# NTNU-Academia Sinica at TRECVID 2010 Content Based Copy Detection

Mei-Chen Yeh
Dept. of Computer Science and Information Engineering,
Nat'l Taiwan Normal Univ., Taiwan
myeh@csie.ntnu.edu.tw

Chao-Yung Hsu
Institute of Information Science,
Academia Sinica, Taiwan, ROC
cyhsu@iis.sinica.edu.tw

Chun-Shien Lu
Institute of Information Science,
Academia Sinica, Taiwan, ROC
lcs@iis.sinica.edu.tw

## ABSTRACT

This paper presents two video copy detection systems built for the TRECVID 2010 content-based copy detection task. Three runs were submitted using video-only content. Two systems differ in terms of the feature design as well as the matching scheme. In this paper we overview the underlying methodologies and discuss the various design choices for developing a practical video copy detection system.

## Keywords

Content-based copy detection, video matching.

## 1. INTRODUCTION

With increasing bandwidth available to average users and the exploding popularity of social media, digital video availability has grown exponentially through the use of online distribution technologies such as web-TV, video blogs, and video sharing websites. To manage video contents and to protect intellectual properties, Content-based Copy Detection (CCD) techniques provide an alternative approach to watermarking for identifying video sequences from the same source. Based on content alone, CCD attempts to identify segments in a query video that are *copies* from a reference video database. A copy is not an exact duplicate but, in general, either a transformed or a modified version of the original document that remains recognizable [2][1]. Transformations to digital content such as cropping and inserting logos are frequently performed and the resulting near-duplicates could be different from the source in terms of not only formats, but also content [7].

TRECVID (TREC Video Retrieval Evaluation) is sponsored by the National Institute of Standards and Technology (NIST) with the goal of encouraging research in information retrieval by providing a large test collection, uniform scoring procedures, and a forum for organizations interested in comparing their results. Since 2008, TRECVID has organized the copy detection task which initially used the CIVR 2007 Muscle benchmark. In 2010, only one sort of query is tested: audio + video. The video transformation includes:

T2: Picture in picture type 1
T3: Insertions of pattern
T4: Strong re-encoding
T5: Change of gamma
T6: Decrease in quality: a mixture of three transformations among blur, change of gamma, frame dropping, contrast, compression, ratio, white noise
T8: Post production: a mixture of three transformations among crop, shift, contrast, caption, flip, insertion of pattern, picture in picture type 2
T10: Combinations of transformations chosen from T2 to T8

In this paper we describe two systems that used solely visual data but different strategies for detecting copy segments given a query video and a reference video dataset. The first system—the NTNU CCD system — employed a global visual descriptor that were extracted from each sampled frame and a voting-based matching approach to identify the copy segments based on visual features. The second system—the Academia Sinica CCD system—applied a local-feature-based representation and a sliding-window matching scheme. We submitted three video based runs: one for the NoFA and two for the balanced profile. The evaluation shows some interesting results on the performance comparison of our different feature designs and matching schemes; nevertheless, more effort is required to further improve the effectiveness of our system by including audio features.

## 2. NTNU VIDEO COPY DETECTION

The overview of the NTNU video copy detection system is illustrated in Fig. 1. First, a video is partitioned into a sequence of frames. For simplicity, we sampled one frame per second in our system. Next, we applied a simple method based on edge information summarized in an averaged frame to detect letterbox and picture in picture (Section 2.1). Each frame is then summarized by a descriptor that calculates pair-wise content proximity over a pre-defined grid (Section 2.2). We used the $\chi^2$ statistics for comparing two frame descriptors. Finally, we applied a vote-based approach to determine the dissimilarity between each query video and the reference videos (Section 2.3) and retrieved the most similar video segment for each query video as our CCD results.

### 2.1 Letterbox and PinP Detection

Adding borders on video frames is one of the most common transformations made to a copy, as well as one of the transformations evaluated by TRECVID CCD task. We first remove the borders using a simple, heuristic method similar to that in [3].

We rely on the edge information and the temporal intensity variance of pixels to detect letterbox. The idea is that boarder areas usually have consistent pixel intensities in the frame sequence. For a video clip, we calculate the mean frame and apply the Sobel edge detector on the mean frame. We then project the edge pixels into the *x* and the *y* axis and determine the peak value respectively. The boundary between the frame content and the border is detected if the number of edge points on a particular line is more than a threshold and the variance of the pixel values of the border is small. This suggests a plain and narrow area exists. Furthermore, the aspect ratio of resulting frame and the percentage of the video pixels are verified. Figure 2 shows a few examples of the procedure, where red lines indicate the range of resulting frame. A similar procedure can be used to detect picture-in-picture type of frames with setting a different constraint on the locations of the boundary.

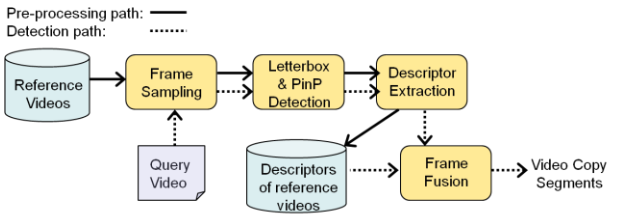

**Figure 1. Overview of the NTNU video copy detection system**

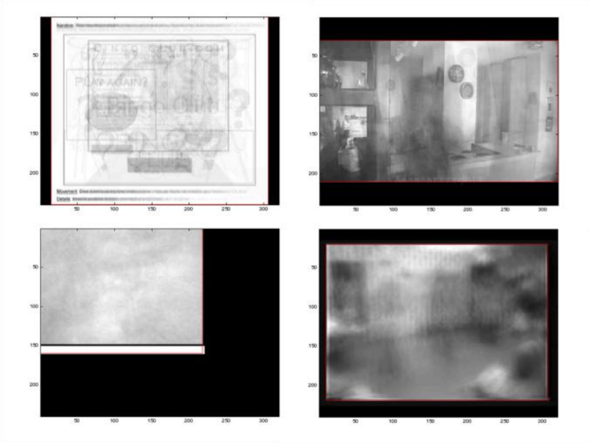

**Figure 2. Examples of letterbox detection.**

## 2.2 Frame Descriptor

A good frame descriptor should be distinctive for reliably distinguishing one frame of interest from others, compact, and robust with respect to transformations and random noises. In this subsection, we briefly describe our design of frame descriptor.

Our descriptor is constructed by firstly dividing each frame into a $K$ by $K$ equal-sized grid. We then compute the content proximity for each pair of blocks. Fig. 3 (a) shows an example using $K = 2$. Notice that the diagonal entries have a value of ones and the correlation matrix is symmetric. We extract the upper triangular part of the matrix and have our descriptor of $K^2$ $(K^2\text{-}1)$ / 2 dimensions.

For simplicity, we define the content proximity between block $X_i$ and block $X_j$ as:

$$w(i,j) = \exp(-\frac{1}{A} D(X_i, X_j)), \qquad (1)$$

where $D(X_i, X_j)$ is the sum of square differences (SSD) between block pixels[1] and $A$ is a scaling parameter. Fig. 3 (b) shows the proximity matrix of the left image. In our system, we divide a frame into 4x4 blocks, resulting in a 120-d descriptor.

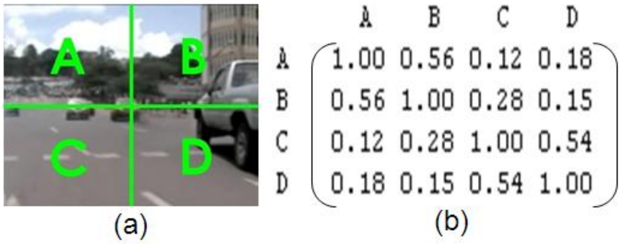

**Figure 3. The frame descriptor based on pair-wise correlations between predefined blocks.**

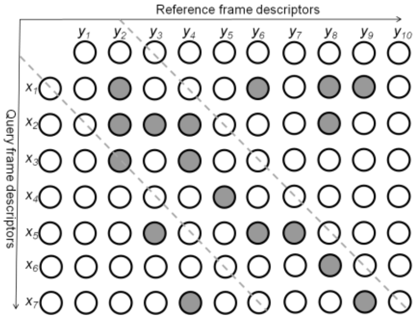

**Figure 4. The dot plot for frame fusion**

## 2.3 Frame Fusion

Given two frame descriptors, we use the $\chi^2$ statistics for comparing two frame descriptors. The$\chi^2$ statistics is defined as:

$$d_{\chi^2}(x, y) = \sum_i \frac{(x_i - y_i)^2}{x_i + y_i} \qquad (2)$$

For a query video, we calculated the pair-wise distances between every frame of a query video and those of the reference videos. Although a number of strategies can be applied to accelerate the matching process, we implemented a naïve approach which compares all descriptors in our database to each query frame descriptor. We set a threshold on the distance and identify those matching frame pairs if their distance is below the threshold. After we obtain the matching frame pairs between the query video and the reference videos, we can easily determine the candidate video copy segments as follows.

We first create a visual method called a *dot plot*. A dot plot puts a dot at $(i, j)$ in an $m$ by $n$ matrix if the descriptor $i$ and descriptor $j$ are matched. Figure 4 shows an example of the dot plot. The dot plot is extremely sparse even if two videos under comparison are partially related. Similar to the method in [1], the copy segments are then identified by computing the timestamp difference $\Delta = \mathrm{T}(i) - \mathrm{T}(j)$ of these dots. We find all matching pairs whose timestamp difference is in the range of $[\,\Delta - \delta, \Delta + \delta\,]$, where $\delta$ is set to 5 seconds in our system. Next, we identify the timestamp difference $\Delta$ with the maximal number of votes and retrieve all associated frame pairs. Among these frame pairs, we find the smallest and the largest timestamps for the query and the reference video streams respectively and obtain an approximate matching length $N$. The augmented similarity score for the segment is simply the sum of top $N$ scores of the matching pairs. To simplify the computation, we only retrieve the most relevant video segment for each query video based on the segment-level similarity scores.

[1] We computed the averaged value using the CIE L*a*b* space.

# 3. ACADEMIA SINICA VIDEO COPY DETECTION

For our second system, we apply different strategies for frame sampling, descriptor extraction and frame fusion. This subsection briefly discusses our method for these components.

## 3.1 Frame Sampling

Extracting features from all frames of the query and the reference videos would be too costly and inefficient due to the temporal redundancy between video frames. The extraction cost can be significantly reduced using frame sampling. Two popular frames sampling schemes are uniform sampling and key-frame sampling. We observe that key-frame sampling doesn't suit for video copy detection tasks since major key-frame detection scheme (i.e. shot change detection) may not be able to detect consistent frame pairs between the query and the reference videos. Furthermore, the number of key-frames is usually not sufficient to retrieve duplicate videos if the deformations applied on the query videos are strong. We therefore use uniform sampling to select a fix number of frames per second (5 frames per second in the system) for both the query and the reference videos.

## 3.2 Vide Representation

Unlike the NTNU copy detection system that uses a global feature for summarizing content in a frame, we apply an interest-point-based approach and describe a frame using a set of local features.

### *3.2.1 Key point extraction*

For each sampled frame, we select the local extrema in different Gaussian domains to be the key point candidates. First, each frame is transformed to a multi-scale space by convolving with Gaussian filters at different scales, and then each generated Gaussian-blurred image is subtracted with its neighbor image to generate the differences of Gaussian (DoG) images. Next, we identify key points as maxima/minima of the DoG that occur at multiple scales. In our implementation, we used the software of David Lowe with default parameters [5]. Finally, we select *N*+1 key points with the highest energy of local extrema on the DoG domain and discard the remaining key points. Each frame is, thus, represented by *N*+1 key points. We also sort the key points based on their energy and the sequence of key points will be used in the next step for identifying interest regions.

### *3.2.2 Interest region and orientation assignment*

Unlike most approaches that define so-called interest regions around each interest point, we use two neighboring key points in the sorted key point sequence to decide an interest region. Fig. 5 shows two interest regions marked with circles, where $P_i$ is the *i*-th key point in the sorted key point sequence (i.e. $P_i$ is the key point with the *i*-th largest energy on the DoG domain). For two key points $P_1(x_1,y_1)$ and $P_2(x_2,y_2)$ we find a circle with the center $(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$ and the radius $\frac{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}{2}$. The region within the circle between two key points is defined as an interest region. Since there are *N*+1 key points in each frame, we have in total *N* interest regions in each frame. This is the key step to achieve rotational invariance since the vector between $P_1$ and $P_2$ can be used for rotation correction. Next, the orientation [5] is assigned to the circle region as a feature.

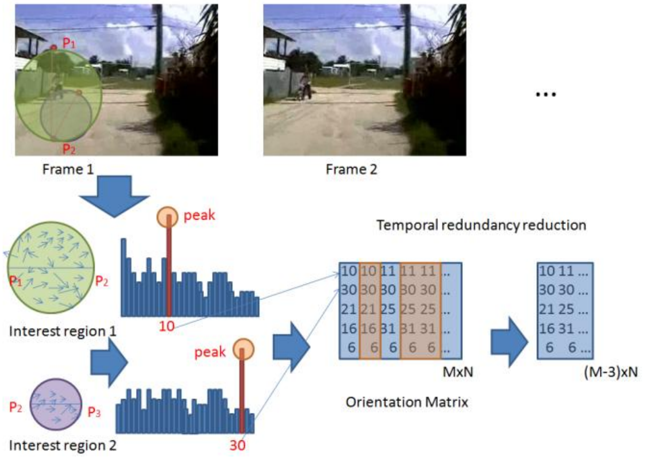

**Figure 5. The feature extraction process in the Academia Sinica copy detection system.**

In the orientation assignment step, each region is assigned only one orientation based on local image gradient directions in the circle. First, the Gaussian-smoothed image $L(x, y, \sigma)$at the scale $\sigma$ of the key point with the largest energy is taken so that all computations are performed in a scale-invariant manner. For an interest region $I(x, y)$ at scale $\sigma$, the gradient magnitude $a(x, y)$ and orientation $\theta(x, y)$ are computed as follows:

$$a(x, y) \\ = \sqrt{\left(I(x+1, y) - I(x-1, y)\right)^2 + \left(I(x, y+1) - I(x, y-1)\right)^2} \quad (3)$$

$$\theta(x, y) = \tan^{-1}\frac{I(x, y+1) - I(x, y-1)}{I(x+1, y) - I(x-1, y)}. \quad (4)$$

The magnitude and orientation calculations are done for every pixel in the interest region in the Gaussian-blurred image L. As shown is Fig. 5, an orientation histogram with 36 bins is formed, with each bin representing 10 degrees. Each pixel in the region is weighted by its gradient magnitude m(x,y). After the histogram is constructed, the orientation corresponding to the peak that is denoted with the red bars in Fig. 5 is assigned to the interest region. The orientations of the *m*-th interest region in *n*-th frame of a video are collected as a matrix to represent a video sequence for video matching.

### *3.2.3 Video presentation and temporal redundancy reduction*

After the orientation is assigned to each region, the video sequence can be represented as an *M* by *N* orientation matrix R:

$$R = [r_{m,n}]_{M\times N}, 1 \le m \le M, 1 \le n \le N, 1 \le r_{m,n} \le 36, (5)$$

where M represents the number of orientations in a frame, N is the number of frames in a video, and $r_{m,n}$ is the orientation of the *m*-th interest region in the *n*-th frame. Since the orientation can be considered the direction of the interest region, the difference between two orientations is derived as the angle between two unit vectors:

$$d\left(r_i, r_j\right) = \frac{\cos^{-1}(\sin(10*r_i)\sin(10*r_j)+\cos(10*r_i)\cos(10*r_j))}{180}. \quad (6)$$

The difference is used in our system for video matching which is described in next section.

Although the video frames are sampled as we have described in Section 3.1, the temporal redundancy still exists if the video is motionless. In order to reduce the temporal redundancy of video with few motions, we introduce a threshold $T$ to remove orientations of unnecessary frames. If the motion between two neighbor frames is small, the difference of two neighboring orientations in the same row is also small. Thus, we can summarize the difference of two neighboring frame with a weighting vector that gives higher weight to regions of high energy key points. The difference between two frames can be computed as:

$$D(r_n, r_{n-1}) = \sum_{m=1}^{M} w_m d\left(r_{m,n}, r_{m,n-1}\right). \qquad (7)$$

If the difference is smaller than the threshold (i.e. $D(r_n, r_{n-1}) < T$), the frame corresponding to $r_n$ (i.e. the *n*-th frame of video) is dropped and the orientations of whole column are removed from the matrix. The temporal redundancy reduction procedure is tested for all columns of each video's orientation matrix. Fig.5 shows an example, where the second, the fourth, and the fifth columns are removed since each is the identical to its left column. After the removal, the orientation matrix is simplified and more compact.

## 3.3 Video Matching

To match the query video and the reference videos, we use a matching scheme for two orientation matrices that can have different lengths.

Given two video sequences with the same number of frames, the difference between two videos can be computed as:

$$D\left(R^i, R^j\right) = \sum_{n=1}^{N} \sum_{m=1}^{M} w_m d\left(r^i_{m,n}, r^j_{m,n}\right). \qquad (8)$$

However, the lengths of the query and reference video are usually different. Thus, Eq. (8) cannot be directly used for evaluating the difference between two videos. To solve this problem, we adopt a sliding window strategy to divide the longer sequence into a number of subsequences for matching. As shown in Fig. 6, given a query video and a reference video with lengths $p$ and $q$, respectively, and suppose $p \le q$, the reference video sequence is first divided into $q$-$p$+1 overlapped sub-sequences with the length of $p$. The difference between query video and each sub-sequence can then be calculated using Eq. (8). The output of matching a query to the reference videos is a set of tuples (*ts*,*te*,*k*,*s*), where *ts* and *te* are the beginning and the ending timestamp of the matched subsequence of the reference video, *k* is the index of the reference video, and *s* is the score computed by (8) with the alignment suggested by *ts* and *te*. If the score between the query and the sub-sequence of the reference video *k* is minimum, the query is retrieved as a copy of the reference video *k* and the copy video segment is identified with timestamps *ts* and *te*.

# 4. EVALUATION

## 4.1 Dataset

TRECVID 2010 CBCD dataset contains 11256 queries and 11524 reference videos. For sampling a frame per second, we get more than 800,000 frames for the query video set and ~1,500,000 frames for the reference video set.

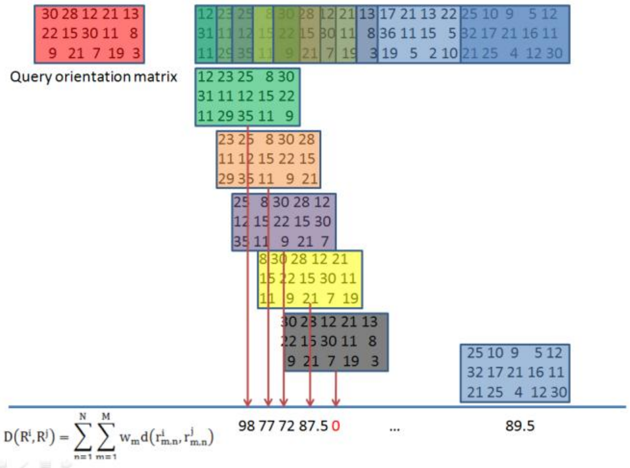

**Figure 6. An example of video matching in the Academia Sinica copy detection system.**

## 4.2 Evaluation Criteria

The normalized detection cost rate (NDCR) is used as the measure for the detection performance [8]. The detection cost rate (DCR) is defined as:

$$DCR = C_{Miss} \times P_{Miss} \times R_{Target} + C_{FA} \times R_{FA}, \qquad (9)$$

where $C_{Miss}$ and $C_{FA}$ are the costs of a miss and a false alarm, $R_{Target}$ is the priori target rate. The NDCR is defined as:

$$NDCR = \frac{DCR}{C_{Miss} \times R_{Target}} = P_{Miss} + \beta \times R_{FA}, \qquad (10)$$

where $\beta = C_{FA}/(C_{Miss} \times R_{Target})$ . Results of individual transformations within each run are evaluated separately. The minimal NDCR is found for each transformation and the actual NDCR is computed based on the threshold we reported.

To measure the location accuracy, the F1 measure based on the precision and the recall of the true video segments is used. Finally, the computational efficiency is measured by the mean time (in seconds) to process a query.

## 4.3 Results

We submitted in total 3 runs, labeled as NTNU-Academia-Sinica.m.balanced.1 (NTNU system for balanced profile), NTNU-Academia-Sinica.m.balanced.3 (Academia Sinica system for balanced profile), and NTNU-Academia-Sinica.m.nofa.2 (NTNU system for NoFa profile). For all runs we submit only the highest ranked matching video for each query. We set different threshold values that will be used to calculate actual NDCR for the profiles.

As we used solely the video content, it is not reasonable to compare our evaluation results with those who used both the video and the audio data. Evaluation results show that NTNU-Academia-Sinica.m.balanced.1 performs slightly better than NTNU-Academia-Sinica.m.balanced.3. In the rest of the section, we report the results of NTNU-Academia-Sinica.m.balanced.1 and NTNU-Academia-Sinica.m.nofa.2. We discuss the implementation flaws and provide possible improvements on NTNU-Academia-Sinica.m.balanced.3 at the end of the section.

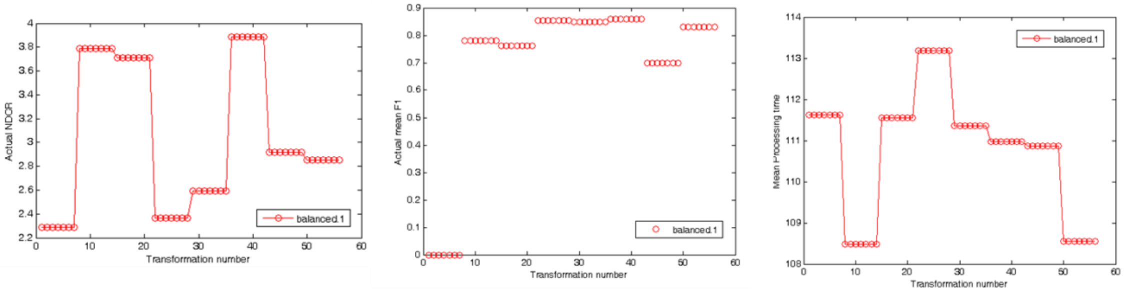

**Figure 7. Evaluation results for NTNU-Academia-Sinica.m.balance.1. Left: Actual NDCR. Lower values correspond to better performances. Midde: Actual mean F1. Right: Mean processing time for each query.**

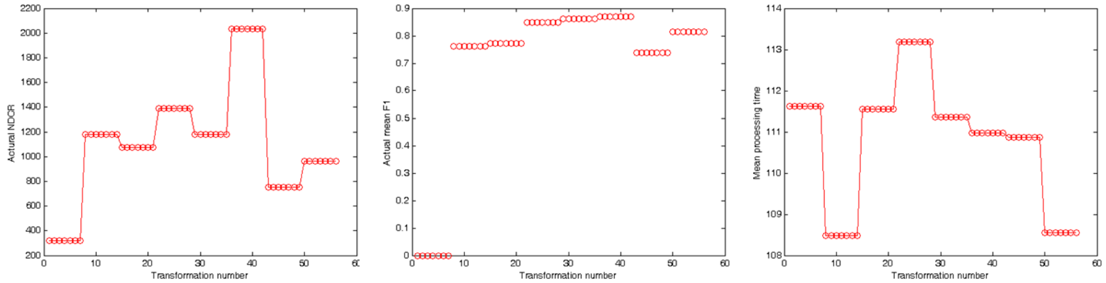

**Figure 8. Evaluation results for NTNU-Academia-Sinica.m.nofa.2. It differs from NTNU-Academia-Sinica.m.balance.1 in only the threshold values used to calculate actual NDCR.**

Since we used visual data alone, the accuracies of those combined audio+video transformations which have the same video transformation are identical. The actual NDCR values ranges from 2.29 to 3.89 (NTNU-Academia-Sinica.m.balanced.1) and from 322.21 to 2035 (NTNU-Academia-Sinica.m.nofa.2). Although we applied a global feature in these two runs, our system seems to be able to handle some of the transformations well without fine parameter tunings and modifications. The mean processing time for each query in these runs is around 110 seconds. The computational bottleneck is the frame-pair matching since we compared every sampled frame from the query video to all frames in the reference video set. An indexing structure for organizing the frame descriptors is needed to accelerate the matching process.

By observing the results of NTNU-Academia-Sinica.m.balanced.3, we find that the detection accuracy of the Academia Sinia System equally low for every TRECVID video transformations. In particular, noising and blurring degrade the detection of local features, i.e., SIFT, significantly, and, thus, affect the subsequent feature descriptor extraction and matching.

To solve the above problems as our future work, affine-SIFT [6] will be adopted to resist affine transforms. In addition, the blurred video can be viewed as a low resolution video. To match a low resolution video with a high resolution video, we will study the low-quality recognition problem [1][3] for designing elaborated features from low-resolution media data.

On the other hand, we also find that the threshold T, which is introduced to reduce the temporal redundancy in Sec. 3.2.3, is not properly set for generating NTNU-Academia-Sinica.m.balanced.3. Thus, the temporal redundancy was not efficiently reduced in NTNU-Academia-Sinica.m.balanced.3. Our system incorrectly matches two different videos when the motions of two videos are both small and when the orientations are close to each other. To correct the problem, the threshold needs to be carefully set.

## 5. CONCLUSIONS

We developed two systems for the TRECVID 2010 content-based copy detection task. We have not explored any audio features mainly due to time constraints but more efforts is definitely required to further improve the effectiveness of the system by including audio features. Besides, we realized that the SIFT-like local features need to be improved for overcoming the low-quality recognition problem. Finally, an indexing structure for organizing the extracted features in the reference video set that will be combined with our matching schemes is required to improve the scalability of our approach.

## 6. REFERENCES

[1] P. Hennings-Yeomans, S. Baker, and B.V.K. V. Kumar. Simultaneous super-resolution and feature extraction for recognition of low-resolution faces. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2008.

[2] A. Joly, O. Buisson, and C. Frelicot. Content-based copy retrieval using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, 9(2): 293-306, 2007.

[3] B. Li, H. Chang, S. Shan, and X. Chen. Low-resolution face recognition via coupled locality preserving mappings. *IEEE Signal Processing Letters*, vol. 17, No. 1, pp. 20-23, 2010.

[4] Z. Liu, T. Liu, D. Gibbon, B. Shahraray. Effective and scalable video copy detection. In *Proceedings of the ACM International Conference on Multimedia Information Retrieval (MIR'10)*, 2010.

[5] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60, 2 (2004), pp. 91-110.

[6] J. M. Morel and G. Yu. ASIFT: a new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, vol. 2, pp. 438-469, 2009.

[7] X. Wu, A. G. Hauptmann, and C. -W. Ngo. Practical elimination of near-duplicates from web video search. In *Proceedings of the ACM International Conference on Multimedia (MM'07)*, pages 218-227, 2007.

[8] http://www-nlpir.nist.gov/projects/tv2010/Evaluation-cbcd-v1.3.htm#eval