Telefonica Research Content-Based Copy Detection TRECVID Submission

Xavier Anguera, Pere Obrador, Tomasz Adamek, David Marimon and Nuria Oliver
Telefonica Research,
Via Augusta 177, 08027, Barcelona, Spain
{xanguera, pere, tomasz, marimon, nuriao}@tid.es

Abstract

This notebook paper presents the systems presented by Telefonica Research within the MESH team for the task of Video copy detection in TRECVID 2009. We participated in the Video-only, Audio-only and Audio+Video tasks. Our main contribution is the combination (when possible) of audio and video features within the same system by using global features extracted both from the reference videos and the queries. We also experimented with SIFT-based search methods and are aiming at building a hybrid search system. This is our first participation year and results are far from optimal, but some of them indicate the potential of the presented systems.

I. Introduction

In today's digital world, efficient multimedia data management tools are a need, in order to organize and search the vast amounts of video content that not only is generated daily but in many cases is made available to the public. Increasing storage capabilities at low prices enable the archival of most of this multimedia content, including professional TV broadcasts and internet video services (*e.g.*, Web-TV, video blogs, video sharing sites, public video archives, etc.) which contain both professionally generated videos and user generated content (UGC).

The detection of video duplicates in a video database is one of the key technologies in multimedia management. Its main applications include: (a) storage optimization; (b) copyright enforcement; (c) improved web search and (d) concept tracking. In storage optimization, the goal is to eliminate exact duplicate videos from a database and to replace them with links to a single copy or, alternatively, link together similar videos (near duplicate) for fast retrieval and enhanced navigation [**?**]. Copyright enforcement strives at avoiding the use and sharing of illegal copies of a copyright protected video. In the context of web search, the goal is to increase novelty in the video search result list, by eliminating copies of the same video that may clutter the results and hinder users from finding the desired content [**?**]. Finally, concept tracking in video feeds [**?**] focuses on finding the relationship between segments in several video feeds, in order to understand the temporal evolution of, for example, news stories.

Traditionally, watermarking techniques have been used to detect copies in images, audio or video [**?**], [**?**]. These techniques insert information (watermarks) into the media that is not noticeable by the user and that is later used to proof the authenticity of the media. One limitation of this approach is that the watermarks need to be included when generating the material, which is typically not the case in UGC. Alternatively, Content-Based Copy Detection (CBCD) techniques do not add any watermarks into the media, as the *media itself* is the watermark.

CBCD techniques analyze the content in order to extract a set of features that are then used to compare it with putative duplicates. In this paper, we focus on the detection of Near-Duplicate Video Clips (NDVC) via a novel multimodal CBCD algorithm.

The application of CBCD algorithms to videos has been usually approached from a video analysis perspective, by extracting and comparing features from the videos. Local features, typically computed at a key-frame level [**?**], [**?**], achieve good performance at high computational cost. Conversely, researchers have also investigated the use of global features ([**?**], [**?**]), extracted from low level features in the video, to conform a *fingerprint* or signature of the video. Video fingerprints allow for fast video comparisons at the expense, however, of lower performance.

This year's TRECVID evaluation on the task of video copy detection has been our first participation. For it we propose two different systems: on the one hand, a purely multimodal approach which takes global features as input for a cross-correlation-based algorithm that fuses both audio and video streams to find segments in the query that might appear in the reference videos with a certain probability. We call this algorithm Change-Based Copy Detection as it only takes into account the changes within the considered streams. On the other hand we present

results of an approach using SIFT local features and our own variant of a search engine based on Hierarchical Vocabularies of Visual Words. The SIFT-based algorithm was working over keyframes extracted with a simple shot boundary detection algorithm, for the task of video-only copy detection.

II. TRECVID PROPOSED SYSTEMS

For our first participation in the TRECVID evaluation for the task of video copy detection we have prepared two different systems for the tasks of video-only, audio+video and audio-only copy detection. For the video-only task we submitted two systems, a cross-correlation based system using global features and a SIFT-Based system. For the audio only task we presented the same cross-correlation based system, but this time only based on audio global features. Finally, for the multimodal task we presented the cross-correlation based system using both audio and video features. Next we describe the feature extraction modules for each one of the used features and the two different systems description.

The following summarizes the main characteristics of each or our submissions:

- **MESH.v.*.xcorr**: Video only submission using global features and a cross-correlation based algorithm
- **MESH.v.*.xcorrTime**: Video only submission using global features and a cross-correlation based algorithm. Instead of outputing the algorithm's matching time, all reference video's time was considered to match.
- **MESH.v.*.sift**: Video only submission using local features.
- **MESH.v.*.siftNoTime**: Video only submission using local features. Same time trick as above
- **MESH.a.*.official**: Audio-only submission using a cross-correlation based algorithm
- **MESH.m.*.multimodal**: Audio+Video submission using a cross.correlation based algorithm with multimodal features.

III. VIDEO COPY DETECTION FRONT-END

A. Audio Global Signature Extraction

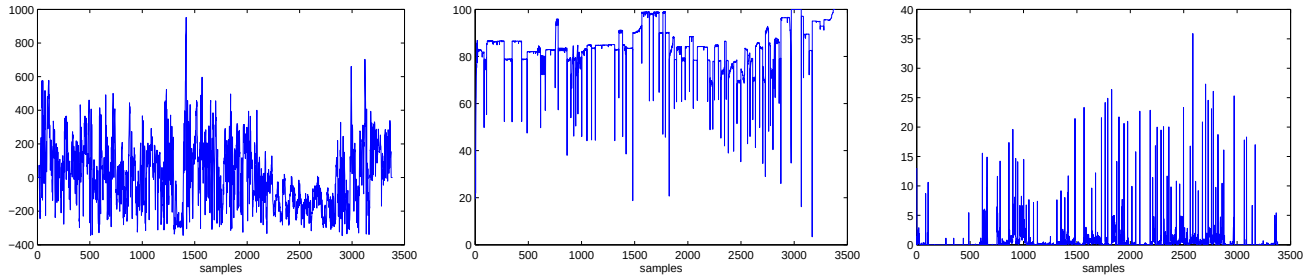
The global audio signature is extracted by analyzing the acoustic changes that take place in the audio track of the video. A one dimensional signal is created from the audio track whose minima correspond to places where there is a change in the speaker or in the sounds being recorded.

First, video and audio tracks are separated into independent streams. The audio track is down-sampled to 16KHz, 16 bits/sample and converted to 24 Mel Frequency Cepstrum coefficients (MFCC), including cepstrum derivatives, acceleration and energy, computed every 10 milliseconds. These features are typically used in speech processing and acoustic segmentation techniques. In particular, we use the Bayesian Information Criterion (BIC) [**?**] as the basis to obtain a measure of how similar the audio is in both sides of the analysis frame (and hence whether there is a change in the audio signal or not).

In our present work, BIC is used as proposed by [**?**] for acoustic segmentation by computing the acoustic metric for frame i as:

$$\begin{aligned} \Delta BIC(i) = \log L(X[i-W, i+W]|M_{ab}) \\ - \log L(X[i-W, i]|M_a) - \log L(X[i, i+W]|M_b) \end{aligned} \tag{1}$$

where $X[i, j]$ is an n-dimensional sequence corresponding to the MFCC features extracted from the audio segment from time $t = i$ to $t = j$; $M_{a,b,ab}$ are three Gaussian Mixture Models (GMM) composed of a weighted sum of Gaussian Mixtures, such that the number of Gaussians in M_a and M_b is the same and half of the number of Gaussians in M_{ab}; and W corresponds to the analysis window, set to 100 MFCC frames (one second of audio). The ΔBIC acoustic metric is computed every 100 milliseconds in a sliding window along the input signal, obtaining an acoustic change signature with 10 points for every second of video, which accounts for a 0.16kbps signature stream. Figure **??**a shows an example of the ΔBIC acoustic change pattern extracted for a 173 second long video. The local maxima in the plot represent places where the surrounding audio is maximally similar. Conversely, the local minima represent acoustic change points; small changes within the same acoustic condition (positive value) or large changes from one acoustic condition to another (negative value).



a) Audio variation b) Hue and Luminance c) Highlights Centroid variation

Fig. 1. *Examples of the proposed signatures for a* 173 *sec long video.*

B. Video Global Signatures Extraction

The global video signatures are extracted by considering the temporal visual changes occurring in the video. Three features are extracted (hue, luminance and highlights centroid variation) from which two signatures are generated that are fed to the fusion module, where they are combined with the audio signature, in order to make the final comparison with the reference video.

1) Picture-in-Picture Extraction Algorithm: Picture in picture (PIP) videos are processed in a two pass manner: first, analyze the whole video for possible PIPs; second, run the analysis algorithm on the video, either on the full video if no PIP was detected, on the full video except for the PIP region, or only on the PIP region. In this section we will describe the PIP detection part.

Only a subset of frames are analyzed for possible PIPs, i.e., the video is temporally down-sampled in order to improve detection time. This jeopardizes the detection temporal accuracy, of course. In order to detect the rectangular regions that form our PIP we first apply a Canny edge detector to the video frame being analyzed. We follow this by a probabilistic Hough transform, since it is more efficient if the picture contains a few long linear segments, which returns line segments rather than whole lines.

In a blank, image *lineIm(n)* for frame n, all the detected lines that are either completely vertical or completely horizontal are plotted. Each frame generated this way is accumulated on an intermediate image, *im(n): im (n)= (im(n-1) +lineIm(n))* which is later normalized to *im(n) = Im(n) * 254/max(Im(n))* This image is then projected on both axes, generating a line density function which may be used to identify the temporally strong vertical and horizontal lines.

In order to reduce the false alarm rate, we analyze this projections only in the most likely segments to include a PIP (we have specific segments for each of the 4 corner PIPs and also for the central one, based on the trecVid 2008 data). Each of the projections (horizontal and vertical) is then analyzed within those likely segments for the peak (max) in the density function and its position. The noise floor is also calculated within that segment.

In this way we can calculate a *PeakToNoiseRatio*, which is one of the two parameters used for PIP detection below: *PeakToNoiseRatio = max(segment)/ noiseFloor* Each of these peaks defines either a vertical column or a horizontal row in the *im(n)* image. These columns/rows are analyzed for their center of gravity, which will further confirm to which PIP quadrant or center they belong. This is calculated as a percentage of either the width or height of the video frame. The final condition, which must be met by the 4 lines composing the PIP box, in order to have positive PIP detection is:

PeakToNoiseRatio ¿ threshold1 AND abs(centroid - Q) ¡ threshold2

Where Q is either 25%, 50% or 75% of the width or height, depending on which PIP position is being analyzed. For each analyzed frame, the coordinates of the detected PIP box are saved, otherwise a NO_DETECTION code is output.

2) Hue and Luminance Variation (HLV): The first descriptor of video variation consists of the change in color and luminance, in a similar way as the audio signature. This approach has been proposed in the past by Hoad and Zobel [**?**], where they considered all color channels (YCrCb) in order to detect video variations. Note that direct comparison of video signals is avoided by using the *change* in color and luminance, instead of the color itself. The authors show that only two channels (Hue and Value) are needed in order to detect near duplicates. Therefore, we first convert the input video signal to the HSV (Hue, Saturation, Value) color space. Next, we compute the Hue

and Value histograms at each frame H_{x_i} and compare them with the histograms from the previous frame, using a simple histogram intersection measure:

$$I(H_{x_i}, H_{x_{i-1}}) = \frac{\sum_{l=1}^{M} min(H_{x_i}(l), H_{x_{i-1}}(l))}{\sum_{l=1}^{M} H_{x_i}(l)} \quad (2)$$

where M is the number of bins in the histograms. Note that $I = 100\%$ implies no change at all, *i.e.*, exactly the same video frame, and $I = 0\%$ implies maximum change, *i.e.*, in checkerboard type images, changing from white to black.

The comparison is performed on a frame by frame basis without any temporal down-sampling. Therefore, this technique allows to track typical consumer camera variations that expand over multiple frames:

- Auto white balance, which adapts to the light source illuminating the scene (*i.e.*, so that white will be rendered white independently of the illuminant);
- Auto contrast and auto levels, which adapt to the amount of light in the scene and its distribution.

Figure **??**b depicts an example of the combined Hue and Value change features for a 173 sec long video. Sudden drops in this histogram intersection plot indicate scene changes, while ramps indicate fade-in/outs or other progressive changes.

3) Highlights and Shadows Centroid Variation(HCV and SCV): This descriptor tracks the highlights (areas of high luminosity) in the image, since the motion of high luminosity areas should be very similar between a video and a transformed version of it. This feature was first proposed by [**?**] in conjunction with a shadow centroid variation feature (areas of low luminosity).

We describe the Highlight Centroid Variation below. The Shadow Centroid Variation would be calculated in a similar way, by analyzing at the dark pixels in the video frame.

The image is first down-sampled by a factor of 8, both in the horizontal and vertical dimensions. A histogram of the Value channel is computed and the top 10% pixels in the histogram (*i.e.*, brightest pixels) are used to calculate the highlight centroid. The centroid is normalized to the diagonal of the frame, in order to be resilient to size change degradations in any of the axes (both horizontal and/or vertical). The centroid of the highlight region is then compared to the centroid of the highlight region from the previous frame. The change in the centroid position from frame to frame is calculated in absolute value, in order to be resilient to the horizontal flip of the entire (or part of the) video clip[1].

Therefore, this descriptor (see example in Figure **??**c) tracks the movement of the highlight centroid from frame to frame in absolute numbers. If the highlight region remains static from frame to frame, this will result in zero absolute change; if it pans slowly, say from left to right, it will generate a roughly constant value; and if there is a scene cut, it will typically generate a high peak at the frame where the cut takes place.

C. Audio/Video Global Signatures Postprocessing

Once the audio and video signatures have been extracted, they are post-processed in order to obtain signals that are better suited to be used by the matching algorithms presented below. A different processing is applied, depending on the signal involved.

The audio signature contains most of the fingerprinting information in the low frequencies, with a high frequency component which is basically noise. Therefore, a low pass filter is applied using a moving average window with width of four frames, centered in each frame. The same filtering is applied to the highlights centroid variation (HCV) signature.

Conversely, the hue and luminance variation feature (HLV) contains the fingerprinting information in the peaks of the signal, indicating the points where a change of scene happens. In this case, the a derivative filter is applied (after computing the absolute value of the signals) using the following regression formula:

$$x_{filtered}[n] = \frac{\sum_{i=1}^{\Theta} i(x[n+i] - x[n-i])}{2\sum_{i=1}^{\Theta} i^2} \quad (3)$$

[1]Described as one of the transformation types in the MUSCLE-VCD dataset.

where $x[n]$ is the hue or luminance variation feature at frame n and $\Theta = 2$. This filtering does a windowed differential sum for each point where the further the samples (in time), the larger their weight.

All feature streams are then normalized to zero mean and unity variance. Finally, as will be seen in the next Section, the number of samples extracted per second in the video channel is larger than in the audio channel. Therefore, the audio signature is warped to fit the video signature by means of a linear warping.

D. Local SIFT Features extraction

In the submitted run using local features these were extracted using Scale Invariant Feature Transform (SIFT) [**?**]. However it should be noted that the approach can provide similar or better performance when used with other local feature extractors such as for example Speeded Up Robust Features (SURF) [**?**].

In order to extract local features from selected keyframes both in the reference and query videos we built a simple (yet pretty effective) shot-boundary detection algorithm. The algorithm works as follows:

1) Retrieve/compute all three available image/based features from the video being considered, normalize them to unit variance and zero mean and compute their average into a single signal.
2) Apply a threshold find all peaks in the averaged signal, ensuring that there is at least a distance of five values (equivalent to 5 video frames, or $1/5$ second) between two peaks.
3) Extract a keyframe from the central location in each segment comprised between two adjacent peaks. If the segment length is greater than 10 seconds, extract several keyframes split along the segment.

Once the keyframes are extracted, the SIFT features are computed using the VLFeat open source library[2]. In all experiments we used the SIFT extraction tool with default values of all input parameters.

IV. Query Matching Algorithms

In this section, we describe the two query matching algorithms used in the Trecvid evaluation.

A. Cross-correlation based video copy detection

The cross-correlation based video copy detection algorithm allows finding sequences inside the query that match a given video in the reference database. The result of the algorithm is a start-end times in the query and reference videos with possible match, and their matching score.

Once the audio and video signatures have been extracted from the videos to be compared[3], the query signal is split into windowed segments of size $T = 1500$ video frames, sliding along the signal every $S = 300$ video frames. Each segment is compared to the entire queried video using each available feature stream independently by means of the GCC-PHAT measure. The GCC-PHAT implements a frequency domain weighting of the cross-correlation between two signals, given by:

$$R^{GCC}_{x_1,x_2}(m) = \mathcal{F}^{-1}(X_1(w) \cdot X_2^*(w) \cdot \psi_{PHAT}(w)) \tag{4}$$

where $X_1(w)$ and $X_2(w)$ are the signals being compared in frequency domain (usually converted from time domain using Fast Fourier Transform, FFT); $\mathcal{F}^{-1}$ is the inverse of the FFT and $\psi(w)$ is a weighting function:

$$\psi_{PHAT}(w) = \frac{1}{||X_1(w) \cdot X_2^*(w)||} \tag{5}$$

where $||\cdot||$ indicates the modulus. The GCC-PHAT returns a normalized cross-correlation plot (with values between 0 and 1) with a maximum at m_{max}, *i.e.,* the delay for which both signals are most similar.

The GCC-PHAT metric is computed for each of the audio and video signature pairs from the two videos, obtaining a cross-correlation plot for each of them. The cross-correlation measures are then fused together into one cross-correlation plot. Then, the n-best ($n = 3$ in the system presented) maxima of the fused cross-correlation plot are found and the associated n-best delays are saved into a delays histogram. Such histogram counts how many times the same delay has been found across all windowed segments. The use of a delays histogram follows the

[2]http://www.vlfeat.org

[3]Note that this is typically done offline for all videos in the system.

rationale that the delays defining segment matches should remain constant throughout the entire segment, ending with a peak in the histogram, whereas non-matching segments should return random delays. Note that these delays are stored in the histogram in *absolute* terms – in video frames, with respect to the beginning of the query signal.

Next, the best alignment delays are determined from the delay histogram by finding the peak count (max_count) and selecting all delays with counts within [max_count-1, max_count] range. For each one of these delays, the query and queried video windowed segments –in each of the available modalities– are retrieved and the weighted dot-product is computed as follows:

$$D_{v_1,v_2}(m_{\max}) = \sum_{i=1}^{N} W_i \cdot \frac{x_1^i \cdot x_2^i[m_{\max}]}{|x_1^i||x_2^i[m_{\max}]|} \tag{6}$$

where $W_i$indicates an *a priory* weight assigned to each available signature i ($i = 1..N$); $x_2^i[m_{\max}]$ is the signature in the query video, delayed to the optimum match delay; and $|\cdot|$ is the $L2$-norm of each signature vector: $|x| = \sqrt{\sum_n x^2[n]}$.

In order to automatically define the appropriate weight for each modality we introduce the use of the entropy. The definition of the entropy is

$$Entropy(X) = -\sum(x[i] \cdot \ln(x[i]) \tag{7}$$

Where $X = x[1] \dots x[N]$ is an unidimensional sequence of probabilities (with values from 0 to 1). The entropy indicates the amount of information present in the signal, i.e. how uncertain is the random variable that generates the signal. High values of entropy indicate low uncertainty, and vice versa. In our particular application we are interested in measuring how well a modality can determine that a segment x_1^i is similar to x_2^i and discriminate it from any other segment. For this purpose we consider that a modality that has an overall low entropy will contain much more discriminatory information that will lead to a more trustful score than higher entropy modalities. We then formulate the weights as

$$w_i = \frac{1}{\max(entropy(x_1^{i'}), entropy(x_1^{i'}))} \tag{8}$$

we take for each pair of compared segments the inverse of the segment with maximum entropy as the lower bound discriminability capability of that modality in those segments. Note that before computing the entropy of the features we need to normalize them between 0 and 1 and use these values as probabilities. We do this by subtracting the minimum value of the segment to each point and dividing by the dynamic range of the signal.

Finally, among all obtained scores we select the highest score (and associated matching segments) as the output for that query-video pair.

B. SIFT-based Matching Algorithm

The video copy detection algorithm relying on local features operates in two stages. In the first stage all query keyframes are compared to all reference keyframes resulting in one ranked list of relevant reference keyframes for each query keyframe. In the second stage all ranked lists produced for keyframes from one query video are combined using a very simple voting mechanism as described below.

The first stage relies on a scalable search engine heavily inspired by the state of the art approaches to object matching based on Vocabulary Trees [**?**], [**?**]. These approaches mimic text-retrieval systems by quantizing key-point descriptors by k-means clustering creating the so- called Vocabularies of Visual Words. In other words, in an off-line process a large number of descriptor examples are clustered into the Vocabulary of Visual Words, that defines a quantization of the descriptor space. From this moment every key-point can be represented by its mapping to the closest Visual Word. Once the dictionary is created all reference images are represented using an inverted file structure that stores all occurrences of every visual word in all reference images. Typically the scoring is based on the vector-space model (TF-IDF scoring) followed by a validation of spatial consistency between matched visual words. It should be noted that in our case, the TF-IDF scoring is replaced by our own solution. In all experiments we used a relatively small hierarchical vocabulary of 10K visual words (4 levels and branching 4) created by clustering SIFT features from 9K randomly selected reference keyframes.

In the second stage, for each keyframe of a particular query we accumulate the top keyframes in the ranked list and classify them in their videos of origin. The video that contains more matches keyframes at the end is the one chosen as possible copy. We obtain a score by normalizing the number of matches by the number of keyframes in the query. In order to return a start-end time for the match we find the maximum and minimum times from all matched keyframes in the selected video. As we were not sure this method could give us accurate matching segments (and tests with TRECVID 08 were ot conclusive) we also submitted an alternate output where the whole reference video is returned as a match.

V. TRECVID Participation Results

In the following we describe the particular configurations of the different systems submitted to the evaluation and our analysis on their performances

For the video-only submission we submitted runs with the two different systems (local feature-based and global feature-based) as described above. For the global feature-based system we ran the algorithm with all (three) available image features described in this paper. In none of the systems did we use an exhaustive method to tune the optimum submission threshold. Also, for this years' evaluation we have not considered any modification from the NOFA to the BALANCED conditions except for a slightly higher threshold in the NOFA condition.

We submitted 4 outputs to this condition, 2 using global features and 2 using local features (each of them with NOFA and BALANCED required submissions). Both in global and local features' submissions we submitted a run where the reference matching segment corresponded to the actual segment returned by the system and a run where the reference matching segment covered all the reference video's time. By far, our best results were obtained with the cross-correlation system using the exact reference matching time as obtained by the algorithm. Results are aligned or slightly better than the median for this task except for query 2 and 10 (where we performed much better than the average) and query 6 where our system underperformed.

Results for the local features-based algorithm are not as good as expected. After a first analysis of the possible reasons we tend to think that most matches for any keyframe with text (or not) in it were returning random keyframes with text, jeopardizing the results.

For the audio-only submission we submitted only runs using the cross-correlation based system with the same exact configuration as with the video-only submission, except that in this case we used the 3 audio features explained above. Results of this run are not good at all as we made the mistake of not including the 2009 reference set in the test.

Finally, for the audio-video submission we were planning on submitting an algorithm that combined both local and global features but did not have time to come up with the right solution. In fact, in order to turn the results in reasonably on time (we were finally late just a few hours) we had to trim the cross-correlation based algorithm to only use one video feature (highlights centroid) and one audio feature (XBIC metric). We also implemented a few more speedups in the code that must have caused a bug in the system as results have not been as expected.

In overall, we believe strongly that change-based features are a very interesting addition to local features as they are very robust to transformations in the media. On the other hand, our current implementation of the cross-correlation based system is very slow computationally, which jeopardizes its scalability (which we felt strongly when having to process almost 10K videos in the a+v condition).

We are working on speeding up the cross-correlation based algorithm and on combining the local and global features for a more effective video copy detection.

References

[1] S. Poullot, M. Crucianu, and O. Buisson, "Scalable mining of large video databases using copy detection," in *Proc. ACM Multimedia*, 2008.

[2] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," in *Proc. ACM Multimedia*, 2007.

[3] Y. Zhai and M. Shah, "Tracking news stories across different sources," in *Proc. ACM Multimedia*, 2005.

[4] I. Cox, J. Kilian, F. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[5] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proceedings IEEE, Special Issue on Identification and Protection of Multimedia Information*, vol. 87, no. 7, pp. 1079–1107, 1999.

[6] S. Poullot, M. Crucianu, and O. Buisson, "Fast content-based mining of web 2.0 videos," in *Proc. PCM, also in LNCS 5353*, 2008.

[7] H.-K. Tan, X. Wu, C.-W. Ngo, and W.-L. Zhao, "Accelerating near-duplicate video matching by combining visual similarity and alignment distortion," in *Proc. ACM Multimedia*, 2008.
[8] T. Hoad and J.Zobel, "Detection of video sequences using compact signatures," *ACM Transactions on Information Systems*, vol. 24, no. 1, pp. 1–50, January 2006.
[9] A. Hampapur, K.-H. Hyun, and R. Bolle, "Comparison of sequence matching techniques for video copy detection," in *Proc. Conf. on Storage and Retrieval for Media Databases*, 2002.
[10] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, pp. 461–464, 1978.
[11] J. Ajmera, I. McCowan, and H. Bourlard, "Robust speaker change detection," *IEEE Signal Processing Letters*, vol. 11, no. 8, pp. 649–651, 2004.
[12] D. Lowe, "Distinctive image features from scale-invariant keypoints," vol. 60, no. 2, 2004, pp. 91–110.
[13] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *Proc. European Conference on Computer Vision (ECCV)*, May 2006.
[14] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Proc. ICCV*, 2003.
[15] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.