i=1}^{M} |\lambda_q^k(i) - \lambda_r^k(p+i-1)|} \qquad (2.9)$$

20.3	50.9	221.3	78.2	150.9	90.5	10.7	250.1	140.4

8	7	2	6	3	5	9	1	4

Table 2.2: Up: The average gray level of a block k and the length in temporal-axis is 9. Down: Temporal rank of block k based on intensity ordering.

p is the temporal shift tested and C_M is a normalizing factor. The best temporal shift p is selected. Table 2 shows an example.

2.5.4 Other techniques

Y. Li et al. in [17, 18] used a binary signature to represent each video - they merged color histogram with ordinal signatures for feature representation. Yuan et al. [19] also used a combination of ordinal histograms and cumulative color histograms for robust similarity search and copy detection. There still other studies used this ordinal measure [20, 21].

2.5.5 Discussion

From [3], the performance of ordinal intensity signature is compared with motion direction and color histogram and shows it dominants the other two techniques. It captures the relative distribution of intensities over space and time. Thus it is immune to global changes in the quality of the video that are introduced by the digitization/ encoding process. And from [20, 21], ordinal measure has been proved to be robust to changes of the frame rate and resolution, illumination shifts and display format.

The drawback of the ordinal measure is its lack of robustness as regards logo insertion, shifting or cropping, which are very frequent transformations in TV post-production. In fact, these transformations are also a big problem to all the global descriptors.

2.6 Transform-based

Techniques in this subsection use more complicated math formulas like Fourier transform to generate the signature.

2.6.1 Compact Fourier Mellin transform (CFMT)

This technique is one of the techniques we used in the project. This is a complicated technique, so in this subsection I just briefly introduce its concept, and I'll make a summary of Julien's thesis later in section 4.

The Fourier-Mellin Transform (FMT) has been studied extensively in the content of ***watermarking*** and ***invariant object recognition*** [5]. All these methods exploit the fact

that this transform generates a ***rotation***, ***translation***, and ***scale-invariant*** representation of the images. The FMT was first introduced in [24] and the implementation here is based on the fast approximation described in [25].

The classical FMT of a 2D function f, $T_f(k, v)$ is defined as:

$$\boldsymbol{T_f(k,v)} = \frac{\mathbf{1}}{\mathbf{2\pi}}\int_{\mathbf{0}}^{\infty}\int_{\mathbf{0}}^{\mathbf{2\pi}} \boldsymbol{f(r,\theta)r^{-iv}e^{-ik\theta}d\theta}\frac{\boldsymbol{dr}}{\boldsymbol{r}} \qquad (2.10)$$

Where (k, v) and (r, θ) are respectively the variables in Fourier-Mellin and polar domain representation of the function f. Ghorbel [26] suggested the concept of Analytical Fourier-Mellin Transform (AFMT), an important modification to the problem associated with the existence of standard FM integral (the presence of 1/r tern in the definition necessarily requires f to be proportional to r around the origin such that when r→ 0, then f→ 0). The AFMT, $T_{f,\sigma}(k, v)$, is defined as:

$$\boldsymbol{T_{f,\sigma}(k,v)} = \frac{\mathbf{1}}{\mathbf{2\pi}}\int_{\mathbf{0}}^{\infty}\int_{\mathbf{0}}^{\mathbf{2\pi}} \boldsymbol{f(r,\theta)\, r^{\sigma-iv}e^{-ik\theta}\, d\theta}\frac{\boldsymbol{dr}}{\boldsymbol{r}} \qquad (2.11)$$

where σ, a strictly positive parameter, determines the rate at which f tends toward zero near the origin.

Let $f_1(x, y)$ be an image and its rotated, scaled and translated version $f_2(x, y)$ is given by the equation:

$$\boldsymbol{f_2(x,y)} = \boldsymbol{f_1(\alpha(xcos\beta + ysin\beta) - x_0, \alpha(-xsin\beta + ycos\beta) - y_0)} \qquad (2.12)$$

where the rotation and scale parameters are β and α respectively, and $[x_0, y_0]$ is the translation. It can be shown that for rotated and scaled images, the magnitudes of the AFM transforms, $|T_{f_1,\sigma}|$ and $|T_{f_2,\sigma}|$, (corresponding to f_1 and f_2, respectively) are related by the equation:

$$\boldsymbol{\left|T_{f_2,\sigma}(k,v)\right| = \alpha^{-\sigma}|T_{f_1,\sigma}(k,v)|} \qquad (2.13)$$

An AFMT leads to a scale and rotation invariant representation after proper normalization by $1/\alpha^{-\sigma}$. Finally, the CFMT representation can be made translation invariant by computing the AMFT on the Fourier transformed image (considering only the magnitude part).

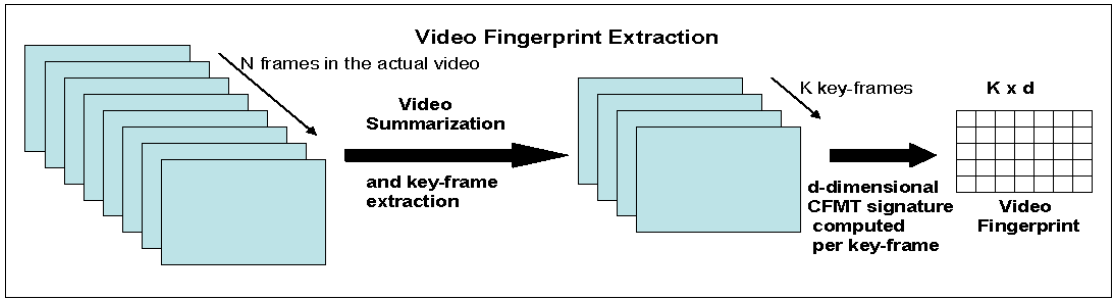


Figure 2.6: Generation of CFMT-Based Signature.

2.6.2 Other techniques

B. Coskun et al. in [27] proposed two robust hash algorithm sfor videos both based on the Discrete Cosine Transform (DCT) for identification of copies. And in [40], a technique based on the wavelets to find replica images on the web is presented.

2.6.3 Discussion

CFMT has best performance in [5] and also in our project, which is shown in ***"Introduction to Video Fingerprinting"***.

3. Local-descriptor-based signature

In contrast to considering all the pixels in a frame, local-descriptor-base signature only exploits features from and around the points of interest.

3.1 Introduction

Points-of-interest extraction is the additional procedure for this class of techniques. The well-know Harris and Stephens detector in [30] is a proposed method. Besides, a pertinent description of each interest point is necessary.

3.2 A. Joly method

This technique described in [29] is based on an improved version of the Harris interest point detector and a differential description of the local region around each interest point. To increase the compression, the feature are not extracted in every frame of the video but only in key-frames corresponding to extrema of the global intensity of motion [32]. The resulting local feature are 20-deimensional vectors in $[0,255]^{\mathrm{D}=20}$ and the mean rate is about 17 local features per second of video. Let $\vec{S}$ be one of the local features, defined as:

$$\vec{\boldsymbol{S}} = \left(\frac{\overrightarrow{\boldsymbol{s_1}}}{\|\overrightarrow{\boldsymbol{s_1}}\|}, \frac{\overrightarrow{\boldsymbol{s_2}}}{\|\overrightarrow{\boldsymbol{s_2}}\|}, \frac{\overrightarrow{\boldsymbol{s_3}}}{\|\overrightarrow{\boldsymbol{s_3}}\|}, \frac{\overrightarrow{\boldsymbol{s_4}}}{\|\overrightarrow{\boldsymbol{s_4}}\|}\right) \qquad (3.1)$$

where $\overrightarrow{s_i}$ correspond to 5-dimensional sub-vectors computed at 4 different spatio-temporal position distributed around the interest point. Each $\overrightarrow{s_i}$ is the differential decomposition of the gray level 2D signal $\vec{I}(x,y)$ up to the second order:

$$\vec{\boldsymbol{s_i}} = \left(\frac{\boldsymbol{\partial}\vec{\boldsymbol{I}}}{\boldsymbol{\partial x}}, \frac{\boldsymbol{\partial}\vec{\boldsymbol{I}}}{\boldsymbol{\partial x}}, \frac{\boldsymbol{\partial^2}\vec{\boldsymbol{I}}}{\boldsymbol{\partial x \partial y}}, \frac{\boldsymbol{\partial^2}\vec{\boldsymbol{I}}}{\boldsymbol{\partial x^2}}, \frac{\boldsymbol{\partial^2}\vec{\boldsymbol{I}}}{\boldsymbol{\partial y^2}}\right) \qquad (3.2)$$

3.3 Video Copy tracking (ViCopT)

ViCopT is the technique I started with for the project, it is briefly introduced in [1] and compared in [2], and in [33] the detail is presented. Harris points of interest are extracted on every frame and a signal descriptor similar to the one used in 3.2 is computed, leading to 20-dimesional signatures. The difference is that the differential decomposition of the gray level signal until order 2 is computed around 4 spatial positions around the interest point (in the same frame).

These points of interest are associated from frame to frame to build trajectories with an algorithm similar to the KLT [31], and the trajectory is used to characterize the spatio-temporal content of videos: it allows the local description to be enriched by adding a spatial, dynamic and temporal behavior of this point. For each trajectory, the signal description finally kept is the average of each component of the local description. By using the properties of the built trajectories, a label of behavior can be assigned to the corresponding local description as a temporal content. For CBCD two popular labels have been selected:

- label Background: motionless and persistent points along frames
- label Motion: moving and persistent points

The final signature for each trajectory is composed of 20-dimensional vector, trajectory properties and a label of behavior. More detail will be presented in section 4.

3.4 Space time interest points

This technique was developed by I. laptev and T. Linderberg in order to detect spatio-temporal events [35]. The space time interest points correspond to points where the image values have significant local variation in both space and time. Previous applications of this detector concerned classification of human actions and detection of periodic motion. And for CBCD, the detector has not been optimized for the task of copy detection and the presented results are preliminary. Space time interest points are described by the spatio-temporal third order local jet leading to a 34-dimensional vector:

$$\boldsymbol{j} = (\boldsymbol{L_x}, \boldsymbol{L_y}, \boldsymbol{L_t}, \boldsymbol{L_{xx}}, \boldsymbol{L_{ttt}}) \qquad (3.3)$$

where $L_{x^m y^n t^k}$ are spatio-temporal Gaussian derivatives normalized by the spatial detection scale σ and the temporal scale τ.

$$\boldsymbol{L_{x^m y^n t^k}} = \boldsymbol{\sigma^{m+n}\tau^k}\left(\boldsymbol{\partial_{x^m y^n t^k} g}\right) * \boldsymbol{f} \qquad (3.4)$$

The L_2 distance is used as a metric for the local signatures.

3.5 Scale invariant feature transform (SIFT)

SIFT descriptors were introduced in [34] as features that are relatively robust to scale, perspective, and rotation changes, and their accuracy in object recognition has led to extensive use in image similarity. Computationally however they are costly since characterization of an image using SIFT descriptors generally solicits thousands of features for a single image which must be pair-wise compared, meaning that an interest point in one image has to be compared with all the points from another image. It is not the dimensionality of the descriptor (typically 128) that bottlenecks SIFT image comparison, but the number of descriptor comparisons, so dimensionality reduction techniques such as PCA-SIFT [35] are insufficient for large datasets such as constitute video frames.

One way to speed the comparison is quantizing the descriptor to a finite ***"codebook"***. Then each image is represented as a weighted vector of the quantized descriptor (***"codeword"***) frequencies. The dimensionality of the signature is a tunable parameter, and can be thought of as the number of "codeword" in the codebook. With a large codebook, the codeword are more representative of the actual descriptors (less quantization error), and therefore in general a large codebook provides signatures that quantify similarity more accurately.

3.6 Discussion

In [28], the authors show that using local descriptors is better than ordinal measure for video identification when captions are inserted. When considering these kinds of image transformation, signatures based on points of interest have demonstrated their effectiveness for retrieving video sequences in very large databases, like in the approach proposed in subsection 3.2. Using such primitives is mainly motivated by the observation that they provide a compact representation of the image content while limiting the correlation of and redundancy between the detected features.

SIFT is a special case of local descriptor methods since the fingerprint of a key frame has combined all the local descriptors of the frame into a vector. This operation reduces computational complexity while reducing the accuracy.

4. Our works—Details

In UIUC, I followed Prof. Pierre Moulin for the project "Video Fingerprinting", and there are still two members for this topic, an exchange student, Julien Dubois, form Belgium and Ryan Rogowski, an junior student in UIUC. The goal of the project is to generate some kinds of technique and build a database for testing. First I started with key-frame extractions and the ViCopT. The key-frame extraction is needed for CFMT and SIFT, and I generated a MATLAB script for it. While on the research of ViCopT, I met some difficulties due to lacking of programming ability, knowledge of matching, and data mining. So finally I stopped this technique to generate the CGO and YCbCr methods.

We generated a database containing 2600 different video clips (each is 15 to 20-second long), and 100 of them are selected to build duplicated clips. We tried Gamma correction, JPEG compression, AWGN, blurring, and frame drops, and 63 duplicated video were generated for each clip then finally there are 8900 clips in the database. The technique we generated are SIFT, CFMT, CGO, YCbCr histogram, and ordinal histogram measure, while only 4 techniques except the CGO are tested before I left UIUC due to insufficient time.

4.1 Introduction

In section 4, I will discuss in depth about three complicated methods: ViCopT, CFMT, and SIFT.

4.2 Compact Fourier Mellin transform (CFMT)

In our project, the CMFT is applied to each of the 8 key frames extracted from a scene to compute a d-dimension signature vector. The total size of the fingerprint of a scene is $8 \times d$, and each element is represented with 8 bits. The parameter d trades-off compactness against retrieval accuracy. And for each key frame, we only work with gray-level image. I first derive the discrete AFMT form for discrete images, then explain how to obtain the compact fingerprint from the AFMT spectrum.

4.2.1 Fourier-Mellin transform and its adaptation

The classical FMT of a function f defined over $\mathbb{R}^2$ and expressed in polar coordinates r and θ is shown in equation (2.10):

$$\forall(\boldsymbol{k},\boldsymbol{v}) \in \mathbb{Z} \times \mathbb{R}: \quad \boldsymbol{T_f(k,v)} = \frac{\mathbf{1}}{\mathbf{2\pi}} \int_{\mathbf{0}}^{\infty} \int_{\mathbf{0}}^{\mathbf{2\pi}} \boldsymbol{f(r,\theta) r^{-iv} e^{-ik\theta} d\theta} \frac{\boldsymbol{dr}}{\boldsymbol{r}} \qquad (4.1)$$

In our case the function f is simply an image (the key frame), and the FMT is a global transform that is applied to all the pixels of the image.

We first need to define a center of coordinates in order to compute this transform, and here we always consider the center of an image for it. Now we focus on the double integral of equation (4.1). The fundamental condition for its existence is:

$$\int_0^{\infty}\int_0^{2\pi}|\boldsymbol{f(r,\theta)r^{-iv}e^{-ik\theta}}|\boldsymbol{d\theta}\,\frac{\boldsymbol{dr}}{\boldsymbol{r}} \leq \int_0^{\infty}\int_0^{2\pi}\frac{\boldsymbol{1}}{\boldsymbol{r}}\boldsymbol{f(r,\theta)d\theta dr} < \infty \qquad (4.2)$$

where the first inequality holds because *f* is positive. The term $1/r$ in the integral is a problem where r tends toward zero. Indeed equation (4.2) is satisfied only if *f* is proportional to Kr^{σ} around the origin of the domain, where K is a constant and $\sigma > 0$. It means that *f* should be equal to zero at the origin but like we said previously, *f* is an image and the origin has the value of the central pixel which is not necessary equal to zero. When f is not equal to zero at the origin, the integral diverges and the FMT does not exist.

For this reason, Ghorbel has introduced an adaption of the classical FMT called the Analytical Fourier-Mellin Transform (AFMT) in [25]. The AMFT is defined as:

$$\forall(\boldsymbol{k},\boldsymbol{v}) \in \mathbb{Z}\times\mathbb{R}:\quad \boldsymbol{T_{f,\sigma}(k,v)} = \frac{\boldsymbol{1}}{\boldsymbol{2\pi}}\int_0^{\infty}\int_0^{2\pi}\boldsymbol{f(r,\theta)}\,\boldsymbol{r^{\sigma-iv}e^{-ik\theta}}\,\boldsymbol{d\theta}\,\frac{\boldsymbol{dr}}{\boldsymbol{r}} \qquad (4.3)$$

where σ is a positive constant. This is the FMT of the function $\boldsymbol{f(r,\theta)r^{\sigma}}$. The general value of σ is 0.5 as suggested in [25]. This modification assures the convergence of the integral and thus the existence of AMFT. This approach is more efficient than canceling the function *f* over a small disc around the origin since the value of *f* around the origin contributes significantly to the FMT computation, owing the $1/r$ factor.

Now we'll see why the AMFT is rotation and scaled invariant. If we transform *f* using a scaling factor α and a rotation of an angle β. The AMFT of the function $g(r,\theta) = f(\alpha r, \theta+\beta)$ is given by:

$$\forall(\boldsymbol{k},\boldsymbol{v}) \in \mathbb{Z}\times\mathbb{R}:\quad \boldsymbol{T_{g,\sigma}(k,v)} = \frac{\boldsymbol{1}}{\boldsymbol{2\pi}}\int_0^{\infty}\int_0^{2\pi}\boldsymbol{f(\alpha r,\theta+\beta)}\,\boldsymbol{r^{\sigma-iv}e^{-ik\theta}}\,\boldsymbol{d\theta}\,\frac{\boldsymbol{dr}}{\boldsymbol{r}} \qquad (4.4)$$

This transform can also be expressed in the Cartesian domain:

$$\boldsymbol{g(x,y) = f(\alpha(xcos\beta + ysin\beta), \alpha(-xsin\beta + ycos\beta))} \qquad (4.5)$$

We operate the following change of variable:

$$\boldsymbol{\alpha r = r'} \quad \boldsymbol{\theta+\beta = \theta'} \qquad (4.6a)$$

$$\boldsymbol{dr} = \frac{\boldsymbol{dr}'}{\boldsymbol{\alpha}} \qquad \boldsymbol{d\theta} = \boldsymbol{d\theta}' \qquad (4.6b)$$

The AMFTs of g and f thus related by:

$$\forall(\boldsymbol{k},\boldsymbol{v}) \in \mathbb{Z} \times \mathbb{R}: \ \boldsymbol{T_{g,\sigma}(k,v)} = \boldsymbol{\alpha^{-\sigma+iv} e^{ik\beta}\, T_{f,\sigma}(k,v)} \qquad (4.7)$$

If we take the magnitude of $T_{g,\sigma}(k,v)$ and $T_{f,\sigma}(k,v)$, we obtain:

$$\forall(\boldsymbol{k},\boldsymbol{v}) \in \mathbb{Z} \times \mathbb{R}: \ \left|\boldsymbol{T_{g,\sigma}(k,v)}\right| = \boldsymbol{\alpha^{-\sigma}}\left|\boldsymbol{T_{f,\sigma}(k,v)}\right| \qquad (4.8)$$

This relation shows that the two magnitude spectra are related by a multiplicative constant ($\alpha^{-\sigma}$) that only depends on the scaling factor. If we can get rid of this constant using a normalization process, we can affirm that the magnitude of the AMFT spectrum is invariant to rotation and scaling. In practice we normalize the magnitude of the FMT spectrum by taking its logarithm and subtracting its mean value. In summary, the AMFT is a global feature invariant to geometric transformations. The adaptation form of the classical FMT assures the convergence of the integral.

4.2.2 Numerical approximation of the AFMT

The definitions presented in section 4.3.2 concern analytical functions and a discrete version of the AFMT does not exist. In this section, we explain numerical approximation that we use to compute the AFMT on discrete images.

We first consider a change of the integration variable of equation (4.3). We use the integration variable $t = ln(r)$ instead. We can thus write:

$$\boldsymbol{T_{f,\sigma}(k,v)} = \frac{\boldsymbol{1}}{\boldsymbol{2\pi}} \int_{-\infty}^{+\infty} \int_{\boldsymbol{0}}^{\boldsymbol{2\pi}} \boldsymbol{e^{t\sigma} f(e^t,\theta)\, e^{-i(k\theta+tv)}\, d\theta dt} \qquad (4.9)$$

This relation shows that AMTF of $f(r,\theta)$ can be seen as the Fourier transform of the deformed image $e^{t\sigma}\, f(e^t,\theta)$. This observation is the key idea of the numerical approximation used in this our project and called the ***fast AFMT approximation***. The fast AFMT approximation was introduced in [25] and this is the approximation used in [5]. It is computed in three steps:

1. Resampling of the discrete image on a log-polar grid
2. Computation of the deformed image $e^{t\sigma}\, f(e^t,\theta)$
3. Numerical integration using the Discrete Fourier Transform

Let's study in more detail about each of these steps. Equation (4.9) is for analytical

function f so we need to adapt if for discrete signal. We have a key frame $f(p,q)$ defined on a discrete rectangular grid and we would like to resample this key frame on a log-polar grid to obtain the image $f(e^{\rho n},\theta_m)$ with $n \in [0,N-1]$ and $m \in [0,M-1]$. The size of the new grid is $M \times N$ and we work with a size of 100×100 in our project. The log-polar grid is defined as the intersection between M concentric circle and N radial lines originating from the origin. It is similar to a polar grid except that the radii of different circles are not linearly spaced but logarithmically spaced. We start with a radius $r = 1$ and the biggest radius is r_{max} which is defined as the distance between the central pixel and one of the four pixels in the corner of the key frame. If the key frame has width W and height H, then:

$$\boldsymbol{r_{max} = \sqrt{(\frac{W}{2})^2+(\frac{H}{2})^2}} \qquad (4.10)$$

Equivalently r_{max} is the radius of the smallest disc that can contain the frame. We space 100 circle radii between $r = 1$ and r_{max}. It is also means that we forget the central pixel of the image during the resampling operation. The variation of angle $\Delta\theta$ between two lines of the grid is linear and is equal to $2\pi/N = 2\pi/100$. Figure 4.1 shows a log-polar grid of size 30×30.

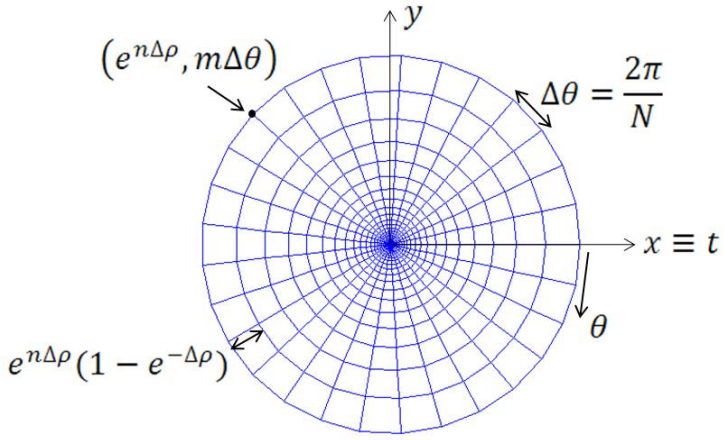
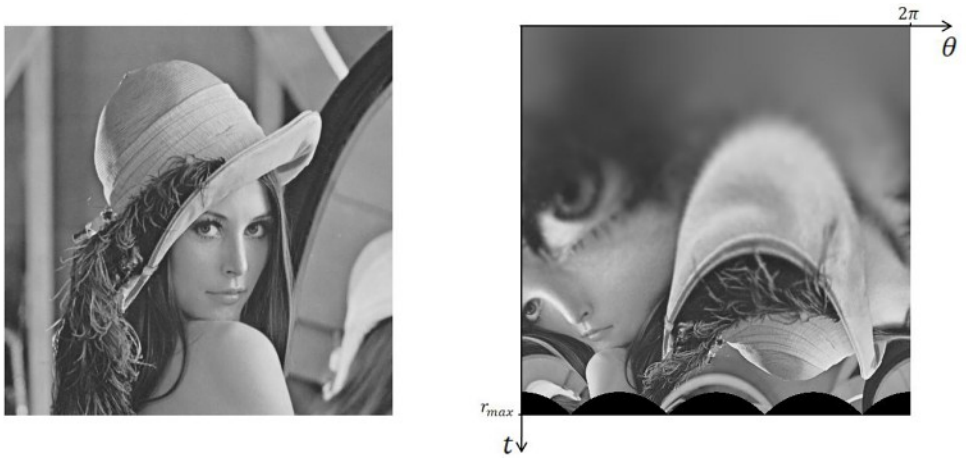


Figure 4.1: Log-polar grid of dimension 30×30

Once the log-polar grid is built, we need to resample the key frames on that grid. We can make two comments about this operation. First the pixels of the key frames are initially on a discrete rectangular grid $f(p,q)$ and do not exactly match the points on the log-polar grid. We do a cubic interpolation to find the resampled key frame. We call it $\hat{f}(e^{\hat{\rho}n},\hat{\theta}_m)$ to clearly make the distinction with the idea case $f(e^{\rho n},\theta_m)$. The interpolation makes the computation of the AFMT imperfect and this explains why we call it fast AFMT approximation. Secondly the shape of the

log-polar grid is circular and the shape of the frame is rectangular. It means that some points on the log-polar grid do not match with pixels on the frame. We give them the value zero. Figure 4.2 shows the classical “Lena” picture after resampling with a log-polar grid.



(a) Original picture (512 × 512 pixels) (b) Resampled picture using a512 × 512 log-polar grid

Figure 4.2: "Lena" picture resampled on a log-polar grid

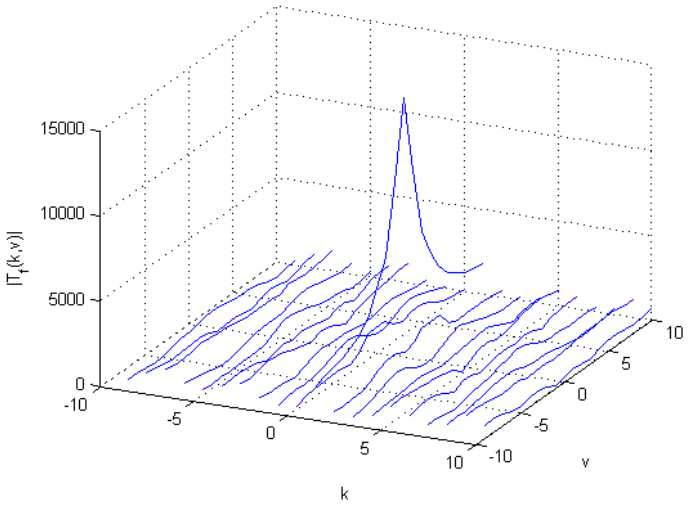


Figure 4.3: AFMT spectrum magnitude of the "Lena" picture using a 20 × 20 log-polar grid

Once the key frame is resampled, we do a numerical integration to compute the AFMT. The formula presented in [25] is equation (4.11). The numerical integration also explains why we only obtain an approximation of the AFMT.

$$\forall \boldsymbol{k} \in [-\boldsymbol{K}, \boldsymbol{K}], \forall \boldsymbol{v} \in [-\boldsymbol{V}, \boldsymbol{V}]: \quad \widehat{\boldsymbol{T}}_{\boldsymbol{f},\boldsymbol{\sigma}}(\boldsymbol{k},\boldsymbol{v}) =_{\Delta} \boldsymbol{\theta} \sum_{\boldsymbol{n}=\boldsymbol{0}}^{\boldsymbol{N}} \boldsymbol{e}^{\widehat{\boldsymbol{\rho}}\boldsymbol{n}} \sum_{\boldsymbol{m}=\boldsymbol{0}}^{\boldsymbol{M}-\boldsymbol{1}} \widehat{\boldsymbol{f}}(\boldsymbol{e}^{\widehat{\boldsymbol{\rho}}\boldsymbol{n}}, \widehat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \boldsymbol{e}^{-\boldsymbol{i}(\boldsymbol{k}\widehat{\boldsymbol{\theta}}_{\boldsymbol{m}}+\boldsymbol{v}\widehat{\boldsymbol{\rho}}\boldsymbol{n})} \tag{4.11}$$

In practical we first compute $e^{\hat{\rho}n}\hat{f}(e^{\hat{\rho}n}, \hat{\theta}_m)$ with a simple matrix multiplication and then we compute the Discrete Fourier Transform (DFT) using the default function of MATLAB. We finally take the magnitude part of the spectrum to obtain an estimation of the AFMT. The spectrum has the same dimensions that the log-polar grid (100×100). Figure 4.3 shows the AFMT spectrum of the "Lena" picture with a log-polar grid of 20×20.

This spectrum is not very accurate because the size of the log-polar grid is much smaller than the size of the original picture (512×512). However the goal is just to illustrate the global shape of an AFMT spectrum. We notice that the components of the spectrum have the largest value at $k = 0$ and $v = 0$. This observation also applies to the AFMT spectrum computed in [25], which validates our results. Finally we verified that the spectra computed with our implementation are scale and rotation invariant.

4.2.3 Building the fingerprints

Now, each key frame has a spectrum composed of $P = M \times N = 100 \times 100$ coefficients. Nevertheless, our initial objective was to compute a compact *d*-dimensional signature vector for each key frame, so we need to drastically reduce the dimension of the spectrum representation. Principal component analysis (PCA) [36] is an efficient way to compress the data.

Here we treat the AFMT spectrum as a *P*-dimensional vector, and a training set of Q key frames is used to find a low *d*-dimensional basis ($d \ll Q$). We want to find the best ***d orthonormal basis vectors*** to represent the *P*-dimensional vector under the criterion to minimize the approximation error. The final fingerprint of a key frame is composed of ***d approximately linear-combinational weights*** (***d projective weights on each basis vector***) of the corresponding AFMT spectrum. In our project, $d = 2,4,6,8,10,12,20,36$ are used for comparison.

Lloyd-Max nonuniform quantization is the last step in the CFMT fingerprint computation process. It consists of quantizing the projection coefficients d_i on 2^8 levels, and this quantization is done with a *k*-means clustering using Lloyd algorithm.

4.3 Scale invariant feature transform (SIFT)

In our project, SIFT features are also based on 8 key frames extracted from a scene. In each key frame, a set of local SIFT descriptors needs located and computed. I'll first present the SIFT algorithm and then explain how to compute a compact *d*-dimensional signature for each key frame from the set of SIFT descriptors.

4.3.1 SIFT algorithm

The algorithm to compute the SIFT descriptors was published by David Lowe in 1999 and this subsection is inspired by his reference paper [34]. The MATLAB-based implementation is done using ***vl_seaft*** [37], a powerful open source library.

Local SIFT descriptors are 128-dimensional vectors invariant to scale and rotation, so they can be used to reliably match two key frames. The process to extract a set of descriptors is composed of four different stages:

1. Scale-space extrema detection
2. Key point localization
3. Orientation assignment
4. Key point descriptor computation

Now I'll start to explain each of the four stages, while some of them are a little bit complicated and I'll just give brief introduction and list the reference.

- **Scale-space extrema detection:** We want not only the content of descriptors is invariant to rotation and scaling, but also the location of descriptors. The first stage consists of detection potential points in the frame that are invariant to scale change. The idea is to ***"use a continuous function of scale (also known as scale space) to search across all possible scale"*** [34]. This function is a variable-scale Gaussian:

$$\boldsymbol{G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} exp(\frac{-(x^2+y^2)}{2\sigma^2})} \qquad (4.12)$$

  The method proposed by D. Lowe involves computing scale-space extrema of the Difference-of-Gaussian function (DOG):

$$\boldsymbol{D(x,y,\sigma) = \big(G(x,y,k\sigma) - G(x,y,\sigma)\big) * I(x,y)} \qquad (4.13)$$

$$\boldsymbol{= L(x,y,k\sigma) - L(x,y,\sigma)} \qquad (4.14)$$

  where $I(x,y)$ is the input key frame, and $L(x,y,\sigma)$ is the function that defines the scale space of an image. The idea is to convolve the input key frame with the difference of two Gaussian functions separated by a multiplicative constant k. This function can be efficiently computed by image subtraction. The maxima and minima of $D(x,y,\sigma)$ give potentially stable locations in the scale space (i.e. points that are scale invariant).

  Extracting the extrema points is based on a multi-resolution and multi-scale framework [34].

- **Key point localization:** The goal of this stage is to ***"perform a detailed fit to the nearby data for location, scale and ratio of principal curvatures"*** [34].In other words, we have too many candidate key points and some of them are unstable. The idea is to reject key points that have low contrast or a bad location along an edge of the picture because these points are more sensitive to degradations like noise.
- **Orientation assignment:** At this point we can find stable key point locations and scales (***the scale where a key point is found in the multi-scale framework, this quantity is used to make the gradient scale-invariant***). This stage is to assign one or several orientations to each key point based on local image properties. The goal is to make the key point descriptors invariant to rotation because they are computed relatively to these orientations.

  For each key point, the Gaussian smooth image L with the closet scale to the key point scale is chosen and local gradient orientations are computed in a small region around the key point using pixel differences. Then an orientation histogram is built. The peaks of this histogram correspond to the dominant directions of local gradient and give the orientations of each key point. We usually find one significant peak in the histogram and thus one orientation for each key point but in some cases (about 15% of the time) there exist several significant peaks (typically two). In that case, one descriptor is computed for each orientation although the key location and scale stay the same.

  Figure 4.4 show the 725 key points computed from the "Lena" picture using the default parameter of the ***vl_seaft*** implementation. The position of each circle center indicates a key point location, its size is proportional to the key point scale, and its radius direction gives the key point orientation.
- **Key point descriptor computation:** Now that each key point is clearly defined with a key frame ***location***, a ***scale*** and an ***orientation***, and now each key point descriptor can be represented as a 128-dimensional vector. The descriptor is based on the computation of the gradient orientation and magnitude in a region around the corresponding key point. The scale of the key point indicates which Gaussian blur image L we have to choose. First of all, the region around the key point is weighted by a circular Gaussian function to increase the weight of gradients near the key point location. The size of the region is a tunable parameter that depends on a magnification factor. The scale of the key point is multiplied by this factor to obtain the width (in pixels) of the region. The value of the magnification factor use in our project is 12.

  Then the region is divided into 4×4 subregions. The gradient orientation and magnitude of all pixels in each subregion are computed and then

accumulated into 16 orientation histograms. Each histogram has 8 possible orientations. The gradient orientation and the orientation histogram are evaluated relative to the key point orientation to be rotation invariant. Figure 4.5 shows an example for 2×2 subregions, The length of the arrows is proportional to the gradient magnitude. The blue circle represents the Gaussian window.

Figure 4.4: Stable key point locations and scales of the "Lena" picture

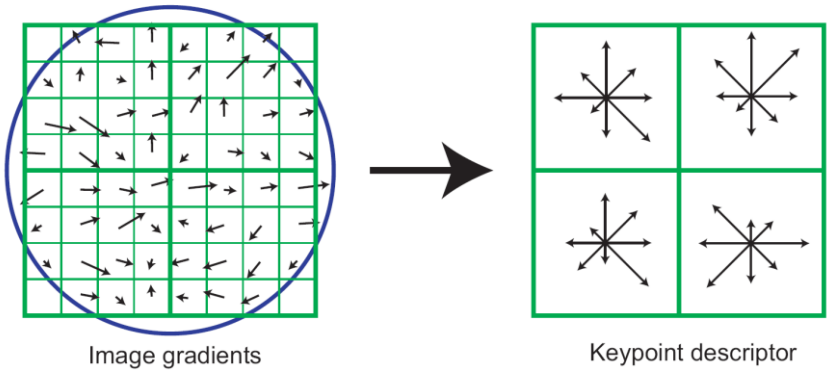


Figure 4.5: Illustration of local image gradients and orientation histograms

Figure 4.6 shows the convention used in ***vl_seaft*** implementation for the numbering order of the different subregion and the directions of the histogram. This is useful to represent the value of the histogram as a 128-dimensional vector. We have 16 subregions and each of term is represented by a 8-bin histogram (8 orientations). The descriptor vector finally contains the values of the different histograms. Figure 4.7 shows a descriptor computation on the “Lena” picture.

We can observe the 16 orientation histograms. This descriptor is shown to be robust against change in illumination or 3D viewpoint which makes SIFT appropriate for applications like object recognition.

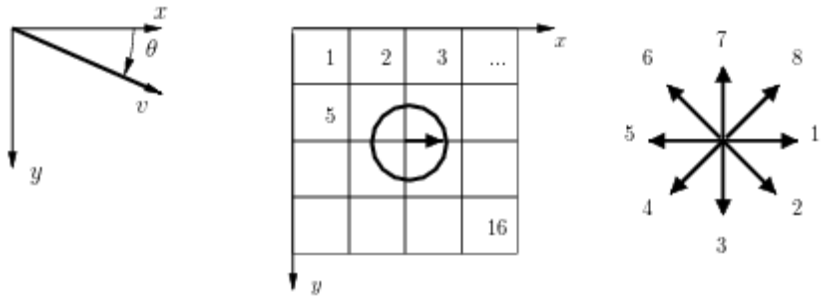


Figure 4.6: Convention for the descriptor computation

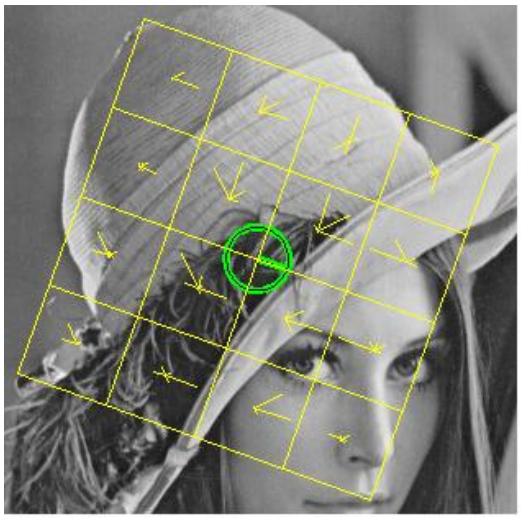


Figure 4.7: Example of one descriptor computation

4.3.2 Descriptor quantization

Each key frame averagely contains 650 descriptors and each of them is a 128-dimensional vector. Then the signature of each key frame could thus be a matrix of 650×128 coefficients in average. From subsection 3.5, this is unacceptable due to the cost of matching, so we need to reduce the dimension of the extracted features and compute a compact *d*-dimensional signature vector for each key frame.

The idea is to compute a set of ***d symbolic descriptors*** using a big training database of descriptors. The value of *d* is tunable and we tried 6,12,24,36,120,320. The *k*-means clustering algorithm is used to classify descriptors of the database into *d* clusters, and the mean vector (we called them cluster center) of each cluster is defined as a symbolic vector. These vectors are saved and re-used from one fingerprint computation to another.

Once d symbolic 128-dimensional descriptors are computed, we save them and quantize each descriptor of the key frame to its closet symbolic descriptor (Euclidean distance criterion). Let's consider that we have computed d symbolic descriptors $(D_1, D_2, \dots\dots, D_d)$ after the training phase and that the number of descriptors computed for a particular key frame is N. After quantization, a certain fraction N_i of these N descriptors is associated to D_i. The sum of the N_i values is obviously equal to N. The key frame signature is simply the d-dimensional vector $[N_1, N_2, \dots\dots, N_d]$ depending on the number of descriptors N.

Implementation is realized with a function of [37] that uses the Elkan algorithm. This algorithm drastically accelerates the training process compared to the classical Lloyd algorithm as presented in [39] (speedup of a factor 20 for d = 120 for instance). Additional information about k-means clustering can be found in [38] and [39]. We finally give the size of the database for each value of D in table 4.1. We notice that we used fewer descriptors in the database for d = 320 because of memory limitation with our MATLAB implementation.

d	number of descriptors
6	1,700,000
12	1,700,000
18	1,700,000
24	1,700,000
36	1,700,000
120	1,700,000
320	680,000

Table 4.1: Size of the training database for the SIFT feature k-means clustering

4.4 ViCopT

ViCopT consists of extracting the dynamic behavior on the local descriptions of interest points and further on the estimation of their trajectories along the video sequence. Analyzing the low-level description obtained allows to highlight trends of behaviors and then to assign a label of behavior to each local descriptor.

The fingerprint of each video is a set of trajectories. And each trajectory is described with its ***range in both space and time***, ***average values of all the local descriptors along the trajectory***, and ***a corresponding label defined for application***. This technique is composed of several steps:

Off-line operation:

1. Extracting points of interest
2. Estimating trajectories
3. Defining labels

On-line operation:

1. Asymmetric Technique
2. Searching and voting function

where the off-line operation means the signature building of videos in database, and the on-line operation is for the test video and matching. The main reference is [33]. In subsections 4.4.1 and 4.4.2, off-line indexing operation is presented, and the efficient retrieval algorithm for on-line matching is in subsection 4.4.3 and 4.4.4. Finally in subsection 4.4.5, I'll explain some easily-confusing points.

4.4.1 Building trajectories of local descriptors

- **Extracting and characterizing points of interest:**
  In this step, a logarithm for finding points of interest and a description for each point is required. The well-know Harris and Stephens detector [30] is used for finding points. The SIFT method described provided in subsection 4.3.1 is also an local descriptor, while for each point the high 128-dimensional description makes it incompatible for ViCopT. After extracting interest points by The Harris detector, a local description of points leads to the following 20-dimensional signatures $\vec{S}$:

$$\vec{S} = \left(\frac{\vec{s_1}}{\|\vec{s_1}\|}, \frac{\vec{s_2}}{\|\vec{s_2}\|}, \frac{\vec{s_3}}{\|\vec{s_3}\|}, \frac{\vec{s_4}}{\|\vec{s_4}\|}\right) \qquad (4.1)$$

  where $\vec{s_i}$ correspond to 5-dimensional sub-vectors computed at 4 different spatio position distributed around the interest point. Each $\vec{s_i}$ is the differential decomposition of the gray level 2D signal $\vec{I}(x, y)$ up to the second order:

$$\vec{s_i} = \left(\frac{\partial \vec{I}}{\partial x}, \frac{\partial \vec{I}}{\partial x}, \frac{\partial^2 \vec{I}}{\partial x \partial y}, \frac{\partial^2 \vec{I}}{\partial x^2}, \frac{\partial^2 \vec{I}}{\partial y^2}\right) \qquad (4.15)$$

  The Gaussian filter is used for computing the derivatives in order to reduce the noise. This feature space is called $\boldsymbol{S_{Harris}}$ in this subsection.

- **Tracking points of interest:**
  Trajectories of interest points are usually analyzed for modeling the variability of points along the video and then enhancing their robustness. There have been several proposed temporal approaches of feature point tracking for point trajectory estimation. The most popular approach is probably the Kanade-Lucas-Tomasi (KLT) tracker [31]. To obtain low-cost computational

techniques, the tracking algorithm chosen is basic and does not depend on the local description adopted. A L_2 distance is computed in $\boldsymbol{S_{Harris}}$ from frame to frame between all the local descriptors of the frame and all of those from q previous frames and the q next frames and for points that match, three decision can be taken: start of a new trajectory, end of a trajectory, add the point to an existing trajectory. q is tunable and in [33] q = 15 is suggested.

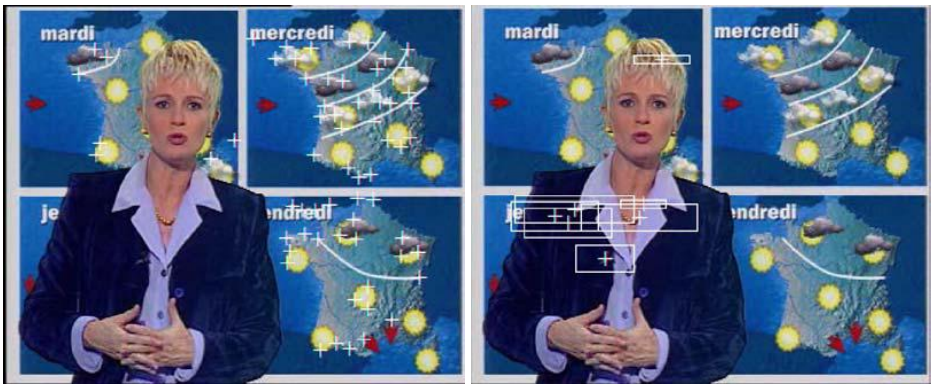
4.4.2 Labeling behavior of points

In this subsection, a high-level description of the set of videos is made.

- **Signal description:**
  For each trajectory, the average of each component of the local descriptors in $\boldsymbol{S_{Harris}}$ is taken as a low-level description $\vec{\boldsymbol{S}}_{\boldsymbol{mean}}$ of the trajectory. Although the local signatures may very along the trajectory, 95% of the points of the trajectories have a distance from $\vec{\boldsymbol{S}}_{\boldsymbol{mean}}$ lower than the matching threshold used during the trajectory building, which makes $\vec{\boldsymbol{S}}_{\boldsymbol{mean}}$ suitable to characterize a trajectory. The signal description space of $\vec{\boldsymbol{S}}_{\boldsymbol{mean}}$ is called $\boldsymbol{S_{Signal}}$ conventionally.
- **Trajectory description:**
  Here the geometric and kinematic behavior of the interest points along a trajectory is considered. The following trajectory parameters are stored during the off-line indexing step:

  - ✦ Time code of the beginning and of the end: $[t_{Cin}, t_{Cout}]$
  - ✦ Variation of the spatial position: $[x^{min}, x^{max}], [y^{min}, y^{max}]$

  In addition to these parameters, the mean local descriptors $\vec{\boldsymbol{S}}_{\boldsymbol{mean}}$ of $\boldsymbol{S_{Signal}}$, associated to the trajectories, provides a richer description of the video content. The feature space will be called the trajectory parameters $\boldsymbol{S_{Traj}}$.
- **Definition of labels:**
  Now we can define trends of behaviors by thresholding $\boldsymbol{S_{Traj}}$. This is a high level description because involving an interpretation of the video content. In [33], labels Background (motionless and persistent points along frames) and label Motion (moving and persistent points) are used and the voting algorithm of labels is introduced in [41]. Figure 4.8 shows an example of labeling.



Label *Background* / Label *Motion*

Fig 4.8: Illustration of labels of behavior dedicated to Copy Detection. The boxes represent the amplitude of moving points along their trajectory (motionless points do not have a box). The ”+” are the mean position of such points.

4.4.3 An Asymmetric technique

Starting from this subsection, the algorithm of retrieval is presented. This method uses $\boldsymbol{S_{Signal}}$ for selecting candidates, and then uses the $\boldsymbol{S_{Traj}}$ and the labels computed during the off-line indexing part for taking a decision. This voting function is robust to outliers and is the key of the whole retrieval system. This spatial-temporal robust registration is the main contribution of ViCopT.

In this subsection, we'll talk about the asymmetric technique for matching. The off-line operation described in previous two subsections needs long time computation, while the on-line retrieval is in real-time, so the same process can't be used for test videos. Consequently, the retrieval approach is so asymmetric. For a test video, the features are $\boldsymbol{S_{Harris}}$ of ***local interest points*** selected depending on two parameters:

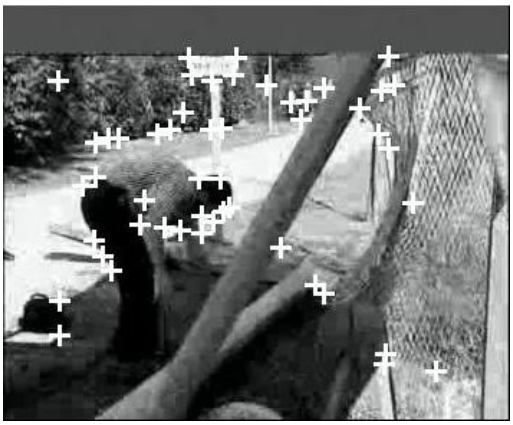
- period p of chosen frame in the video stream
- number n of chosen points per selected frame

The advantage of the asymmetric technique is that the number of local points and the temporal precision can be chosen on-line, which gives more flexibility to the system. The main challenge is that a set of $\boldsymbol{S_{Harris}}$ is the signature of a test video while for videos in database, the signature of each video is a set of $\boldsymbol{S_{Traj}}$ and $\boldsymbol{S_{Signal}}$. Figure 4.9 illustrates the registration challenge with the cross on the query points on the left and the representation of the $\boldsymbol{S_{Traj}}$ space on the right.

4.4.4 Other steps

There're still three steps, ***similarity searching***, ***spatio-temporal registration***, and

combination of labels, required to finish the copy detection process. These steps make ViCopT a robust fingerprinting technique. I'll just make a stop here since the algorithms and concepts behind them are hard to explain. Readers who are interested in this technique could find details in [33].



Test video

The + represent the queries

(Points of interest)

Original video in the database

The boxes represent S_{Traj}

$[x^{min}, x^{max}], [y^{min}, y^{max}]$

Figure 4.9: Illustration of the feature spaces involved in the asymmetric method.

4.4.5 Some easily-confusing points

In the report ***"Introduction to Video Fingerprinting"***, I use the term ***"candidate"*** as a possible match in the database of a test video, while in [33], this term is used for possible fragment of the testing video which may contain copy sequences. The main difference comes from different test video benchmarks, system frameworks and purposes in each proposed paper. Figure 4.10 shows the framework of ViCopT and the term "candidate" is used for test videos.

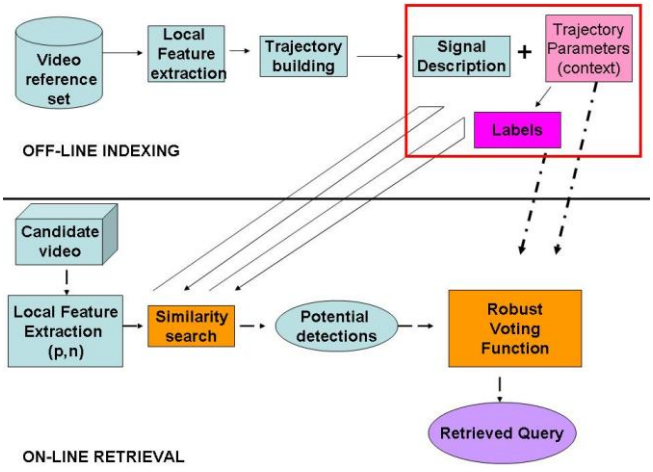


Figure 4.10: Video Copy Detection Framework from [33]

In general, the videos in the database are complete movies, TV programs or sport games, etc., while the test video may have several variations. The whole test video can be a fragment of one of the video in database, or just a part of the test video does. But in [33], the test video benchmark shown in figure 4.11 where the test video combines several fragments coming from different video sources make the detection more complicated.

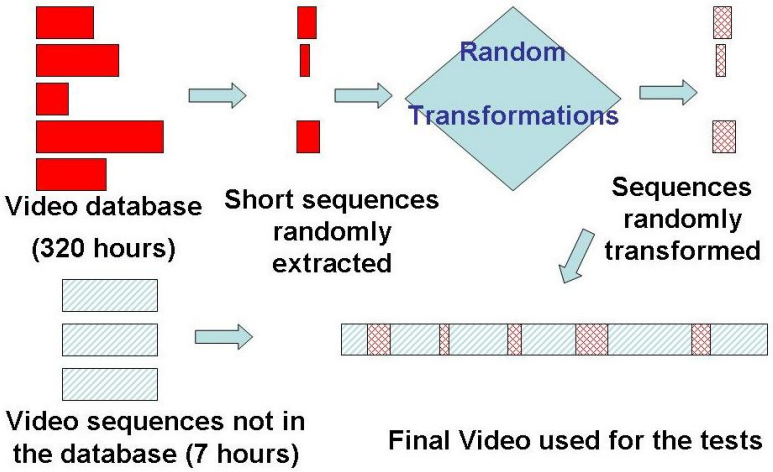


Figure 4.11: Video Benchmark Building from [33]

5. Conclusion

This report introduces nearly all CBCD techniques I have read. Wish that this work can gives readers more understanding about video copy detection.

Reference

[1] J. Law-To, O. Buisson, V. Gouet-Brunet, and N. Boujemaa. Video copy detection on the Internet: the challenges of copyright and multiplicity. In ICME'07: Proceedings of the IEEE International Conference on Multimedia and Expo, pages 2082_2085, 2007.

[2] 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 CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval, pages 371_378, New York, NY, USA, 2007. ACM.

[3] A. Hampapur, K. Hyun, and R. M. Bolle. Comparison of sequence matching techniques for video copy detection. volume 4676, pages 194_201. SPIE, 2001.

[4] Sunil Lee and Chang D Yoo. Video fingerprinting based on centroids of gradients orientations. In ICASSP '06: Proceedings of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 401_404, Washington, DC, USA, 2006. IEEE Computer Society.

[5] A. Sarkar, P. Ghosh, E. Moxley, and B. S. Manjunath. Video fingerprinting: Features for duplicate and similar video detection and query-based video retrieval. In SPIE – Multimedia Content Access: Algorithms and Systems II, Jan 2008.

[6] P. Indyk, G. Iyengar, and N. Shivakumar, "Finding pirated video sequences on the internet," Technical report,Stanford University, 1999.

[7] M. Y. M. Naphade and B.-L. Yeo, "A novel scheme for fast and efficient video sequence matching using compact signatures.," in *Proc. SPIE, Storage and Retrieval for Media Databases 2000*, Vol. 3972, pp. 564–572, Jan. 2000.

[8] C. K. R. Lienhart and W. Effelsberg, "On the detection and recognition of television commercials," in *Proc. of the IEEE Conf. on Multimedia Computing and Systems*, 1997.

[9] J. V. J. M. Sanchez, X. Binefa and P. Radeva., "Local color analysis for scene break detection applied to TV commercials recognition.," in *Proceedings of Visual 99*, pp. 237–244, June 1999.

[10] Changick Kim, "Spatio-Temporal Sequence Matching for efficient video copy detection," SPIE Storage and Retrieval for Media Databases 2004, Jan. 2004.

[11] Job Oostveen, Ton Kalker, and Jaap Haitsma, "Feature Extraction and a Database Strategy for Video Fingerprinting", in *Proc. International Conference on Recent Advances in Visual Information Systems*, pp. 117-128, 2002.

[12] Xian-Sheng HUA, Xian CHEN, Hong-Jiang ZHANG, "Robust Video Signature Based on Ordinal Measure,", in *Proc. International Conference on Image Processing (ICIP 2004*), vol. 1, pp. 685-688, October 24-27, Singapore, 2004.

[13] Arun Hampapur and Rudolf M. Bolle, "VideoGREP: Video Copy Detection using Inverted File Indices," Technical Report, IBM Research, 2001.

[14] D. Bhat and S. Nayar, "Ordinal measures for image correspondence," *PAMI*, 1998.

[15] R. Mohan, "Video sequence matching," in *ICASSP*, 1998.

[16] L. Chen and F. W. M. Stentiford. Video sequence matching based on temporal ordinal measurement. Technical report no. 1, UCL Adastral Park, 2006.

[17] Y. Li, L. Jin, and X. Zhou. Video matching using binary signature. In Int. Symposium on Intelligent Signal Processing and Communication Systems, pages 317–320, 2005.

[18] Y. Li, J. S. Jin, and X. Zhou, "Matching commercial clips from TV streams using a unique, robust and compact signature," in Proc. of Digital Image Computing: Technqiues and Applications, 2005.

[19] J. Yuan, L. Y. Duan, Q. Tian, S. Ranganath, and C. Xu, "Fast and robust short video clip search for copy detection," in Springer: Lecture Notes in Computer Science - 3332, pp. 479–488, 2004.

[20] X-S Hua, X Chen, and H-J Zhang, "Robust video signature based on ordinal measure," in *ICIP*, 2004.

[21] C. Kim and B. Vasudev, "Spatiotemporal sequence matching techniques for video copy detection," *Trans. on circuits and systems for video technology*, 2005.

[22] J. Oostveen, T. Kalker, and J. Haitsma. Feature extraction and a database strategy for video fingerprinting. In VISUAL '02: Int. Conf. on Recent Advances in Visual Information Systems, UK, 2002.

[23] X. Yang, Q. Tian, and E. C. Chang, “A color fingerprint of video shot for content identification,” in Proceedings of the 12th annual ACM international conference on Multimedia Systems, pp. 276–279, 2004.

[24] D. Casasent and D. Psaltis, “Scale invariant optical transform,” Opt.Eng. 15(3), pp. 258–261, 1976.

[25] S. Derrode and F. Ghorbel, “Robust and efficient Fourier-Mellin transform approximations for gray-level image reconstruction and complete invariant description,” Computer Vision and Image Understanding: CVIU 83(1), pp. 57–78, 2001.

[26] F. Ghorbel, “A complete invariant description for gray-level images by the harmonic analysis approach,” in Pattern Recognition Letters, 15, pp. 1043–1051, October 1994.

[27] B. Coskun, B. Sankur, and N. Memon. Spatio-temporal transform-based video hashing. Ieee Transactions on Multimedia, 8(6):1190–1208, 2006.

[28] K. Iwamoto, E. Kasutani, and A. Yamada, “Image signature robust to caption superimposition for video sequence identification,” in *ICIP*, 2006.

[29] A. Joly, O. Buisson, and C. Frelicot. Content-based copy detection using distortion-based probabilistic similarity search. ieee Transactions on Multimedia, 2007.

[30] C. Harris and M. Stevens. A combined corner and edge detector. In 4th Alvey Vision Conf., pages 153–158, 1988.

[31] C. Tomasi and T. Kanade, “Detection and tracking of point features,” Tech. Report CMU-CS-91-132, 1991.

[32] S. Eickeler and S. M¨uller. Content-based video indexing of tv broadcast news using hidden markov models. In ICASSP, 1999.

[33] J. Law-To, O. Buisson, V. Gouet-Brunet, and N. Boujemaa. Robust voting algorithm based on labels of behavior for video copy detection. In ACM Multimedia, MM’06, 2006.

[34] D. Lowe, “Distinctive image features from scale-invariant keypoints,” in International Journal of Computer Vision, 20, pp. 91–110, 2003.

[35] Y. Ke and R. Sukthankar, “PCA-SIFT: A more distinctive representation for local image descriptors,” 2004.

[36] J. Shlens. Tutorial on principal component analysis, 2005.

[37] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[38] B. Gosselin. Statistical pattern recognition: Lecture notes (Faculté Polytechnique de Mons), 2008.

[39] C. Tomasi and T. Kanade, “Detection and tracking of point features,” Tech. Report CMU-CS-91-132, 1991.

[40] E. Chang, J. Wang, C. Li, and G. Wilderhold. Rime – a replicated image detector for the world-wide web. In *SPIE Symp. of Voice, Video and data communications*, pages 58–67, 1998.

[41] J. Law-To, V. Gouet-Brunet, O. Buisson, and N. Boujemaa. Local Behaviours Labelling for Content Based Video Copy Detection. In *International Conference Pattern Recognition*, 2006.