National Institute of Informatics, Japan at TRECVID 2008

Duy-Dinh Le ^{*1}, Xiaomeng Wu ^{*1}, Sheetal Neeraj Rajgure ^{**2}, Jan van Gemert ^{***3}, Shin'ichi Satoh ^{*1}

* *National Institute of Informatics*
2-1-2 Hitotsubashi, Chiyoda-ku, Japan 101-8430
^{1} ledduy, wxmeng, satoh@nii.ac.jp

** *New Jersey Institute of Technology*
University Heights, Newark, New Jersey 07102, USA
^{2} sheetal@nii.ac.jp,

*** *University of Amsterdam,*
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
^{3} jvgemert@uva.nl,

***Abstract*—This paper reports our experiments for TRECVID 2008 tasks: high level feature extraction, search and content-based copy detection. For the high level feature extraction task, we use the baseline features such as color moments, edge orientation histogram and local binary patterns with SVM classifiers and nearest neighbor classifiers. For the search task, we use different approaches including search by the baseline features and search by concept suggestion. And for the video copy detection task, we study two approaches that are based on the pattern of motions in feature point trajectories and matches of all frame pairs using normalized cross correlation. Our approaches can be considered as one of the baseline approaches for evaluation of these tasks.**

I. High Level Feature Extraction

A. Method Overview

In our framework, features are extracted from the input keyframe images representing for shots. We extracted five keyframes per shot that are spaced out equally within the provided shot boundary. In the training stage, we use these features to learn SVM classifiers and nearest neighbor classifiers. These classifiers are then used to compute the raw output scores for the test image in the testing stage. These output scores can be further fused by taking the average for computing the final output score. In order to return K shots most relevant for one concept query that then are evaluated and compared in TRECVID benchmark, all normalized final output scores of shots are sorted in descending order and top K shots are returned. In the case of a shot consisting of several sub-shots, only the maximum score among subshots' scores is used for that shot.

As for feature extraction, we used three types of features color moments, edge direction histogram and local binary patterns. These features are extracted from a 5x5 grid of the input image, normalized to zero mean and unit standard deviation and then stored for training and testing. Specifically, the normalized vector $x^{norm} = (x_1^{norm}, x_2^{norm}, ..., x_N^{norm})$ of an input raw vector $x^{raw} = (x_1^{raw}, x_2^{raw}, ..., x_N^{raw})$ is defined as follows:

$$x_i^{norm} = \frac{(x_i^{raw} - \mu)}{\sigma}$$

where x_i^{norm} and x_i^{raw} is the i-th element of the feature vectors x^{norm} and x^{raw} respectively, N is the number of dimensions. μ is the mean $\mu_i = \frac{1}{N}\sum_{i=1}^{N} x_i^{raw}$ and σ is the standard deviation

$$\sigma_i = \sqrt{(\frac{1}{N}\sum_{i=1}^{N} x_i^{raw} - \mu_i)^2}$$

We used the annotation data of TRECVID 2005 and TRECVID 2008 [6] to form the training sets. We divided the original training set into 4 subsets. We form each subset by randomly selecting a maximum of 10,000 samples from the original positive and negative sets. By this way, we can handle the problem of imbalanced training sets (99% is negative). Furthermore, we did not merge the training sets of TRECVID 2005 and TRECVID 2008. Instead, we used them separately. Therefore, we have 8 sub-training sets, 4 subsets for each.

As for SVM classifiers, LibSVM [1] is used to train SVM classifiers with RBF kernel. The optimal (C, g) parameters are found by conducting a grid search with 5-fold cross validation on a subset of 1,500 samples stratified selected from the original dataset.

As for nearest neighbor classifiers, in the training stage, for each training set, we build a SASH structure [1] for fast estimation of nearest neighbors. In the testing stage, for each test image, we make a query using this image to the images in the training set and select top 100 nearest neighbors. The

[1]http://www.csie.ntu.edu.tw/$\sim$cjlin/libsvm

prediction scores is computed as follows:

$$fScore = \frac{fDistNeg - fDistPos}{fDistNeg + fDistPos};$$

where $fDistNeg$ and $fDistPos$ are the nearest distances from the query image to negative and positive images in the nearest neighbor set described above.

B. Result

We submitted 6 runs and the results are shown in Table I. There are two type a runs among the 6 runs that only use TRECVID 2005 dataset. Our best run is NII-2-A (MAP 0.088), which is fusion of SVM classifiers trained on 3 baseline features using only TRECVID 2008 dataset. The training data set had significant effect on the final performance. For example, performance of NII-2-A using TRECVID 2008 training set (MAP 0.088) is much better than that of NII-3-a using TRECVID 2005 training set (MAP 0.037). Fusion of those of using different training sets made performance a bit worse (NII-1-A (MAP 0.082) vs NII-2-A (MAP 0.088)).

In addition, our trial on using nearest neighbor classifiers is not so bad. It is close to the median. It is interesting to see NII-3-a run, which uses TRECVID 2005 training set with SVM, has the same performance with NII-5-A, which uses TRECVID 2008 training set with K-NN.

II. Search

A. Method Overview

We used ASR/MT transcripts for the required text-based run (NII-6-a). Specifically, for each shot, we extracted the transcript of that shot and that of the neighbor shots (5 shots before and after the current shot. In total, the transcripts of 11 shots were used). The transcripts were indexed using Lucene[2] built-in Solr[3], an open source for search using text. For each query, it text was used to retrieve and rank relevant shots.

We used the same features as in the high level feature extraction task for search by visual examples (NII-5-a). We extracted key frames from the example shots and compute the similarity scores between the query frames to all keyframes of the test set (we extracted five keyframes per shot for examples shots and test shots) using the above features and Euclidean distance. These scores of each feature are then fused and used to rank relevant shots.

In run NII-3-a, we first perform concept suggestion using the description of the query and 374 LSCOM concepts. We picked top 3 relevant concepts and their relevant scores. We used the prediction scores of these concepts on the test set for fusion. The fusion weights are the relevant scores returned by the concept suggestion stage. To implement the concept suggestion stage, we used Solr to index the text description of LSCOM concepts. For each query, its description was used to find relevant concepts. We used the system developed for the high level feature extraction task to train classifiers for the 374 LSCOM concepts.

[2]http://lucene.apache.org/java/docs/index.html
[3]http://lucene.apache.org/solr/

Most of our runs are type-a runs since we only use the annotation data of TRECVID 2005.

B. Result

We submitted 6 runs (fully automatic) and the results are shown in Table II. Our best run (NII-1-A) is close to the median (MAP 0.013). This run, which fuses the scores of text-based search and image-based search, achieved max with query 253. Our system usually has low recall while the precision of top 20 shots is quite good. Therefore, when the total number of relevant shots is small, our system is usually good. For example, query 253, there are only 17 relevant shots. We found 2. 1 of them is ranked in first 5 shots. It might explain the case of US-Flag of our system last year. There are only 6 relevant shots and we found 2 of them in the first 5 shots. The result of NII-4-A is bad since it has bug in our implementation. The result of NII-3-a is not good since we did not process the phrase ”one or more people ...” in the query text that causes the same suggested concepts for all queries having this phrase. Our best type-a run NII-2-a achieved promising result with MAP 0.011.

III. Content-Based Copy Detection

This section reports our experiments on the content-based copy detection pilot task of TRECVID 2008. In these experiments, we have addressed two approaches that are based on the pattern of motions in feature point trajectories and matches of all frame pairs using normalized cross correlation (NCC). As for the former one, we investigate the following issues: (i) whether motion features are appropriate for the copy detection task? (ii) whether the fusion of spatial registration based on local feature can help to improve the final detection performance? As for the latter one, we perform all pairwise comparison between frames of database videos and frames of a reference video and investigate how efficient this approach is.

A. Motion-Based Approach

This approach is composed of an offline process and an online process (Fig. 1). The offline process first decomposes given videos into shots by comparing RGB histograms of consecutive frames. KLT tracker is applied to obtain trajectories from each shot. From each trajectory, the approach extracts inconsistency sequence and discontinuity sequence (these terms will be explained later). Note that a shot may correspond to several trajectories (in our case up to 200 trajectories), and each trajectory is then associated with an inconsistency sequence and a discontinuity sequence.

The online process then matches all pairs of shots in the archive to obtain copies. Given a pair of shots, the approach evaluates the similarity between shots assuming all possible temporal offsets. The similarity between a pair of shots is then evaluated by matching all possible combinations of pairs of trajectories between the two shots. If a certain fraction of the trajectory pairs are similar to some extent, the similarity between the shots will become close to one (match). The

RunID	Description	MAP
A_NII-1-tv08+05-svm-bl_1	NII-1-A: Fusion of TV2008 and TV2005 devel sets - Classifier: SVM - Feature: G_CM+G_EOH+G_LBP - Total: 24 classifiers per concept	0.082
A_NII-2-tv08-svm-bl_2	NII-2-A: Fusion of TV2008 devel sets - Classifier: SVM - Feature: G_CM+G_EOH+G_LBP - Total: 12 classifiers per concept	0.088
a_NII-3-tv05-svm-bl_3	NII-3-a: Fusion of TV2005 devel sets - Classifier: SVM - Feature: G_CM+G_EOH+G_LBP - Total: 12 classifiers per concept	0.037
A_NII-4-tv08+05-knn-bl_4	NII-4-A: Fusion of TV2008 and TV2005 devel sets - Classifier: KNN - Feature: G_CM+G_EOH+G_LBP - Total: 24 classifiers per concept	0.031
A_NII-5-tv08-knn-bl_5	NII-5-A: Fusion of TV2008 devel sets - Classifier: KNN - Feature: G_CM+G_EOH+G_LBP - Total: 12 classifiers per concept	0.037
a_NII-6-tv05-knn- bl_6	NII-6-a: Fusion of TV2005 devel sets - Classifier: KNN - Feature: G_CM+G_EOH+G_LBP - Total: 12 classifiers per concept	0.014

TABLE I
THE PERFORMANCE OF NII'S RUNS FOR THE HLF TASK.

RunID	Description	MAP
F_a_2_NII.R1-FuseAll_1	NII-1-A: Fusion of NII-3-a, NII-4-A, NII-5-a and NII-6-a	0.013
F_a_2_NII.R2-VisualConceptFusion_2	NII-2-a: Fusion of NII-3-a and NII-5-a	0.011
F_a_2_NII.R3-NearbyLSCOMConcept_3	NII-3-a: Concept suggestion using Solr and 374 LSCOM concept detectors	0.007
F_a_1_NII.R4-LearnedDistance_4	NII-4-A: Fusion similarity scores based on the global distance, which is learned by RCA	0.001
F_a_1_NII.R5-VisualBL_5	NII-5-a: Fusion similarity scores based on 3 baseline features G_CM+G_EOH+G_LBP	0.010
F_a_1_NII.R6-TextBL_6	NII-6-a: Text only - Search by Solr	0.012

TABLE II
THE PERFORMANCE OF NII'S RUNS FOR THE SEARCH TASK.

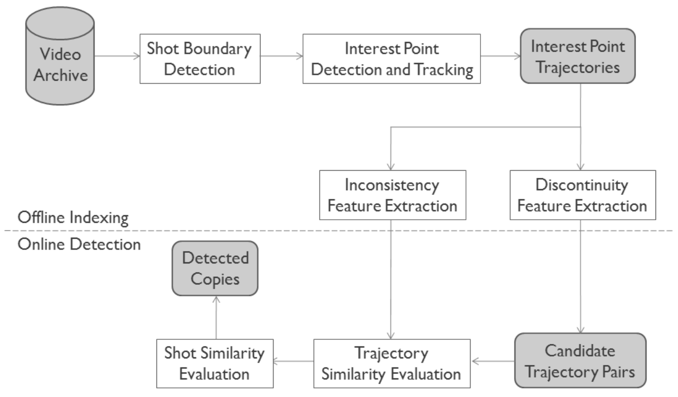
Video Archive
Shot Boundary Detection
Interest Point Detection and Tracking
Interest Point Trajectories
Inconsistency Feature Extraction
Discontinuity Feature Extraction
Offline Indexing
Online Detection
Detected Copies
Shot Similarity Evaluation
Trajectory Similarity Evaluation
Candidate Trajectory Pairs

Fig. 1. Framework overview.

inconsistency sequences and discontinuity sequences generated in the offline process are referred to for evaluating the similarity between trajectories.

1) Feature Calculation: To detect motion patterns, we use inconsistency [3], which is known to work well on motion estimation from video sequences of complex dynamic scenes. Image is decomposed into many small spatio-temporal patches($7[pixels] \times 7[pixels] \times 3[frames]$). For all small patches containing a single uniform motion, the inconsistency will be close to 0. For all small patches located at spatio-temporal motion discontinuities or changes in the motion direction or velocity, the inconsistency will be close to 1.

In the original study [3], image is decomposed into many small patches, and inconsistency is computed from all patches. This requires excessive computational burden. In our work, we first extract trajectories from shots, and only extract inconsistency values of patches on the trajectory (Fig. 2). This results in a sequence of values for each trajectory. We call such a sequence an inconsistency sequence, $c(t; T_i^j)$ of trajectory T_i^j, expressing the inconsistency value at time t. Fig. 3 shows an example of inconsistency sequences. After smoothing an inconsistency sequence with a Gaussian, we detect the local maxima and regard them as discontinuities. We generate a discontinuity sequence $d(t; T_i^j)$ of T_i^j as a binary sequence where 1 corresponds to a maximum (discontinuity) and 0 otherwise. This pattern captures spatio-temporal motion discontinuities or changes in the motion direction or velocity.

2) Temporal Registration: We use these two sequences for further video matching. The basic idea is to check whether the motion discontinuities of each trajectory occur at almost the same timing. Given two trajectories, with trajectory T_1 fixed,

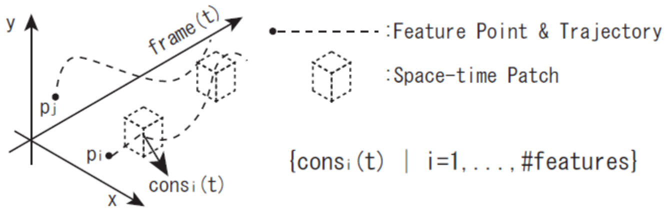
y
frame(t)
x
p_j
p_i
cons_i(t)
:Feature Point & Trajectory
:Space-time Patch
{cons_i(t) | i=1, . . . , #features}

Fig. 2. Calculation of features for trajectories.

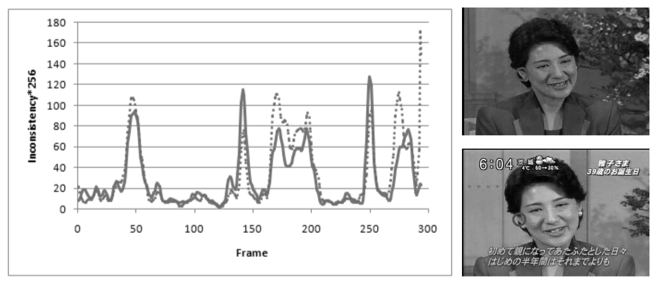

Fig. 3. Inconsistency of feature points.

The graph shows two inconsistency sequences corresponding to the feature points of two duplicate shots taken from different viewpoints. Since the feature points are in correspondence, the two inconsistency sequences have almost identical temporal patterns.

trajectory T_2 is gradually slid by changing the temporal offset τ (Fig. 4). If there is at least 1 frame where discontinuities match, we compute the similarity between these two trajectories.

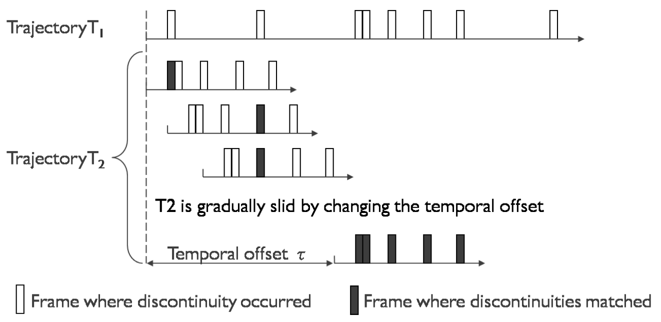
TrajectoryT₁
TrajectoryT₂
T2 is gradually slid by changing the temporal offset
Temporal offset τ
Frame where discontinuity occurred
Frame where discontinuities matched

Fig. 4. Detection of matching discontinuities.

3) Shot Similarity: To evaluate the similarity of trajectories T_1 and T_2, we compute the local normalized cross correlation centered at all discontinuities within a certain window width w, and the average of these values is used as the similarity of trajectories. The similarity of trajectories T_1 and T_2 given a temporal offset τ is then defined as follows:

$$Sim(T_1||T_2;\tau) = \frac{\sum_t (d(t;T_1)+d(t-\tau;T_2))NCC(c(t;T_1),c(t-\tau;T_2);t-w,t+w)}{\sum_t (d(t;T_1)+d(t-\tau;T_2))} \quad (1)$$

$$NCC(c(t;T_1),c(t;T_2);t_1,t_2) = \frac{\sum_{t=t_1}^{t_2}(c(t;T_1)-\overline{c(T_1)})(c(t;T_2)-\overline{c(T_2)})}{\sqrt{\sum(c(t;T_1)-\overline{c(T_1)})^2\sum(c(t;T_2)-\overline{c(T_2)})^2}} \quad (2)$$

where $NCC(c_1,c_2;t_1,t_2)$ is the normalized cross correlation between inconsistency sequences for $t_1 \le t \le t_2$, $\overline{c(T_i)}$ is the average of $c(t;T_i)$ for $t_1 \le t \le t_2$. NCC is computed at frames where $d(t;T_1)=1$, $d(t-\tau;T_2)=1$ or both, and averaged.

On the basis of the similarity between trajectories, we then define the similarity between a trajectory and a shot, i.e., a set of trajectories. From here onwards, the temporal offset τ is omitted for the sake of readability. The similarity between the j^{th} trajectory in S_1 (T_1^j) and another shot (S_2) is defined as the similarity between the trajectory T_1^j and the most similar trajectory among T_2^k:

$$Sim(T_1^j||S_2) = \max_k Sim(T_1^j||T_2^k) \quad (3)$$

The similarity between shot S_1 and S_2 is defined as follows:

$$Sim(S_1||S_2) = \underset{\text{top } \rho\%}{\text{avg}}\ Sim(T_1^j||S_2) \quad (4)$$

$$Sim(S_1,S_2) = \frac{1}{2}(Sim(S_1||S_2)+Sim(S_2||S_1)). \quad (5)$$

Among j, the top ρ% of $Sim(T_1^j||S_2)$ will be used for the average. This is to discard noises (outliers) due to occlusion of telop or captions.

4) Fusion with Local Feature Registration: We use a local-feature-based method (LIP-IS+OOS) [2] as preprocessing. LIP-IS+OOS is known to work well on distinguishing near duplicate from noisy video pairs that are entirely visually different. Because it uses spatial coherence as the video matching criteria, LIP-IS+OOS is much more sensitive to the difference between completely unrelated videos. We use LIP-IS+OOS to generate as many copy candidates as possible. The motion-based approach is used with only these candidates as the target.

We extract multiple keyframes from each shot in the test data, and extract the middle keyframe from each shot in the query data. In the case of the test data, the shot length is equally divided, and the frames at the points of division are selected as the keyframe. In equation terms, given the shot length L, the $\frac{i \cdot L}{N+1}^{\text{th}}$ frames are extracted as the keyframe, with $i = 1 \cdots N$ and $N = 3$. The similarity between the query shot and the test shot is evaluated by calculating the maximum similarity among all possible keyframe pairs of the two shots:

$$Sim(S_{QUERY},S_{TEST}) = \max_i Sim(K_{QUERY},K_{TEST}^i) \quad (6)$$

5) Experiments: We tested our approach with the datasets of TRECVID 2008. All experiments were performed on a workstation (Dell Precision T7400, 2.83 GHz, 16 GB RAM, $4GB \times 4$, Windows Vista(R) Ultimate). All code was written in C++ and compiled by GCC.

The experiment required certain parameters to be set first. The KLT tracker needs the number of feature points to be tracked. This value was set to 200, as mentioned before. The approach required two shots to overlap by more than a certain number of frames θ_o [frames]. If the overlap is less than $\theta_o = 120$, the similarity between the pair of shots is defined to be zero (no match). A shot was considered to be lacking enough motion information and unsuitable for our trajectory-based approach if it contained $\theta_t = 20$ trajectories or less for which the discontinuities were more than $\theta_{DoT} = 5$. The window size to calculate NCC was set to $w = 14$ (totally $2w + 1 = 29$ frames). The window size to calculate the local maxima of inconsistency sequences to obtain the discontinuity sequences was set to $w = 5$ (totally $2w + 1 = 11$ frames). ρ which was used to calculate the similarity between shots was set to $\rho\% = 50\%$. All parameters including the ones mentioned above were determined empirically.

The evaluation results on detection accuracy, location accuracy, and copy detection processing time are shown in Fig 5, Fig 6, and Fig 7.

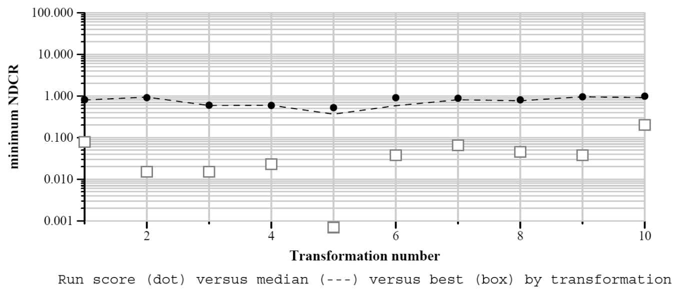

Fig. 5. Detection accuracy.

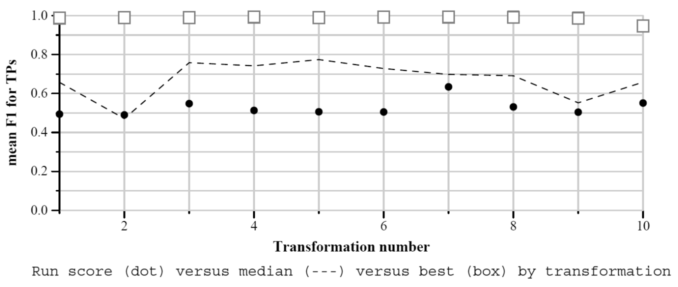

Fig. 6. Location accuracy.

6) Discussion: **Detection Accuracy** From Fig. 8, we found that the miss probability of our motion-based approach is higher relative to the false alarm rate. In other words, our motion-based approach tends to obtain higher precision and lower recall. There are two possible reasons: (1) excessive filtering of LIP-IS+OOS; (2) shot-based implementation; (3) dependence of motion.

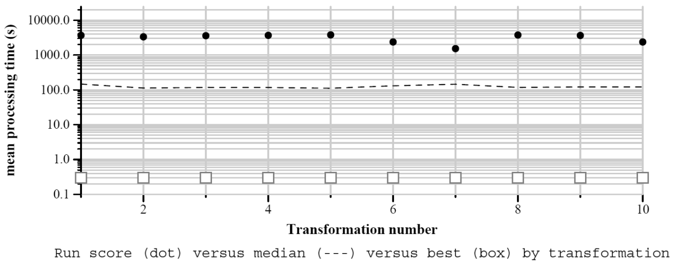

Fig. 7. Copy detection processing time.

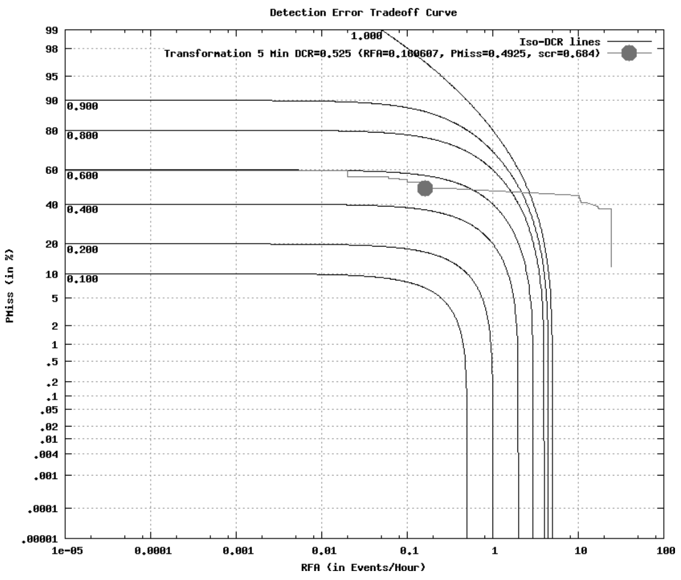

Fig. 8. DET curve of transformation 5.

In this work, we used LIP-IS+OOS for removing noise and filtering out entirely visually different video pairs. 279,805,680 pairs were reduced to 1,053 pairs so that many copies might be excessively filtered out. This problem can be improved by further parameter tuning. Second, our approach is shot-based. Too short shots, shots with too few trajectories, and shot pairs, the overlapping part of which is too short, are all rejected. Especially shaky videos, e.g. home videos, videos with frequent flashes, or cartoon films, tend to generate very short shot. As a result, 1,053 pairs are reduced to 773 pairs. Many useful and informative trajectories might be rejected so that corresponding copies cannot be successfully detected. This problem can be improved by more reasonable implementation. Finally, our approach depends on motion so that videos of static scene, where no enough motion information is available, are not suitable for this approach.

Location Accuracy By observing the experimental result, we found that in terms of location accuracy, the mean precision of our approach is 89.90% while the mean recall is only 41.16%. This might be due to our shot-based implementation.

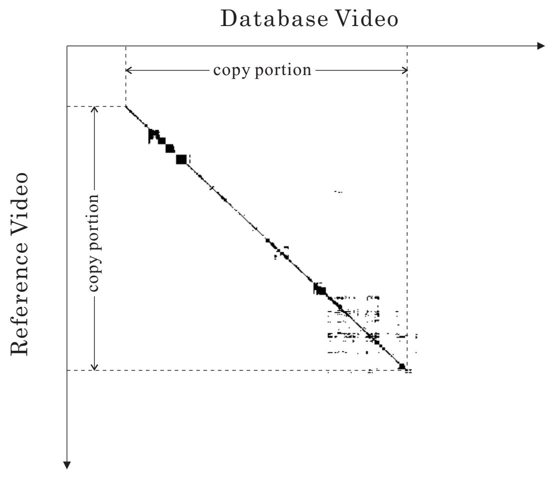

Fig. 9. Frame-to-frame matching matrix

Especially because many short shots were rejected in filtering steps, copies detected by our approach are always composed of discontinuous shots. Therefore, we need a more reasonable implementation on estimation of temporal offset between the query data and the test data.

Because the motion pattern used in this work is relatively invariant on scaling, translation, viewpoint change, etc., the performance of the motion-based approach is independent on transformation type. Therefore, from Fig. 6, we can see that the median location accuracy varies from transformation to transformation while the performance of our approach is almost the same over all transformations.

B. *Frame-based Video Copy Detection*

1) Method Overview: Basic idea of this video copy detection method is first to perform all pairwise comparison between frames of database videos and frames of a reference video. Assume that an N-frames database video and an M-frame reference video is given. After comparison of all frame pairs finishes, the matching result can be represented by an M-by-N matrix, whose elements show if a corresponding frame pair matches (1 or black) or does not match (0 or white). Figure 9 shows an example of matrix. As shown in the figure, video copy portions can be observed as diagonal lines. Issues are how to obtain matching result of frame pairs, and how to search diagonal lines in the matrix.

Frame pairs is matched by using normalized cross correlation (NCC). No color information is used, and thus frame images are first converted into grey scale images. NCC computation requires pixel-by-pixel comparison and thus computationally very intense. So, to approximate NCC, we convert (grey scale) images into normalized intensity histogram (NIH) [4], [5], and image similarity is calculated by L1 distance between NIH. It was proved that NIH can well approximate NCC.

Then the matrix is scanned to find diagonal line segments. This is basically done by raster scanning of the matrix. However, since the matrix is typically very sparse and the diagonal lines may be thick to some extent, we scan the matrix at a fixed interval. The method scans elements of every 32 frames vertically and horizontally, and it finds element 1 (matched frame pair), the method start scanning diagonally to find start and end points of the line segment.

2) Discussion: Since the method is basically based on NCC, i.e., pixel-by-pixel comparison, the method is very sensitive to geometrical distortion such as picture-in-picture, cropping, etc. Therefore, the method achieves very poor performance for T1-3, and the combination of these, T8-10. The method shows relative robustness to T4-7, where intensity-based distortion is added such as contrast change, blur, noise, etc. Obviously handing of geometric distortion is needed. Since the method requires comparison of all frame pairs, and then raster scan of matrix showing frame matching result, it is very slow. Intense speedup is desired.

References

[1] M. E. Houle and J. Sakuma, "Fast approximate similarity search in extremely high-dimensional data sets," in *Proc. Int. Conf. on Data Engineering (ICDE)*, 2005, pp. 619–630.

[2] C.-W. Ngo, W. Zhao, and Y.-G. Jiang. Fast tracking of near-duplicate keyframes in broadcast domain with transitivity propagation. In ACM Multimedia, pages 845–854, 2006.

[3] E. Shechtman and M. Irani. Space-time behavior based correlation. In CVPR, pages 405–412, 2005.

[4] Fuminori Yamagishi, Shin'ichi Satoh, Takashi Hamada, and Masao Sakauchi, "Identical Video Segment Detection for Large-Scale Broadcast Video Archives," *Proc. of International Workshop on Content-Based Multimedia Indexing (CBMI'03)*, pp. 135–142, 2003.

[5] Fuminori Yamagishi, Shin'ichi Satoh, and Masao Sakauchi, "A News Video Browser Using Identical Video Segment Detection," *Proc. of Pacific-Rim Conference on Multimedia (PCM2004)*, pp. 205–212, Vol. II, 2004.

[6] Sheng Tang, et al., "TRECVID 2008 High-Level Feature Extraction By MCG-ICT-CAS," *Proc. TRECVID 2008 Workshop,Gaithesburg, USA , Nov. 2008.*