AT&T Research at TRECVID 2009 Content-based Copy Detection

Zhu Liu[1], Tao Liu[2], Behzad Shahraray[1]
[1]AT&T Labs – Research, Middletown, NJ 07748
[2]Polytechnic Institute of NYU, Brooklyn, NY 11201
[1]{zliu, behzad}@research.att.com, [2]tliu02@students.poly.edu

ABSTRACT

AT&T participated in one evaluation task at TRECVID 2009: the content-based copy detection (CBCD) task. In the proposed CBCD framework, we employed local visual features to match the video content, and adopted locality sensitive hashing techniques to maintain the scalability and the robustness of our approach. This year, we focused on the visual only CBCD task, and submitted three runs: one for NoFA (no false alarm) profile, and two for balanced profile. Evaluation results show that our CBCD system is highly effective and accurate.

1. INTRODUCTION

Video copy detection is essential for many applications, for example, discovering copyright infringement of multimedia content, monitoring commercial air time, querying video by example, etc. Generally there are two complimentary approaches for video copy detection: digital video watermarking and content-based copy detection (CBCD). The first approach refers to the process of embedding irreversible information in the original video stream, where the watermarking can be either visible or invisible. The second approach extracts content-based features directly from the video stream, and uses these features to determine whether one video is a copy of another. TRECVID copy detection task focuses on the second approach.

The goal of video copy detection is to locate segments within a query video that are copied or modified from an archive of reference videos. Usually the copied segments are subject to various transformations such as picture in picture, strong re-encoding, frame dropping, cropping, stretching, contrast changing, etc. Some of these transformations are intrinsic to the regular video creation process, e.g., block encoding artifacts, bit rate and resolution changes, etc. Others are introduced intentionally for various purposes. These include flipping, cropping, insertion of text and patterns (graphic overlays), etc. All these transformations make the detection task more challenging.

TRECVID 2009 CBCD considers the following 7 categories of visual transformation:

T2. Picture in Picture (PiP)
T3. Insertions of pattern
T4. Strong re-encoding
T5. Change of gamma
T6. Decrease in quality: a mixture of 3 transformations among blur, gamma, frame dropping, contrast, compression, ratio, white noise.
T8. Post production: a mixture of 3 transformations among crop, shift, contrast, text insertion, vertical mirroring, insertion of pattern, picture in picture.
T10. Combinations of 5 transformations chosen from T2 - T8.

In the proposed CBCD framework, we employed local visual features (Scale Invariant Feature Transform - SIFT) to match the content among query and reference videos. Locality Sensitive Hashing (LSH) and Random SAmple Consensus (RANSAC) techniques are exploited to maintain the scalability and increase the robustness of our approach. In total, we submitted three video based CBCD runs: one for NoFA and two for balanced profile. Evaluation results show that our system is effective and accurate. More effort is required to further improve the scalability of our approach.

2. OVERVIEW

Figure 1 illustrates the high level block diagram of our approach. The processing consists of two parts as indicated by different colors. The top portion illustrates the processing components for the query video, and the bottom portion shows the processing stages for the reference videos.

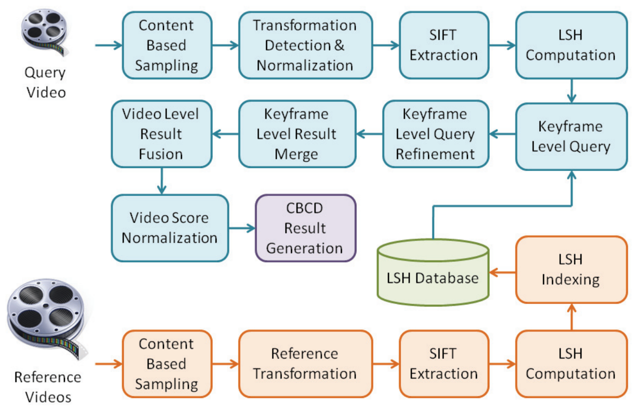


Figure 1. Overview of the proposed CBCD algorithm

The processing of reference videos is as follows. We first apply a content based sampling algorithm to segment the video into shots with homogeneous content and select one keyframe for each shot. The aim of content based sampling is data reduction, since indexing each reference frame is computationally intractable. To proactively cope with the transformations in the query video, we apply two transformations to each original reference keyframe: half resolution and strong re-encoding. These two additional versions of the reference keyframe are helpful to detect query keyframes with PiP and strong re-encoding transformations. All three sets of keyframes (one original and two transformed) are processed independently. Scale invariant feature transform (SIFT) features are extracted to represent the image content, and locality sensitive hashing (LSH) is adopted for efficient indexing and query. The LSH indexing results are saved in the LSH database.

For each query video, we first apply content based sampling to extract keyframes. We then detect a few pre-defined video transformations, including video stretch, PiP, scale, shift, etc. For each type of detected transforms, we normalize the original keyframe to reverse the transformation effect. For example, the stretched video is rescaled to the original resolution and the embedded picture-in-picture region is rescaled to half original resolution. Similar to the reference video processing, the original keyframe and the normalized keyframes go through the following processing independently, which includes SIFT feature extraction, LSH computation, keyframe level query based on LSH values, and keyframe level query refinement. The refinement module basically adjusts the relevance scores of all retrieved reference keyframes for each query keyframe by a more accurate but slower SIFT matching method based on RANdom Sample Consensus (RANSAC). Then for each query keyframe, the query results from different sets (the original and the transformed version) are merged based on their relevance scores. The keyframe level query results are then passed into the video level result fusion module, where the temporal relationship among the kefyrame query results as well as their relevance scores are considered. Finally, based on the different criteria of CBCD tasks, we normalize the decision scores of all detected reference videos, and then generate the final copy detection runs in TRECVID format.

3. VIDEO PROCESSING COMPONENTS

3.1 Content Based Sampling

Indexing every reference video frame and then searching all query frames is too computationally expensive. Content based sampling is an important step for data reduction. The goal is to segment the original video into shots, where each shot contains homogeneous content. The first frame within each shot is selected as a keyframe. All subsequent processing is applied only to the keyframe.

The content based sampling module is based on our shot boundary detection (SBD) algorithm developed for TRECVID 2007 evaluation task [3]. The existing SBD algorithm adopts a "divide and conquer" strategy. Six independent detectors, targeting the six most common types of shot boundaries, are designed. They are cut, fade in, fade out, fast dissolve (less than 5 frames), dissolve, and motion. Essentially, each detector is a finite state machine (FSM), which may have different numbers of states to detect the target transition pattern and locate the transition boundaries. Support vector machines (SVM) based transition verification method is applied in cut, fast dissolve, and dissolve detectors. Finally, the results of all detectors are fused together based on the priorities of these detectors. The FSMs of all detectors depend on two types of visual features: intra-frame and inter-frame features. The intra-frame features are extracted from a single, specific frame, and they include color histogram, edge, and related statistical features. The inter-frame features rely on the current frame and one previous frame, and they capture the motion compensated intensity matching errors and histogram changes.

By TRECVID SBD criteria, a shot may contain totally different scenes, as long as they belong to one camera shot, for example, a long panning shot. The existing SBD algorithm only reports one keyframe for such heterogeneous yet smooth shot. To make the content based sampling more effective, we expanded the existing SBD algorithm with an additional detector: sub shot detector. This detector will measure the color histogram dissimilarity between current frame and the most recently detected keyframe. If their difference is large enough, a subshot boundary is declared and a new keyframe is extracted.

Another modification of the existing content based sampling module is to deal with frame dropping. The dropped frames in the query videos provided by NIST are mostly replaced by blank frames. These blank frames created a large number of false cut boundaries. We updated the finite state machine model of the cut detector to tolerate up to 4 adjacent blank frames.

3.2 Transformation Detection and Normalization For Query Keyframe

While generating the video queries, TRECVID CBCD evaluation task introduced the following transformations: Cam cording, picture in picture, insertions of pattern, strong re-encoding, change of gamma, decrease in quality (e.g., blur, frame dropping, contrast change, ratio, noise, crop, shifting, flip, etc.), and combinations of individual transforms. It is not realistic to detect all kinds of transform, and recover the effect, since some of the transformation is irreversible. In this work, we mainly focus on letterbox detection and picture in picture detection. The letterbox may be introduced by shifting, cropping, or stretching. We do not differentiate them, but simply remove the letterbox by rescaling the embedded video into the original resolution.

3.2.1 Letterbox detection

We rely on both edge information and the temporal intensity variance of each pixel to detect letterbox. For each frame, the Canny edge detection algorithm is applied to locate all edge pixels. Then based on the edge direction, a horizontal edge image and a vertical edge image are extracted. To cope with the noise in the original video, the edge images have to be smoothed. The smoothing process includes removing the short edge segments, and merging adjacent edge segments that are less than 5 pixels away. After all frames of the query video are processed, the mean horizontal (and vertical) edge image is computed for the entire video and it is projected horizontally (and vertically) for computing the horizontal (and vertical) edge profile. For stretched, cropped, or shifted videos, there are significant peaks in the horizontal and/or vertical edge profiles. We search the maximum horizontal (and vertical) peak near the boundary, such that it is within one eighth of the height (and the width). Usually, these detected profile peaks indicate the letterbox boundaries.

When the video is too noisy, profile peaks may not be detected reliably. In this case, we rely on the intensity variation of each pixel across the entire video. For the embedded video region, the intensity variance of each video is relatively large due to the dynamic video content. For the letterbox region, the variance is much smaller. Comparing the pixel intensity variance value to a preset threshold, we classify each pixel into either the video pixel or the non-video pixel. We then search the letterbox boundaries such that the percentage of video pixels in the letterbox region is less than 1%.

3.2.2 Picture-in-picture detection

TRECVID specifies that picture-in-picture may appear at five locations: the center and the four corners. The size of the picture-in-picture region is less than half of the original resolution, and bigger than 0.3 of the original size. These constraints are very useful in reducing the searching complexity in picture-in-picture detection. Figure 2 (a) shows an example of reference video embedded in PiP, and Figure 2 (b) shows the enlarged version of the PiP region. Orhan et al. described a PiP extraction algorithm in [4], where persistent strong horizontal lines in image derivatives in x-axis are found using Hough lines, and the PiP region is extracted by connecting horizontal parallel lines. The algorithm achieves an accuracy of 74%. However, it still misses many positive PiPs. In this work, we devise a more effective approach.

As in the letterbox detection algorithm, the devised method is based on both edge information and pixel intensity variance. Our method contains three basic steps. First, all peaks of the horizontal (and vertical) edge profile are located. Based on these peak locations, we find the original horizontal edge segments (and the vertical edge segments). These edge segments are the candidate PiP region boundaries. Due to the size of PiP region, we remove those edge segments that are either too short or too long. Figure 2 (c) – (f) illustrates the horizontal (and vertical) edge images and their profiles. Second, we determine the PiP region candidates by grouping a pair of horizontal edge segments and a pair of vertical edge segments together. The pair of the horizontal edge segments should be not too close or too far away from each other vertically, and should overlap in terms of their horizontal positions. The vertical edge segments need to follow a similar constraint. The sample shown in Figure 2 is simple; it only contains one pair of horizontal edge and vertical edge segments. Note that the broken horizontal edge segments in Figure 2 (c) are merged into one segment during the smoothing process. Third, each candidate PiP region need to be verified, and a likelihood score is assigned to it. The verification criteria include the aspect ratio of the region, the percentage of the video pixels. The summation of the boundary edge intensity is used as the likelihood value. Finally the region with the maximum likelihood is picked as the PiP region.

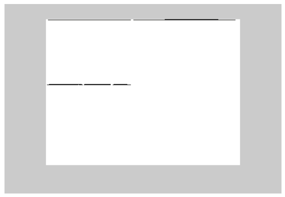
Given that the PiP region is smaller than half of the original resolution, we rescale the detected PiP region to be half size as a normalization process. Generally speaking, SIFT features are robust to the scale change, but we did find that the transformation detection helps to further improve the robustness of SIFT feature extraction. Transformation detection also benefits copy detection methods based on global visual features, including Gabor texture, edge direction histogram, and grid color moments, etc. Although these features can be easily computed, they are very sensitive to geometric changes. Transformation detection helps to improve the robustness of these approaches.

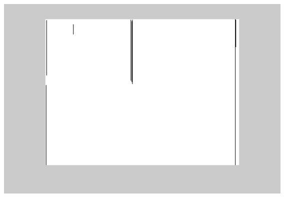
3.2.3 Query Keyframe Normalization

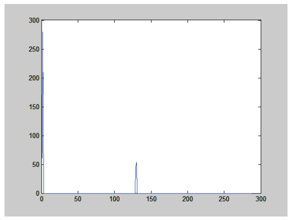
In addition to the letterbox removal and PiP rescaling, we equalize and blur the query keyframe to overcome the effect of change of Gamma and white noise transformations. Due to the fact that the SIFT features are not invariant to mirror transformations, we have to create a flipped version for each of these normalized query keyframes. In summary, we have 10 types of query keyframe: original, letterbox removed, PiP scaled, Equalized, blurred, and flipped versions of these five types.

(a) A Query keyframe

(b) PiP region


(c) Horizontal edge Image


(d) Vertical edge image


(e) Horizontal edge profile

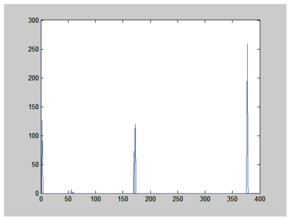

(f) Vertical edge profile

Figure 2. An example for PiP Detection

3.3 Reference Keyframe Transformation

Complementary to normalizing the query keyframes, the reference keyframes can be pre-processed to eliminate the transformation effect as much as possible. Because of the large volume of the reference video set, we cannot afford to have too many variations for all reference keyframes. In our work, we simply apply two transformations on reference keyframes. They are half resolution rescaling and strong re-encoding. The first transformation is for comparing with the detected PiP region in the query keyframes, and the second transformation is useful in dealing with the strong re-encoded query keyframes. So in total, we have 3 different types of reference keyframes.

3.4 SIFT Extraction

The scale-invariant feature transform (SIFT) features [5] have been proven effective in various video searching and near duplicate image detection tasks. In this work, we adopted SIFT as the main feature for locating the video copies. SIFT extraction is composed of two steps: 1) Locating the keypoints that have local maximum Difference of Gaussian values both in scale and in space. These keypoints are specified by location, scale, and orientation. 2) Computing a descriptor for each keypoint. The descriptor is the gradient orientation histogram, which is a 128 dimension feature vector.

We rely on Vedaldi's implementation [6] of SIFT because of its ease of use and the accompanying Matlab code for visualization and debugging purpose. More efficient versions of SIFT, including the SURF (Speeded Up Robust Features) features [7], are also considered in this work. But at this point, we have not evaluated their effectiveness.

The default parameters for the SIFT feature computation program can generate a few thousands of SIFT features for a single image. This brings more computational complexity in the SIFT matching step. For video copy detection, we need a much smaller number of SIFT features to verify that a frame is from the reference video. To reduce the number of SIFT features, we set the edge threshold to be 5 and the peak threshold to be 7. With this set of parameters, the number of SIFT features is reduced to about two hundred for each frame, which is roughly ten times less than the default setting.

3.5 Locality Sensitive Hashing

Directly comparing the Euclidian distance between two SIFT features in the 128 feature dimension is not scalable at all. In this work, we utilize the locality sensitive hashing (LSH) [8] for efficient SIFT feature matching.

The idea of the Locality sensitive hashing is to approximate the nearest neighbor search in high dimensional space. Following is a brief introduction on LSH. Each hash function $h_{\mathbf{a},b}(\mathbf{v})$ maps a 128 dimensional vector $\mathbf{v}$ onto the set of integers (bins),

$$h_{\mathbf{a},b}(\mathbf{v}) = \left\lfloor \frac{\mathbf{a}\cdot\mathbf{v}+b}{w} \right\rfloor,$$

where w is a preset bucket size, b is chosen uniformly in the range of $[0, w]$, and $\mathbf{a}$ is a random vector following a Gaussian distribution with zero mean and unit variance. This hash function possesses the desirable property that when $\mathbf{v}_1$ and $\mathbf{v}_2$ are closer in the original vector space (e.g., $\|\mathbf{v}_1 - \mathbf{v}_2\|$ is smaller), their hash values $h_{\mathbf{a},b}(\mathbf{v}_1)$ and $h_{\mathbf{a},b}(\mathbf{v}_2)$ are more likely to be the same, and when $\mathbf{v}_1$ and $\mathbf{v}_2$ are farer away in the original vector space (e.g., $\|\mathbf{v}_1 - \mathbf{v}_2\|$ is bigger), their hash values $h_{\mathbf{a},b}(\mathbf{v}_1)$ and $h_{\mathbf{a},b}(\mathbf{v}_2)$ are less likely to be the same. Using a Hash function, the complex high dimension vector distance comparison is converted into one integer comparison, which is extremely efficient.

Two additional parameters to tune the hashing performance are: 1) Combine k parallel hashing values to increase the probabilities that vectors far away fall into the same bin; 2) Form L independent k-wise Hash values to reduce the probability that similar vectors are "unluckily" projected into different bins. For the SIFT features we computed, we found that the following parameters give satisfactory results: $w = 700$, $k = 24$, and $L = 32$. This basically creates 32 independent Hash values for each of the SIFT features. For two SIFT features, if one or more of their L Hash values are the same, the two SIFT features are reported as matching.

3.6 Indexing and Search by LSH

Indexing the L LSH values for the reference videos is straight forward. We simply sort them independently and save the sorted hash values with their SIFT identifications (a string that is composed of the reference video ID, the keyframe ID, and the SIFT ID) in a separate index file.

Querying the index file for an LSH value by binary search is very efficient, since the complexity is in the order of $\log(N)$, where N is the cardinality of the index file. For a query keyframe, the task is to find all reference keyframes with matching SIFT features and using the number of matching SIFT pairs as the relevance score for ranking purpose. To reduce the computational complexity for the following process, we only keep the top 256 reference keyframes on the matched list.

3.7 Keyframe Level Query Refinement

It is obvious that keyframe matching purely based on SIFT and LSH is not reliable enough. There are two issues: 1) the original SIFT matching by Euclidian distance is not reliable, especially when the number of SIFT features is large and various transformations may introduce noise in SIFT extraction, 2) it is possible that two SIFT features that are far away mapped to the same LSH value. We need to rely on an additional mechanism to validate the list of retrieved reference keyframes produced in the last section. In this work, RANdom Sample Consensus (RANSAC) [1] [2] is utilized for this purpose.

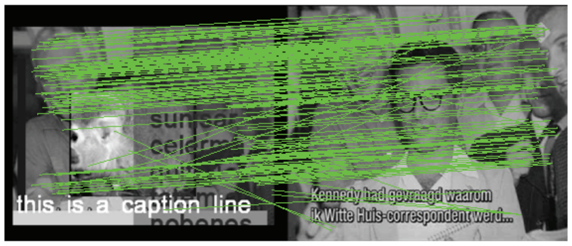
RANSAC is an iterative method for estimating model parameters from observed data with outliers. Here, RANSAC is used for estimating the affine transform that maps the keypoints in the original reference keyframe to those in the query keyframe. The affine transform is able to model the geometric changes introduced by following transforms: PiP, shift, ratio, etc. Specifically, the keypoint at pixel (x, y) in the reference keyframe is mapped to pixel (x', y') in the query keyframe by the following formula,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

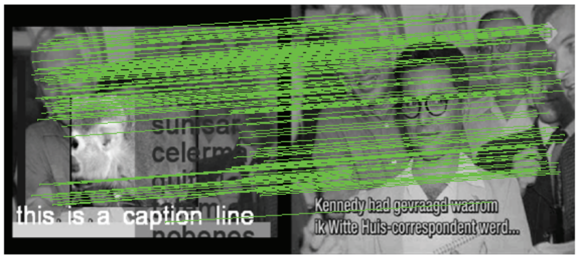
To determine the affine model parameters (a, b, c, d, e, f), three pairs of keypoints, where the keypoints in the reference keyframe are not on a line, are required. For a pair of keyframes, the detailed RANSAC procedure is as follows,

1. Randomly select 3 pairs of matching keypoints (having the same LSH values).
2. Determine the affine model.
3. Transform all keypoints in the reference keyframe into the query keyframe.
4. Count the number of keypoints in the reference keyframe whose transformed coordinates are close to the coordinates of their matching keypoints in the query keyframe. These keypoints are called inliers.
5. Repeat steps 1 to 4 for a certain number of times, and output the maximum number of inliers.

Figure 3 presents an example of RANSAC verification. Figure 3 (a) displays a query keyframe (on the left side) and a reference keyframe side by side, and links all original matching keypoints by lines. Figure 3 (b) shows the matching keypoints after RANSAC verification. All of the wrong matching keypoints are removed.



(a) Original matching keypoints



(b) Matching keypoints after RANSAC verification

Figure 3. SIFT verification by RANSAC

After RANSAC verification, we use the maximum number of inliers as the new relevance score between a query keyframe and a reference keyframe.

3.8 Keyframe Level Result Merge

As mentioned in Section 2, to cope with the visual effects introduced by various transformations, we transform the reference keyframes to mimic the query transformations, as well as normalize the query keyframes to recover the original content. Table 1 shows all the combinations of transformed reference keyframe and normalized query keyframe that we considered. The choice of combinations in this work is is driven by the requirements of the TRECVID CBCD evaluation task. For other applications, a reduced or expanded list may be more feasible. For example, in the application of commercial detection, the pair of original query keyframe vs. original reference keyframe alone is sufficient. Considering more pairs introduces more computational complexity. So it is a tradeoff in the real system to find the optimized pair list in terms of detection accuracy and speed.

As shown in Table 1, in total we considered 12 combinations in this work. Hence for each query keyframe, there are 12 lists of ranked reference keyframes. The merging process is to simply combine these 12 lists into 1 list. If one reference keyframe appears more than once in the 12 lists, its new relevance score is set to be the maximum of its original relevance scores, otherwise, its relevance score is the same as the original one. The new list is ranked based on the new scores, and then only the top 256 reference keyframes are kept for further processing.

3.9 Video Level Result Fusion

Based on the query results of all keyframes of a query video, we can easily determine the list of best matching videos for it. Let us consider one query video and one reference video, where at least one keyframe of the reference video is on the matched list of at least one keyframe of the query video. The query video has Q keyframes, and the reference video has R keyframes, the relevance scores of between query keyframe i and reference keyframe j is denoted by S(i, j). The timestamp of query keyframe i is QT(i), and that of reference keyframe j is RT(j). Figure 4 plots the relevance score matrix, where all non-zero entries are marked by dark squares. For each pair of matched keyframes, we compute an extended relevance score for it, and the relevance score between these two videos is the maximum of all extended relevance scores. In the following, we describe the details on how to compute the extended relevance score.

Table 1. Normalized query keyframes vs. transformed reference keyframes

Pair	Query keyframes	Reference keyframes
1	Original	Original
2	Flipped	
3	Letterbox removed	
4	Letterbox removed & flipped	
5	Equalized	
6	Equalized & flipped	
7	Blurred	
8	Blurred & flipped	
9	Original	Encoded
10	Flipped	
11	Picture in Picture (PiP)	Half
12	PiP & flipped	

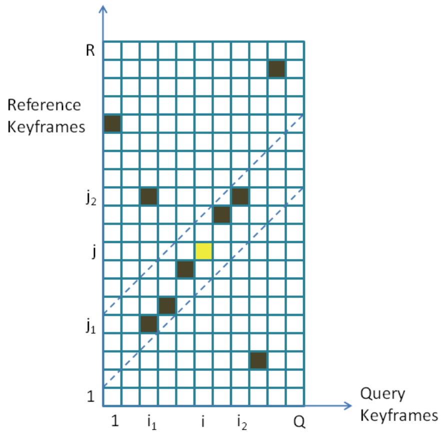


Figure 4. Video Level Result Fusion

For a matching keyframe pair (i, j), which is highlighted in Figure 4, we first compute the timestamp difference: $\Delta = RT(j) - QT(i)$. Then we find all matching keyframes whose timestamp difference is in the range of $[\Delta - \delta, \Delta + \delta]$, where δ is set to 5 seconds in this work. The extended relevance score for pair (i, j) is simply the sum of the relevance scores of these filtered pairs. To reduce the impact of the number of query keyframes, N, we normalize the extended relevance score by 1/log(N). In Figure 4, the time difference range is marked by two dashed lines. The video level matching determined by pair (i, j) is query keyframes $[i_1, i_2]$ => reference keyframes $[j_1, j_2]$, and $N = i_2 - i_1 + 1$. Once we have the extended relevance scores for all pairs, the one with maximum score is picked as the video level matching between corresponding reference video and query video.

Based on the video level relevance scores, we sort all matching videos for a query video. To simplify the computation, we only keep the top 8 matching videos for each query video.

3.10 Video Relevance Score Normalization

TRECVID CBCD evaluation is conducted for copy detection results of all query videos together, but not on each query video individually. Different decision thresholds are used to truncate the submitted results to measure the probability of miss and the false alarm rate. Therefore, it is important to normalize the video relevance scores across the entire collection of query videos to boost the system performance.

First, we normalize the relevance scores into the range of [0, 1] by the following sigmoid function,

$$y = \frac{2}{1+\exp(-x/50)} - 1,$$

where x is the original relevance score, and y is the normalized one.

Then, for each query video, we normalize the score of the best matching reference video. Let's assume that there are N matching reference videos for a query video, and their relevance scores are $\{x_i\}$, i = 1,…, N, satisfying $x_i > 0$ and $x_i >= x_j$, if i < j. x_1 is normalized as $x_1 = x_1*(x_1/x_2)$. The motivation for this normalization is that if the best matching video has a much higher relevance score than the second best matching video, it is more likely that the best matching video is correct. Reference videos whose relevance scores are less than half of the best matching one are removed, since they are likely to be wrong.

Finally, the scores are normalized across all query videos. As we mentioned in the last section, each query video may have up to 8 matching reference videos. The idea is to map the relevance scores of all best matching reference videos of all query videos into the range of [7, 8] by a sigmoid function, and then map the relevance scores of the second best matching reference videos of all query videos into the range of [6, 7], and so on and so forth. In such a way, the higher ranked matching videos have higher priority to be included in the final detection results.

3.11 CBCD Result Generation

TRECVID requires that the copy detection results for all query videos of one system be compiled in one run file. Within the run file, we need to provide a list of result items, where each item specifies the query video ID, the found reference video ID, the boundary information of the copied reference video segment, the starting frame of the copied segment in the query video, and a decision score – the higher the number, the stronger the evidence. In addition to the copy detection results, participants also need to provide additional information, including the processing time for each query video, the operating system, the amount of memory, etc. The timing information is used to measure the efficiency of the submitted approach.

4. CBCD Evaluation Results

TRECVID 2009 CBCD dataset contains 1407 short query videos, and 838 reference videos (from the Netherlands Institute for Sound and Vision) and 208 non-reference videos (from the British Broadcasting Corporation). In total, we extracted ~268 thousand keyframes and ~57 million SIFT features for the entire reference video set, and ~18 thousand keyframes and ~2.6 million SIFT features for the query video set.

4.1 CBCD Evaluation Criteria

There are three performance measures specified for CBCD task [9]. They are minimal normalized detection cost rate (min NDCR), copy location accuracy, and the processing time.

The detection cost rate (DCR) is defined as,

$$DCR = C_{Miss} \cdot P_{Miss} \cdot R_{target} + C_{FA} \cdot R_{FA},$$

where C_{Miss} and C_{FA} are the costs of a miss and a false alarm, R_{target} is the priori target rate. In order to compare the detection cost rate values across a range of parameter values, DCR is normalized as follows,

$$NDCR = \frac{DCR}{(C_{Miss} \cdot R_{target})} = P_{Miss} + \beta \cdot R_{FA}, where$$

$$\beta = C_{FA} / (C_{Miss} \cdot R_{target})$$

Results of individual transformations within each run are evaluated separately. Different decision thresholds θ are applied to generate a list of pairs of increasing P_{Miss} and decreasing R_{FA}. The minimal NDCR is found for each transformation. The actual NDCR is computed based on the optimal threshold that we reported. TRECVID 2009 CBCD task has two profiles: NoFA (no false alarm) and Balanced. The parameters used for NoFA profile are

$$R_{target} = 0.5/hr, C_{Miss} = 1, C_{FA} = 1000,$$

and the parameters used for Balanced profile are

$$R_{target} = 0.5/hr, C_{Miss} = 1, C_{FA} = 1.$$

Copy location accuracy is defined to assess the accuracy of finding the exact extent of the copy in the reference video. This is only measured for the correctly detected copies. Mean F1 (harmonic mean) score based on the precision and recall of the detected copy location relative to the true video segment is adopted.

Copy detection processing time is the mean time to process a query. It includes all processing from reading in the query video to the output of results.

4.2 CBCD Evaluation Results

In total, we submitted 3 runs, labeled as ATTLabs.v.nofa.1 (for NoFA profile), ATTLabs.v.balanced.2 (for balanced profile), and ATTLabs.v.balanced.3 (for balanced profile). The only difference of them is the way how we generate the final CBCD run files. For ATTLabs.v.nofa.1, we want to reduce the false alarm rate as much as possible. In this run, for each query video, we only include the highest ranked matching video that also satisfies the following two criteria: 1) the relevance score is higher than a threshold 0.1, 2) the relevance score is higher than 1.5 times of the second highest score. For ATTLabs.v.balanced.2, we only include the highest ranked matching videos for each query video as long as its relevance score is higher than 0.1. For ATTLabs.v.balanced3, we further relax the criteria. For each query video, we include up to 12 matching videos as long as their relevance scores are higher than 0.1 and they are higher than one third of the top relevance score.

Evaluation results show that run ATTLabs.v.balanced.2 performs slightly better than run ATTLabs.v.balanced.3. In the rest of this section, we mainly discuss the evaluation results of two runs: ATTLabs.v.nofa.1 and ATTLabs.v.balanced.2.

There are 24 runs submitted in the NoFA profile from all 20 participants. Figure 5 and Figure 6 show the performance of run ATTLabs.v.nofa.1; one is the actual NDCR and the other is the minimum NDCR. Each figure contains 3 plots. From the top to the bottom, they are the NDCR, F1 measure, and the processing time. Each transformation category is measured separately. The performances of the best runs are marked with squares, and the median performances are marked in dashed line. The results of our system are marked with dots. From these figures, it is obvious that the difficulty of copy detection for different transformations varies. Overall, our system achieves excellent NDCR performance: top in four transformation categories (T4, T5, T8, and T10) for the actual NDCR, and top for all transform categories for the minimum NDCR. The detection accuracy (F1) of the system is reasonable good too, well above the median run for all cases.

There are 29 runs submitted in the balanced profile. Figure 7 and Figure 8 show the performance of run ATTLabs.v.balanced.2. Again, our system performs well in the balanced profile: top in T4 for the actual NDCR, and top for five transformation categories (T2, T4, T6 T8, and T10) for the minimum NDCR. Table 2 lists the detailed NDCR performance of these two runs.

The good performance of our system is mainly due to the following two factors: 1) SIFT and RANSAC provides reliable content matching mechanism; 2) The combination of transformation detection for query video and transformation for reference video improves the chance of detecting the original videos.

The processing time of our system is roughly the same for all different transformations. That is anticipated since all query videos go through the same processing steps. The mean processing times of our system are slightly higher than those of the median runs. This shows that there is space to improve the efficiency of our system. A few thoughts on further reducing the computational complexity are 1) decreasing the number of SIFT features and/or the number of LSH hashing sets; 2) improving the implementation, for example, adopting a more efficient way to read in the large index file (e.g., larger than 2GB) and parallelizing the query processing; 3) investigating more on the pairs of the transformed reference keyframes and the normalized query keyframes (see Table 1).

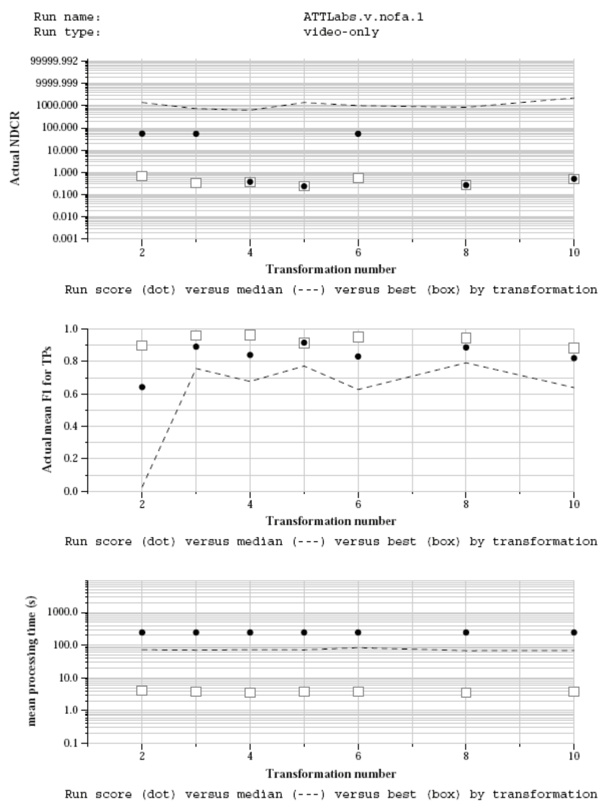


Figure 5. Actual NDCR and F1 for ATTLabs.NoFA.1

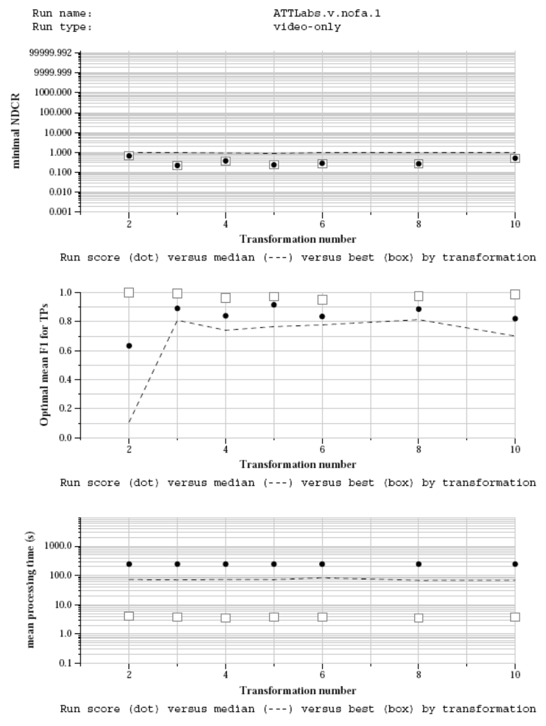


Figure 6. Optimal NDCR and F1 for ATTLabs.NoFA.1

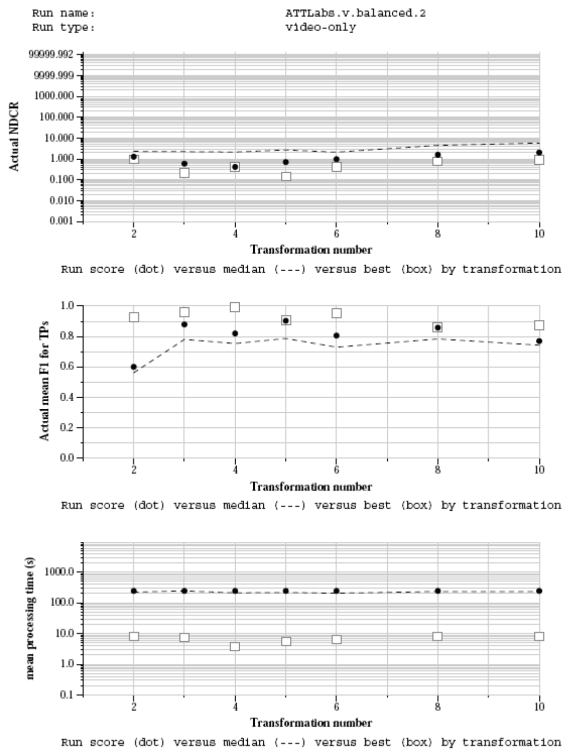


Figure 7. Actual NDCR and F1 for ATTLabs.Balanced.2

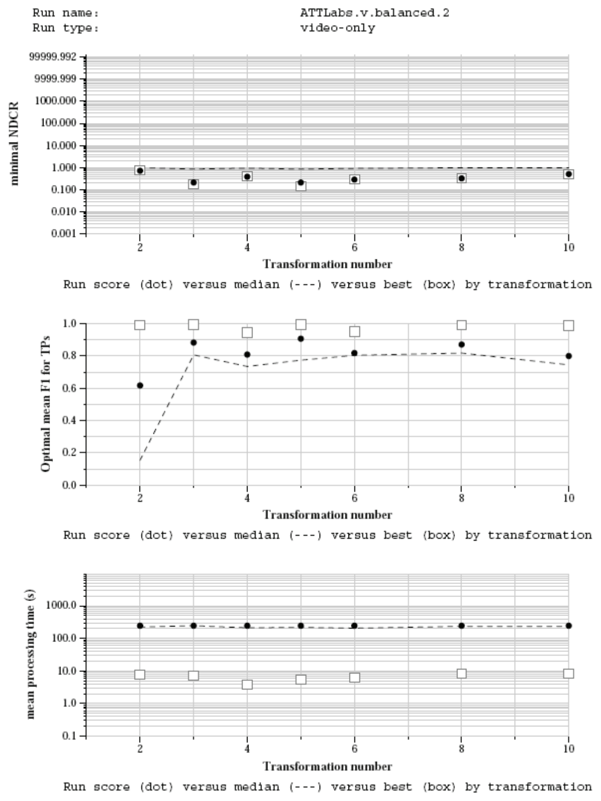


Figure 8. Optimal NDCR and F1 for ATTLabs.Balanced.2

Table 2. NDCR for ATTLabs.NoFA.1 & ATTLabs.Balanced.2

Transformations	ATTLabs.NoFA.1		ATTLabs.Balanced.2	
	Actual	Minimum	Actual	Minimum
T2	55.4	0.672	1.283	0.732
T3	55.0	0.224	0.59	0.214
T4	0.381	0.381	0.413	0.391
T5	0.239	0.239	0.7	0.214
T6	55.1	0.284	0.974	0.291
T8	0.269	0.269	1.585	0.329
T10	0.515	0.515	2.045	0.52

5. Conclusions

In this paper, we presented a content based copy detection system. The system is based on SIFT features and relies on LSH for scalability. RANSAC is used for reliable SIFT matching. The evaluation results show that our system achieves good performance for both detection cost rate and detection accuracy. More effort is needed to further improve the scalability of our approach.

6. REFERENCES

[1] Martin A. Fischler and Robert C. Bolles (June 1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. Of the ACM 24: 381–395.

[2] W. Zhang and J. Kosecka, "Generalized RANSAC Framework for Relaxed Correspondence Problems," 3DPVT 2006, Chapel Hill, NC.

[3] Z. Liu, D. Gibbon, E. Zavesky, B. Shahraray, P. Haffner, "A Fast, Comprehensive Shot Boundary Determination System," ICME 2007, Beijing, China, July 2-5, 2007.

[4] O. Orhan, et al, "University of Central Florida at TRECVID 2008 Content Based Copy Detection and Surveillance Event Detection," TRECVID Workshop, Nov. 17-18, 2008, Gaithersburg, MD.

[5] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," IJCV, 2 (60), pp. 91-110, 2004.

[6] A. Vedaldi and B. Fulkerson, "VLFeat: An Open and Portable Library of Computer Vision Algorithms," 2008, http://www.vlfeat.org/.

[7] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, 2008.

[8] A. Andoni and P. Indyk, "Near-Optimal Hashing Algorithms for Near Neighbor Problem in High Dimension," Communications of the ACM, Vol. 51, No. 1, 2008.

[9] W. Kraaij, G. Awad, P. Over, "TRECVID 2008 Content-based Copy Detection Task Overview," TRECVID Workshop, Nov. 17-18, 2008, Gaithersburg, MD.