KDDI LABS AND SRI INTERNATIONAL AT TRECVID 2010: CONTENT-BASED COPY DETECTION

Yusuke Uchida, Shigeyuki Sakazawa

KDDI R&D Laboratories Inc.
2-1-15 Ohara, Fujimino-shi
Saitama, Japan

Motilal Agrawal, Murat Akbacak

SRI International
333 Ravenswood Avenue
Menlo Park, CA, USA

ABSTRACT

We describe our system for Content-Based Copy Detection (CBCD) task submitted to TRECVID 2010. Our system is multi-modal and integrates the results of global visual features, local visual features and audio features to produce the final run results. Each of these features is designed to take care of different aspects of the video transformations. We submitted two runs each for BALANCED as well as NOFA profile:

- KDDILabs-SRI.m.balanced.1
- KDDILabs-SRI.m.balanced.2
- KDDILabs-SRI.m.nofa.1
- KDDILabs-SRI.m.nofa.2

These runs all use the same CBCD framework for each of the three modalities and differ only in the parameters for the final integration step. Our CBCD framework has made significant advances in video based copy detection. We introduce novel algorithms to obtain robust results against various transformations: dense-sampling-based global SIFT features, improved indexing methods for both global and local features and handling temporal burstiness. TRECVID 2010 evaluation results show that our system achieves good performance for both detection accuracy especially on NOFA profile and localization accuracy.

1. INTRODUCTION

In recent years, Content-Based Copy Detection (CBCD) technology, more generally referred to as *digital fingerprinting*, has attracted considerable research attention. There are many applications of CBCD technology such as detecting copyright infringement on video sharing sites, content identification and tracking, or query-by-example searches.

Given a test collection of videos and a set of queries, the goal of the CBCD task in TRECVID is to determine for each query the place, if any, that some part of the query occurs, with possible transformations, in the test collection.

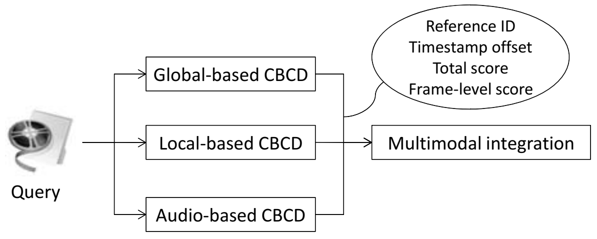


Fig. 1: Framework of our CBCD system.

Figure 1 gives an overview of our CBCD system. Our system runs global, local and audio based CBCD simultaneously for each query. Each of the three modes produces results independently in the form of a reference video identifier, corresponding timestamp offset, total score and frame-level scores. The results of each of the modalities are integrated in a post-processing stage to make the results reliable and accurate. We fuse these results using a voting scheme that emphasizes results where at least two of the three modes are consistent.

The basic idea behind using multiple modalities for CBCD task stems from the fact that each of these modes work well for different transformations that are in some sense complementary. While it is apparent that audio and video have completely different and complementary properties, it is also easy to see that global and local features have different characteristicts: local features are robust even for geometric transformations (T1, T2, T8, T10), while global features are more robust for non-geometric (photometric) transformations (T4, T5, T6). In addition, global features are much faster to compute and will work well for all transformations with small or no-change in geometry. Our audio features are simple and fast although we believe that their performance could be greatly enhanced by a tighter integration with video and use of more robust audio features.

In the following sections, we describe each of the three modalities in some detail and finally discuss the integration framework and evaluation results.

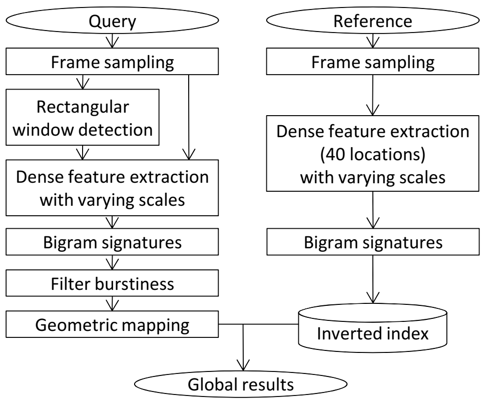


Fig. 2: Overview of global feature based CBCD.

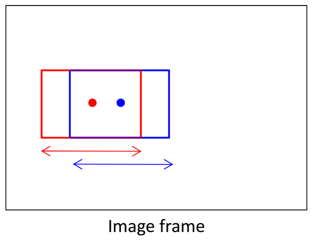


Fig. 3: Two neighboring windows corresponding to a scale of 1/3 the width and height. The feature locations (shown as circles) are such that the windows overlap 75% between the neighbors.

2. GLOBAL FEATURE BASED DETECTION

Figure 2 illustrates a functional diagram of our global feature based CBCD scheme. Our global feature based detection relies on dense sampling of the image at fixed locations. Our global features are obtained by densely sampling the frame at multiple fixed locations in the image. The scale of the features dictates the window size to be used for computing the descriptor at that feature location. We have used scales that correspond to window sizes greater than or equal to one third the total width or height.

For a given scale, feature locations are chosen such that the neighbors in the x and y directions overlap at least 75%. This results in a total of 121 windows although our approach for TRECVID only used a subset of 40 such windows. Figure 3 shows two such neighboring feature locations for a scale of 1/3 the width and height of the image. In comparison to local features, we do not have to perform feature detection and scale selection of the selected features, thereby making it faster. As discussed earlier, global features however are less robust to drastic changes in geometry.

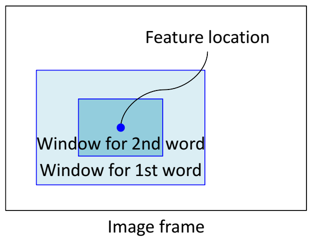


Fig. 4: The two nested windows for making a bigram feature descriptor at a given feature location and scale.

2.1. Bigrams as feature descriptors

Conventional global descriptors such as rank-based features [1, 2] are not distinctive enough in our dense sampling strategy. SIFT descriptors [3] on the other hand are known to perform well. We used bigrams of SIFT descriptors to make it more distinctive.

Given a feature location and scale, we extract two 128 dimensional SIFT descriptors from the window corresponding to that scale and use the bigrams as the descriptor for that feature location and scale. The first descriptor is extracted from the full window and the second descriptor is extracted from a sub-window centered on the feature location with a width and height half of that of the full window. We quantize each of the two descriptors into 10,000 words independently and the global descriptor for that feature location and scale is then represented by their bi-grams. For a vocabulary size of 10,000 for each of the words, this results in a bigram vocabulary size of 100 Million. We have found that using the bigram results in much better performance and also results in faster execution time. Figure 4 shows the two nested windows for bigram computation of a feature location and scale.

2.2. Indexing with multiple assignments

We use the bigrams to make an inverted index and index each frame of the reference video. Use of bigrams results in a large vocabulary size but in order to increase the recall, we use multiple assignments for indexing. We have found that using multiple assignments in both query and reference side helps improve the recall. Each descriptor in the bigram is assigned to five words. Therefore each bigram in both the reference and query side gets assigned to 25 words.

2.3. Automatic geometric correspondence

On the query side, global features are computed in a manner similar to the reference video. When querying, for each feature location in the query video, its correspondence is with the same feature location and scale in the reference video. Therefore we do not have to perform any matching step of the feature locations. In order to make the approach robust

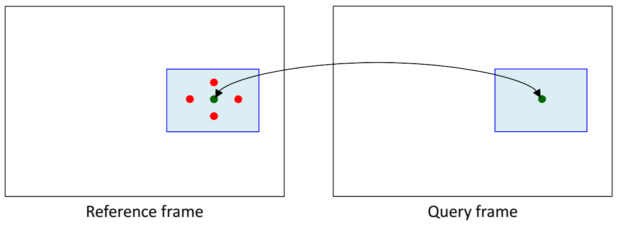


Fig. 5: Automatic correspondence determination based on position and scale. The green features are corresponding locations. The red feature locations are neighbors in the reference frame and are included to make the matching robust to small shifts.

to small shifts we also match it with its four neighbors in the reference video with the same scale.

Figure 5 demonstrates these 4 neighbors in the reference video.

2.4. Special case: Picture-in-Picture detection

Since our global features are not scale invariant, they would not work well for cases where the frame undergoes drastic changes in scale. Therefore our approach described so far will not work well for the picture-in-picture (PIP) transformation. Our approach detects picture-in-picture and pattern insertion transformations by detecting rectangular windows in the query video. Windows are detected by accumulating image gradients in the x and y directions over time. We run edge detection in each video frame with a very low threshold and then accumulate the number of edges in each row and column temporally. Rows and columns with sufficient number of accumulated edges are candidates for the edges of the window. The intersection of a pair of row and column edges is a candidate corner for the rectangular window. Finally, those windows with sufficient number of edges in at least two of the four sides are the candidate detected windows. In our experiments, we have set thresholds such that our system hardly misses any such window. Of course, this means that sometimes it detects a lot more windows than actually present. Figure 6 show the detected PIP windows in a few frames.

If a rectangular window is detected in the query window (for a PIP transformation), the feature locations and scale are determined relative to the detected rectangular region instead of the whole frame. This is illustrated in Figure 7. When a PIP window is detected, querying is performed with the detected window in addition to the whole frame. This takes care of false PIP detections.

2.5. Querying

For each query frame, we use the 40 global features to find a corresponding match in the reference frames. The score of a match is simply the count of the number of global feature

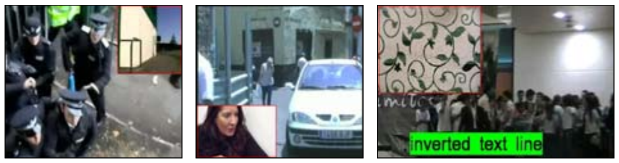


Fig. 6: Examples of detected PIP window.

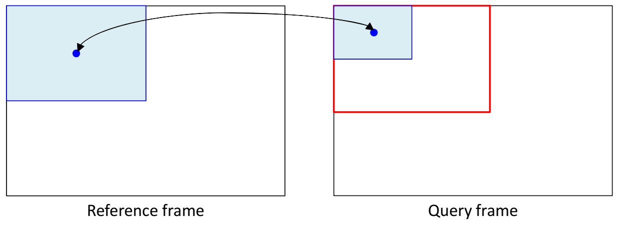


Fig. 7: Geometric correspondence for the PIP transform. Matching is performed relative to the detected window.

locations matched. Each such match results in a vote for the time offset corresponding to that match. The offset that gets the most votes is taken as the best match for that query.

If two consecutive frames are similar, they do not add new information for matching. This is the problem of temporal burstiness. If not accounted for, this will result in a lot of incorrect offsets. Temporal burstiness is easy to handle in our system. We compare the bigram from two consecutive frames and if either of the two words are the same, we do not use that frame for our matching. We skip all frames similar to the last good frame, until we find a frame with bigrams that are different from the last good frame.

2.6. Frame sampling

For both global and local features, each video is resampled at a fixed frame rate to extract key frames. This makes our reference database size tractable and also helps us deal with frame rate changes. In our submission, we sampled at 2.0 fps on both the query and reference side. For the reference set of approximately 400 hours, this produces $400\times60\times60\times2\simeq3$M (key)frames.

3. LOCAL FEATURE BASED DETECTION

Figure 8 illustrates a functional diagram of our local feature based CBCD scheme. We use local scale invariant SIFT features and their bag-of-features representations, which has been used widely in image/video retrieval areas [4–7]. The key idea here is (1) the use of USIFT feature descriptor for distinctiveness, (2) a novel variant of product quantization based indexing for more accurate nearest neighbor searches and (3) considering the temporal burstiness of local features in scoring to suppress false alarms.

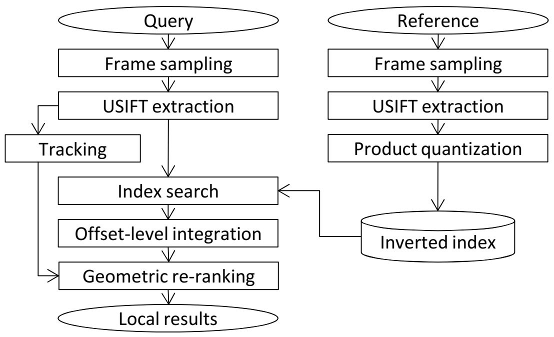


Fig. 8: Overview of local feature based CBCD.

3.1. USIFT extraction

For each keyframe, we extract the SIFT features and use upright SIFT (USIFT) as the feature descriptor. USIFT is more distinctive and faster to compute than SIFT [8]. Since none of the video transformations include a large rotation, USIFT can be used without degrading the robustness of the CBCD system. Each key frame is then represented by a set of features points (bag-of-features). Each feature point contains the following information: video identifier id (only for reference video frame), timestamp ts, position of feature point (x, y), and USIFT feature vector $\mathbf{f}$.

3.2. Product quantization based indexing

We use a novel version of the product quantization based method [9] to index USIFT feature vectors obtained in the USIFT extraction step. The product quantization based scheme can be integrated with an inverted index, referred to as IVFADC in [9]. A reference vector is first quantized by a coarse quantizer with the size of N (50K in our submission), then the residual vector from the corresponding centroid is encoded by product quantization into a short code.

In our CBCD system, we modify the original product quantization based method to use an arbitrary number of codebooks in product quantization, while a residual sub-vector is quantized by a single codebook in the original algorithm. The framework of our product quantization based indexing method is shown in Fig. 9. In the indexing step, codebooks used in the product quantization are switched according to the cell that the input vector is quantized into in the coarse quantization. Each cell (centroid) in the coarse quantizer has pointers[1] to the codebook by which each vector assigned to that cell should be quantized in product quantization. An arbitrary number (M) of codebooks for product quantization are created with the following procedure:

1. For each $n = 1, \cdots, N$, create a set of residual sub-vector R_n from all vectors assigned to the n-th cell in a coarse quantizer and assigned it to a random cluster $m \in \{1, \cdots, M\}$.

[1]Actually, there are the same number of pointers as decomposed sub-vectors (=8 in our submission).

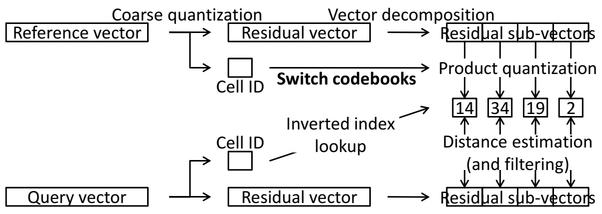


Fig. 9: Modified product quantization based indexing.

2. For each $m = 1, \cdots, M$, update codebook C_m by clustering all residual sub-vectors assigned to cluster m.
3. For each $n = 1, \cdots, N$, assign R_n to the cluster $\hat{m}$, s.t. the codebook $C_{\hat{m}}$ achieves minimum error in quantization of R_n.
4. Repeat Step 2 and Step 3 until convergence.

In the case of $M = 1$, our codebook becomes identical to the codebook used in [9]. Use of larger M reduces the quantization error in product quantization and improves the nearest neighbor search accuracy at the cost of an additional memory required. In our submission, residual vectors are decomposed into 8 sub-vectors and the number of codebooks M used in product quantization is set to 256. In this case, only 256×256×128≃8M byte memory is required to store codebooks.

3.3. Index search

In the search step, USIFT features in a query video are efficiently matched with reference features using an inverted index [4]. In addition, we can filter out many false matches through distance estimation in product quantization [9]. In our submission, we filter out reference features with a distance less than 0.3 (our SIFT features are normalized to a norm of 1.0). The result of the index search step is a set of matched keypoint pairs (Q, R), where each query keypoint Q has timestamp ts_q and coordinate (x_q, y_q), and each reference keypoint R has video identifier id, timestamp ts_r and coordinate (x_r, y_r).

3.4. Offset-level integration

Feature-level results obtained in the index search step are integrated into offset-level results using a voting scheme [7,10]. Every matched keypoint pair (Q, R) votes to the corresponding bin b$[id][ts_r - ts_q]$ in the 2D Hough space. After performing non-maxima suppression and thresholding, we obtain the top 200 hypothesis represented by $(id, \textit{offset})$, where

$offset = ts_r - ts_q$. Each hypothesis $(id, offset)$ has a list of matched keypoint pairs that have voted for the hypothesis, and this information is used for geometric verification. We also tried to incorporate the weak geometric consistency (WGC) method [6] using scale information [2], but it did not contribute accuracy in our preliminary experiments. This is probably because WGC based on scale is not as useful as one based on the orientation shown in [11].

3.5. Keypoint tracking

In the tracking step, query keypoints are tracked against keypoints in one and two previous frames. Then, each query keypoint Q has a list of keypoints Q' s.t. $0 < ts_q - ts'_q \leq 2$, $(x_q - x'_q)^2 + (y_q - y'_q)^2 < r^2$ and $||\mathbf{f}_q - \mathbf{f}'_q||^2 < th$. The lists are used in the geometric re-ranking step to alleviate the *temporal burstiness* effect.

3.6. Geometric re-ranking

Finally, we perform geometric verification and re-rank the result obtained in the offset-level integration step. For each result, we estimate the transformation matrix with 4 dof [5] between the query video and the reference video using the RANSAC algorithm and update the score by the maximum number of inliers. At the same time, temporal burstiness is taken into consideration in a similar way to [12,13]. The basic idea is to suppress bursty false matches that frequently appear in static scenes. For example, we try to put more emphasis on the right case in Fig. 10 than the left case where the same keypoint is matched sequentially. If the same keypoint matched K times in different frames, we tried three scoring strategies for this situation: (1) K, (2) $\sqrt{K}$, and (3) 1. Apparently, (1) produces the same score as the conventional voting scheme does. In our experiment, case (3), which seems to be the most extreme case, achieved the best performance, and we adopted this scoring strategy in our submission. Now, we obtain a list of results based on local features. Each result includes the reference video identifier, corresponding timestamp offset, total score and frame-level scores as described in Section 1.

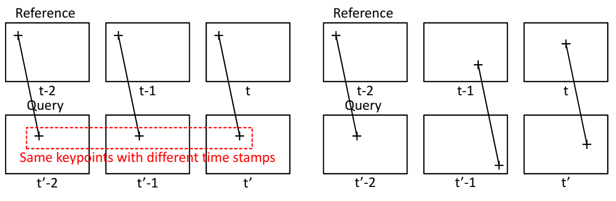
4. AUDIO FEATURE BASED DETECTION

For audio features, we have used binary filterbank energy differences [14] calculated as follows:

$$F(n,m) = \begin{cases} 1 & \text{if } EB(n,m) - EB(n,m+1) > 0 \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where $EB(n,m)$ denotes the energy in m-th filterbank for the n-th frame. Frame size is 25.6 msec, and the skip rate is 10 msec. Our raw filterbank energy features are 23 dimensional. Filterbanks are placed using a Mel-scale as shown in Fig. 11. However, in our evaluation we have found that the higher bands of this feature are noisy and decrease the performance of the system. Therefore, we are using only the lower 15 bands for this feature. This corresponds to a band limit of 5K.

[2]Orientations of matched keypoint pairs are always *consistent* in the USIFT-based scheme.



(a) Bursty case (b) Non-bursty case

Fig. 10: An example of the temporal burstiness effect. (a) The same feature is matched on successive frames. This might occur even between irrelevant videos especially in stationary scenes. (b) Different features are matched on successive frames. This rarely happen between irrelevant videos.

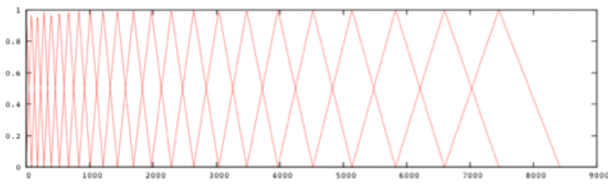


Fig. 11: Filterbank energy in the mel scale.

For binary filterbank energy, we used hashing scheme to make an inverted index [15]. In addition to using the 14 consecutive filterbank energy differences, we are also using 8 non-consecutive filter bank differences to make a 22 dimensional hash code. We have implemented multiple assignments for hashing (up to 32). Our hashing scheme is very fast and does not require quantization steps. Figure 12 shows the use of the first 15 bands from the 23 dimensional feature vector for hashing based indexing. The figure shows two non-consecutive filterbanks that are part of the 8 non consecutive hash codes. The search step includes the same voting scheme described in the local and global parts.

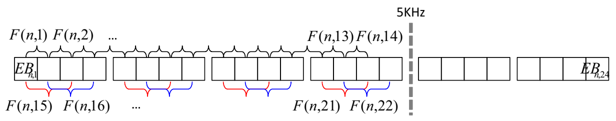


Fig. 12: Binary filterbank energy feature extraction.

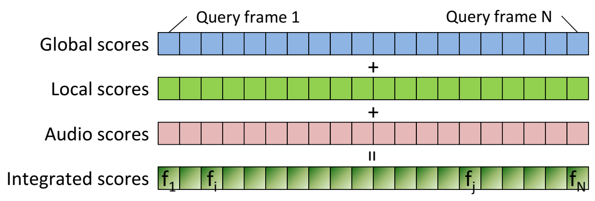


Fig. 13: Frame-level score integration.

5. MULTI-MODAL INTEGRATION

Each of the above three modalities produces a sorted list of results consisting of the reference video identifier, corresponding timestamp offset, total score and frame-level scores. These results are sorted by the total scores with the best result having the highest total scores. These individual results are integrated to produce a final list of results.

5.1. Integration and re-scoring

First, for each modality, all total scores are normalized by the second top score to emphasize distinctiveness. Then, any two or three results out of the three modalities that indicate the same id and $offset$ are integrated by simply summing the total scores. In addition, a result that has all three modes consistently receives an additional score of 5.0 (10.0) for the BALANCED (NOFA) profile, and a result that has two out of three consensus points receives an additional score of 3.0 (5.0) for the BALANCED (NOFA) profile. Frame-level scores are also integrated frame-by-frame after they are normalized so that the sum of the frame-level scores becomes 1.0 as shown in Fig. 13. We have found that this normalization step greatly improved the accuracy of the segment localization.

5.2. Segment localization

Finally, start frame $\hat{i}$ and end frame $\hat{j}$ of the copied segment in the query video is determined by

$$\arg\max_{i,j} S_{i,j}. \tag{2}$$

$S_{i,j}$ is the partial sum of frame-level scores from frame i to j normalized by the segment length:

$$S_{i,j} = \frac{\sum_{t=i}^{j} f_t}{\sqrt{j-i+1+\alpha}+\beta}, \tag{3}$$

where $f_1, \cdots, f_T$ indicate frame-level scores. Although it is solved in a brute-force manner, an integral image can drastically accelerate the computation. The equation somewhat includes heuristics trying to penalize any too short segments from the equation

$$\arg\max_{i,j} \frac{\sum_{t=i}^{j} f_t}{j-i+1}, \tag{4}$$

which results in the trivial solution $i = j = \arg\max_t f_t$.

6. CBCD EVALUATION RESULTS

Figure 14 shows the evaluation results of our run KDDILabs-SRI.m.nofa.1 for the NOFA profile. We found that our results based on the actual threshold (left) were not as good as the optimal results (right) in terms of NDCR. This might be because we determined our threshold using very small subset (20 hours) of the previous year's dataset. Instead, some of our results seem to have achieved the best performance in terms of optimal NDCR. This comes from our improvement in both global and local indexing and especially from an integration scheme where scores depends almost only on a consensus among modalities and distinctiveness, not on absolute voting scores. In Fig. 14, it is also clear that our scheme has achieved almost the best performance in segment localization criteria. This is because, in our framework, copied segments are localized in the integration step, not in each audio or video detection framework. Score normalization prior to localization might also contribute to the performance. In terms of computational cost, our scheme requires almost the same cost as the median for all participants. The bottleneck of our current implementation is quantization of query features, where features are quantized by a flat quantizer. This quantizer should be replaced by more efficient quantizers [5,6,16].

Figure 15 shows the evaluation results of our run KDDILabs-SRI.m.balanced.1 for BALANCED profile.

7. CONCLUSION

The system is based on global SIFT, local SIFT and audio features and efficient indexing methods. The evaluation results show that our system achieves good performance for both detection accuracy, especially in the NOFA profile and localization accuracy. Future work will include tunings and validation in a larger dataset.

8. REFERENCES

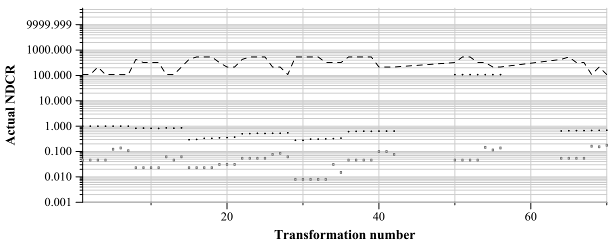
[1] X. Hua, X. Chen, and H. Zhang, "Robust video signature based on ordinal measure," in *Proc. of ICIP*, 2004, pp. 685–688.

[2] M. Usman and C. Kim, "Real time video copy detection under the environments of video degradation and editing," in *Proc. of ICACT*, 2008, pp. 1583–1588.

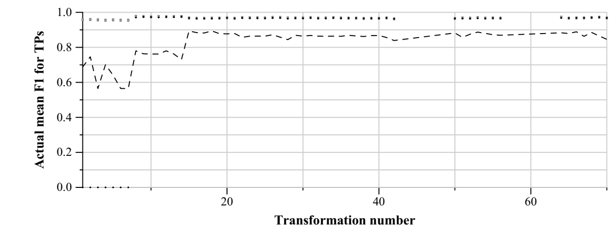
[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[4] J. Sivic and A. Zissermane, "Video google: A text retrieval approach to object matching in videos," in *Proc. of ICCV*, 2003, pp. 1470–1478.

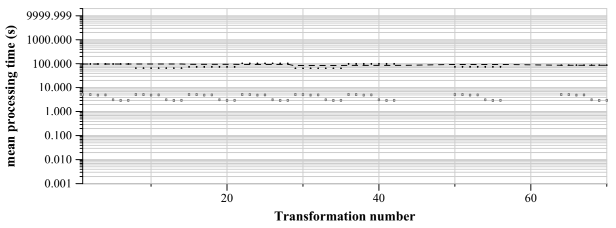
```
TRECVID 2010: copy detection results (no false alarms application profile)

Run name:                       KDDILabs-SRI.m.nofa.1
Run type:                       audio+video
```



Run score (dot) versus median (---) versus best (box) by transformation

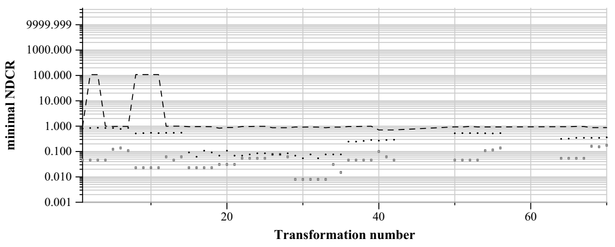


Run score (dot) versus median (---) versus best (box) by transformation

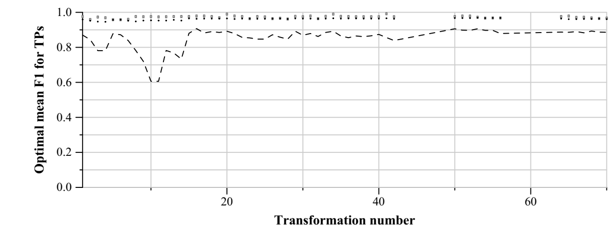


Run score (dot) versus median (---) versus best (box) by transformation

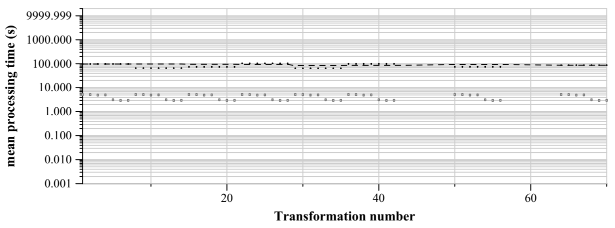
```
TRECVID 2010: copy detection results (no false alarms application profile)

Run name:                       KDDILabs-SRI.m.nofa.1
Run type:                       audio+video
```



Run score (dot) versus median (---) versus best (box) by transformation



Run score (dot) versus median (---) versus best (box) by transformation



Run score (dot) versus median (---) versus best (box) by transformation

Fig. 14: Our actual (left) and optimal (right) results for NOFA profile.

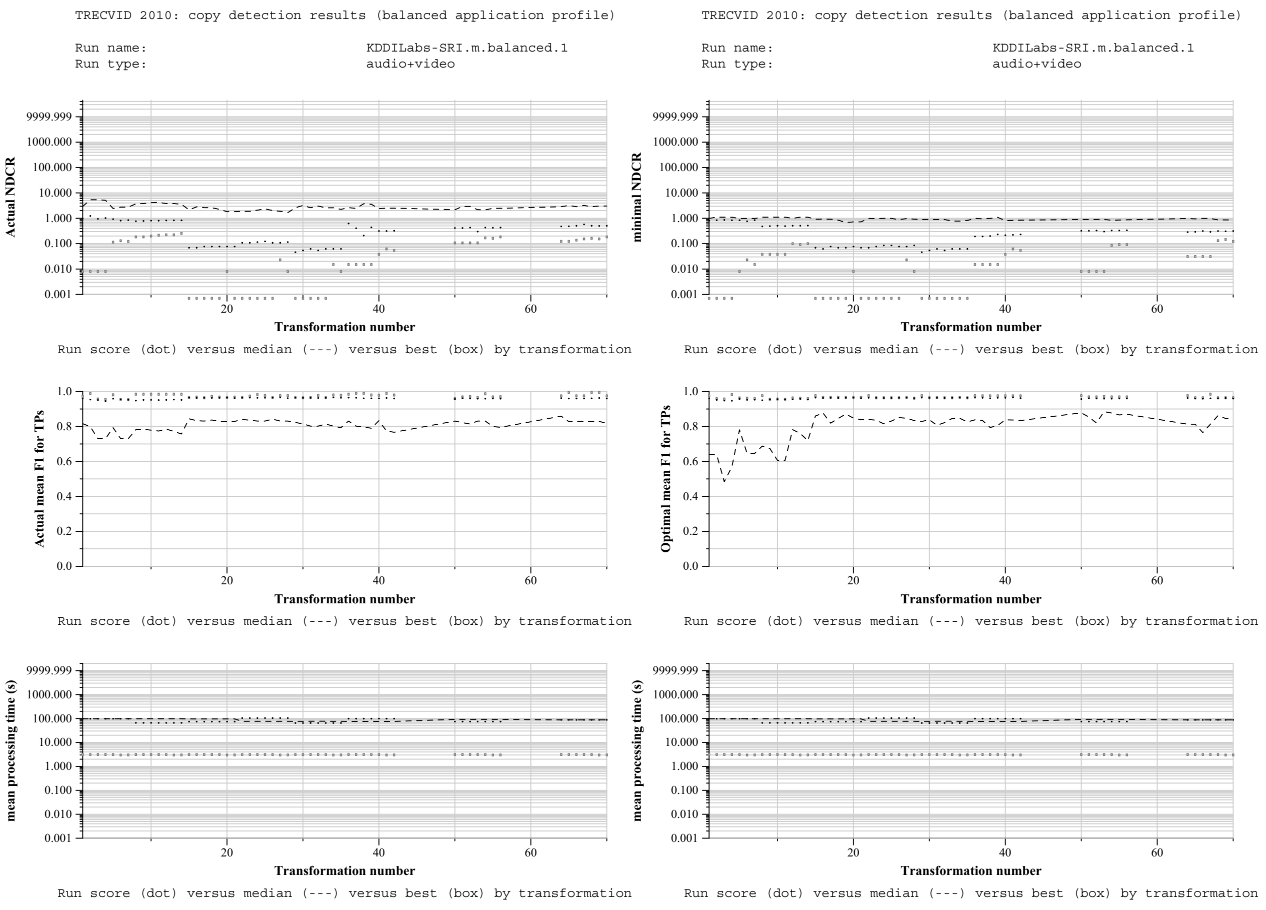


Fig. 15: Our actual (left) and optimal (right) results for BALANCED profile.

[5] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. of CVPR*, 2007.

[6] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. of ECCV*, 2008, pp. 304–317.

[7] M. Douze, H. Jégou, and C. Schmid, "An image-based approach to video copy detection with spatio-temporal post-filtering," *IEEE Trans. on Multimedia*, vol. 12, no. 4, pp. 257–266, 2010.

[8] G. Baatz, K. Koser, D. Chen, R. Grzeszczuk, and M. Pollefeys, "Handling urban location recognition as a 2d homothetic problem," in *Proc. of ECCV*, 2010.

[9] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. on PAMI*, 2010, *preprint*.

[10] J. Law-To, L. Chen, A. Joly, and I. Laptev, "Video copy detection: a comparative study," in *Proc. of CIVR*, 2007, pp. 371–378.

[11] S. S. Tsai, D. M. Chen, G. Takacs, V. Chandrasekhar, R. Vedantham, R. Grzeszczuk, and B. Girod, "Fast geometric reranking for image based retrieval," in *Proc. of ICIP*, 2010.

[12] H. Jégou, M. Douze, and C. Schmid, "Packing bag-of-features," in *Proc. of ICCV*, 2009.

[13] M. Douze, H. Jégou, C. Schmid, and P. Pérez, "Compact video description for copy detection with precise temporal alignment," in *Proc. of ECCV*, 2010.

[14] A. Saracoglu, *et al.*, "Content based copy detection with coarse audio-visual fingerprints," in *Proc. of CBMI*, 2009.

[15] J. Haitsma and T. Kalker, "Robust audio hashing for content identification," in *Proc. of CBMI*, 2001.

[16] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *Proc. of CVPR*, 2006.