NTT Communication Science Laboratories at TRECVID 2010 Content-Based Copy Detection

Ryo Mukai, Takayuki Kurozumi, Kaoru Hiramatsu
Takahito Kawanishi, Hidehisa Nagano, Kunio Kashino

NTT Communication Science Laboratories, NTT Corporation
3-1, Morinosato-Wakamiya, Atsugi-shi, Kanagawa 243-0198, Japan
{ryo, kurozumi, kawanisi, nagano, kunio}@cs.brl.ntt.co.jp
hiramatsu.kaoru@lab.ntt.co.jp

Abstract

In this paper, we describe our approaches that were tested in the TRECVID 2010 Content-Based Copy Detection (CBCD) task. We introduce a method consisting of a feature degeneration and sparse feature selection process for video detection tasks, which is highly robust as regards video signal distortion. For audio detection tasks, we adopt a method based on spectral partitioning to cope with additive interfering sounds. Both methods are key techniques for our Robust Media Search (RMS) technology, which has already been deployed for various commercial services. Evaluation results show the effectiveness of our methods.

1 Introduction

The task of content-based copy detection is to locate a fragment of a copy of known reference media content in response to a media content query. The copy detection of media data, or media search, has a wide range of applications including copyright management and enforcement, derivative tracking, advertising or recommendations. Media search technology examines the similarity between a "query signal," which is a fragment of audio or video, and a "reference signal," which is stored in a database. In media search, robustness is particularly important because media signals are often converted to various encoding formats, mixed with other signals such as background music, or even edited and re-edited into different versions. Search speed is also crucial, considering the rapidly growing volumes of audio and video being created, distributed and exchanged by individuals, public institutions and corporations around the world.

TRECVID 2010 CBCD task evaluates the performance of copy detection systems with a test set that includes various kinds of transformations. We participated in the campaign with one of the recent implementations of our media search technology called Robust Media Search (RMS).

2 Robust Media Search (RMS) Technology

The RMS is a core technology for content-based audio and video media search and identification developed by NTT. It has been used, for example, in copyright monitoring systems for video sharing sites on the Internet that can instantaneously detect the uses of known content [1]. RMS offers excellent robustness and a very high search speed by using coarsely-quantized features and spatiotemporal consistency.

As shown in Fig. 1, the audio and video signals are first converted to sequences of coarsely-quantized digits. Note that not all of those digits are necessarily used for matching; we found that appropriately choosing digits to be matched from among all digits not only simplifies the search but also greatly improves the robustness of the search, which is performed by matching those features. Specifically, the spatiotemporal accumulation of many such features enables us to achieve extremely high identification accuracy.

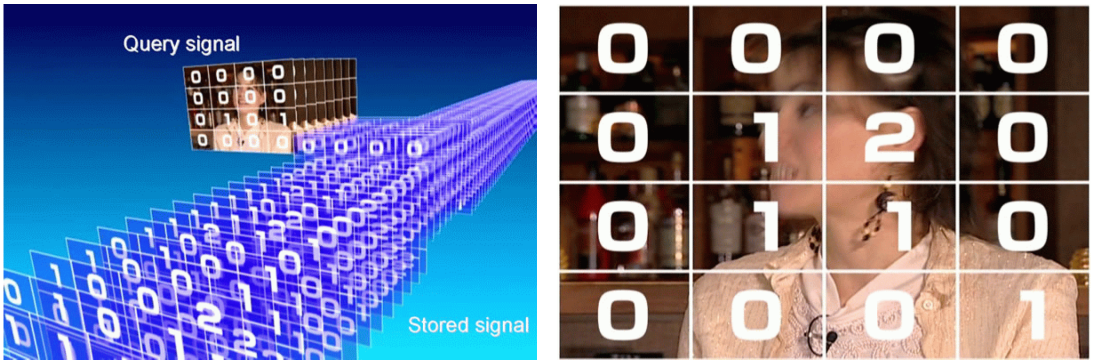


Figure 1: RMS basic principle (left) and example of coarsely-quantized video feature (right)

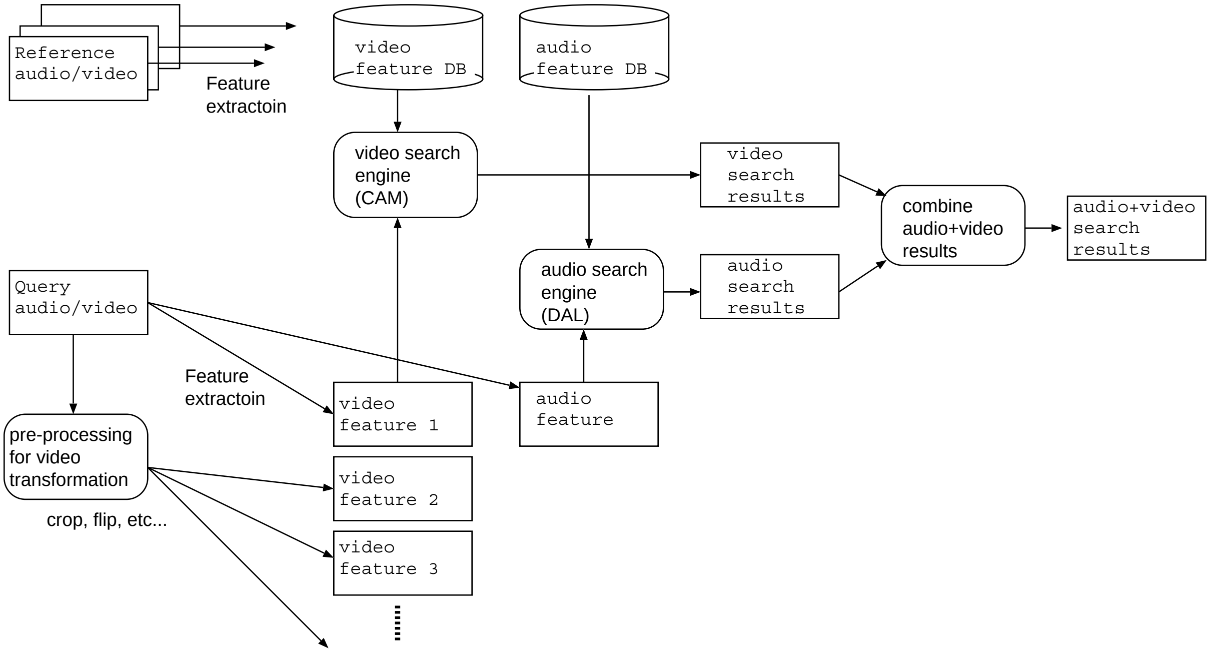


Figure 2: System Overview

2.1 System Overview

We tackled the TRECVID 2010 CBCD task with an RMS system configured for the task. The transformations in the TRECVID CBCD task includes more complex patterns than the transformations found in the real world, we introduced a pre-processing stage in the video feature extraction. Figure 2 shows the flow of our copy-detection system configured for the TRECVID 2010 CBCD task.

The reference audio/video data are converted into feature data and stored in a reference database. On the other hand, query video data are pre-processed by several transformations as described in 3.2. The extracted audio and video features are processed in an audio search engine and a video search engine, respectively. Finally, we merge the audio and video results to generate a submission run.

The rest of the paper describes the approaches we used in the video and audio search engines and the evaluation results.

3 Video Copy Detection

This section describes our approach to the copy detection task for video.

3.1 Coarsely-quantized Area Matching Method (CAM)

The Coarsely-quantized Area Matching method (CAM) is our video fingerprinting technology, which has excellent robustness against various kinds of distortion [2, 3]. The CAM method consists of the following five procedures.

3.1.1 Video feature extraction

Let $v_c(p,t)$ be the RGB value of pixels in the video data where $c \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$ denotes a color, p is a pixel coordinate and t denotes time or frame index. We adopt the raw RGB value of a reduced-size image of each frame in the video as a primitive video feature, that is,

$$x_c(i,t) = \frac{1}{|I_i|} \sum_{p \in I_i} v_c(p,t), \tag{1}$$

where I_i $(i = 1, 2, ..., W)$ is a whole set of pixels in the i-th sub image. Here, W is determined empirically.

3.1.2 Temporal local normalization

The i-th element of the normalized feature vector $\mathbf{y}_c(t) = [y_c(1,t), ..., y_c(W,t)]$ is defined as follows.

$$y_c(i,t) = \frac{1}{\sigma_c(i,t)} (x_c(i,t) - \mu_c(i,t)) \tag{2}$$

where

$$\mu_c(i,t) = \frac{1}{M} \sum_{j=-\lfloor M/2 \rfloor}^{M-\lfloor M/2 \rfloor -1} x_c(i,t+j), \tag{3}$$

and

$$\sigma_c(i,t) = \left(\frac{1}{M} \sum_{j=-\lfloor M/2 \rfloor}^{M-\lfloor M/2 \rfloor -1} (x_c(i,t+j) - \mu_c(i,t))^2 \right)^{1/2} \tag{4}$$

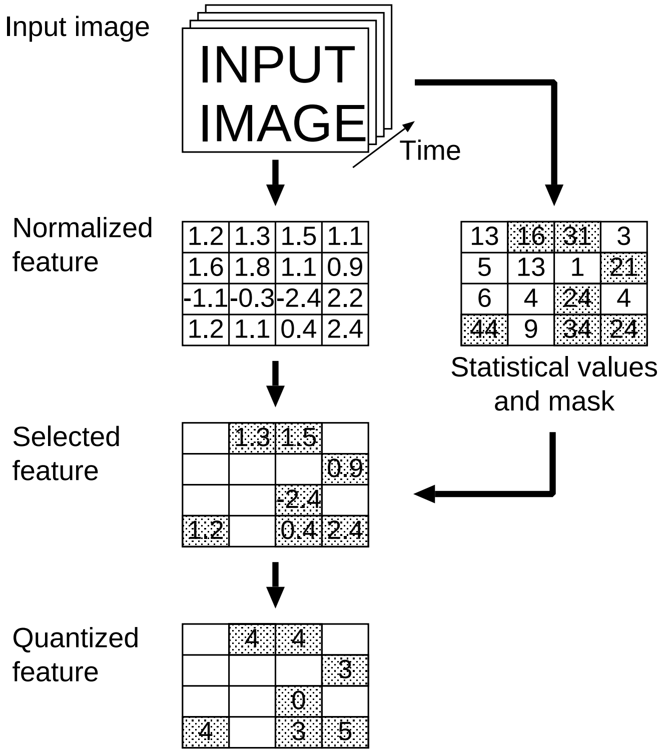
are an average and a standard deviation over a time window of size M. Here $\lfloor \cdot \rfloor$ denotes a flooring operation.

3.1.3 Feature selection

We assume that an area that has a large deviance from a temporal local average is a salient part. Based on this assumption, we select the top-N features for each frame with respect to the magnitude

$$z_c(i,t) = |x_c(i,t) - \mu_c(i,t)|. \tag{5}$$

Figure 3 shows the flow of this procedure.



1.2	1.3	1.5	1.1
1.6	1.8	1.1	0.9
-1.1	-0.3	-2.4	2.2
1.2	1.1	0.4	2.4

13	16	31	3
5	13	1	21
6	4	24	4
44	9	34	24

	1.3	1.5	
			0.9
		-2.4	
1.2		0.4	2.4

	4	4	
			3
		0	
4		3	5

Figure 3: Example of feature selection and quantization

3.1.4 Quantization

Next, we quantize selected feature values. The quantization is carried out on locally normalized feature values. There are many approaches to quantization including non-linear quantization and vector quantization. Here, we use a linear scalar quantization method for simplicity. The procedure is as follows:

$$y'_c(i,t) = \begin{cases} \lfloor \frac{Q\times(y_c(i,t)+r)}{2r} \rfloor & \text{if } |y_c(i,t)| < r \\ Q-1 & \text{else if } y_c(i,t) \geq r \\ 0 & \text{otherwise} \end{cases}, \qquad (6)$$

where r is the maximum number of quantization levels.

3.1.5 Time-series search

Finally, we perform a time-series search for the query feature in the reference feature database using codes obtained by quantizing locally normalized feature values. The database is scanned with a sliding window that has the same length as a query segment. Similarity is measured in terms of the Hamming distance between masked query codes and reference codes, that is, by counting the number of co-occurrences of the quantized codes at the corresponding positions in the window. This procedure is implemented very efficiently by using a hash table whose key is a pair consisting of the code and coordinates.

3.2 Pre-processing for multiple feature generation

The conventional CAM is robust against various kind of transformations and distortions such as insertions of pattern (T3), strong reencoding (T4), change of gamma (T5), blur, contrast, noise (T6, T7), and picture in picture type 2 (the original video is in the background). However it is weak against geometrical transformations such as picture in picture type 1 (T2, the original video is small) or flip, since the feature of conventional CAM is dependent on the positions of the pixel values. Frame dropping, or the insertion of black/white frame, also harms the CAM feature, because it causes a rapid change of intensity in the temporal direction and CAM mistakes it for a “feature”.

To cope with video transformations in TRECVID, we introduced a pre-processing stage including several transformations for generating multiple features. The stage was inserted before the video feature extraction stage described in 3.1.1. The operations in the pre-processing stage are as follows:

anti-frame drop: Discard a frame in which most of the sub-regions are approximately 0 or 1.

monochrome: Convert the (R,G,B) color value to (Y, Pb, Pr).

intensity normalization: Normalizing the intensity by the average and standard deviations over the whole frame. This process is effective for detecting slow or small movements.

flip: Flipping horizontally.

fixed crop: Cropping an image into sub-images. The combinations of the location and size of the sub-images are { ul, ur, bl, br, c }× { 0.5, 0.4, 0.3 }.

automated crop: Find vertical and horizontal lines and crop an image.

4 Audio Copy Detection

This section describes our approach to the audio copy detection task.

4.1 Divide And Locate Method (DAL)

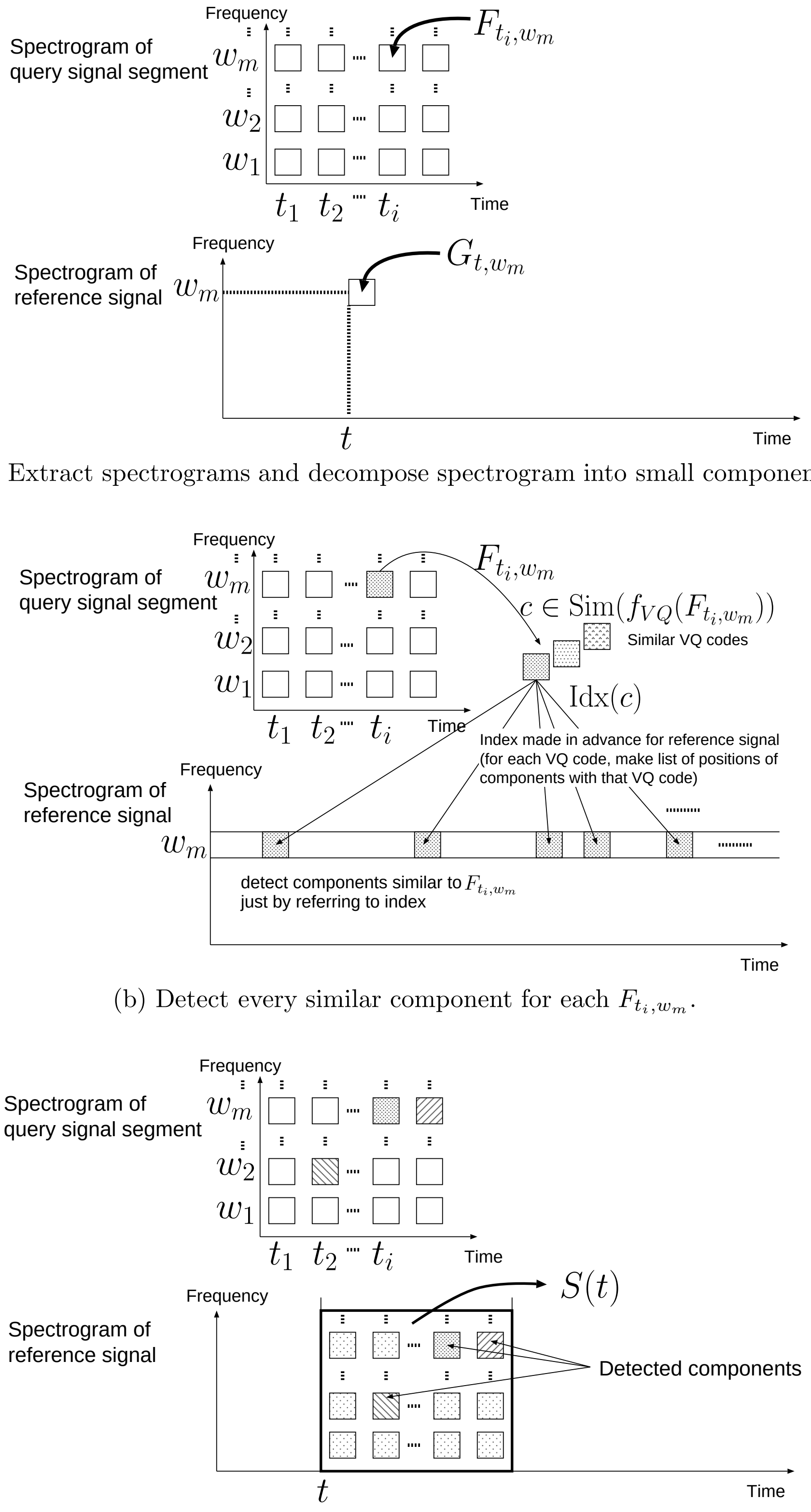
The Divide-And-Locate method (DAL) is an audio version of our RMS technology, which is especially robust as regards additive noise [4]. The basic idea of the DAL is to divide a spectrogram into a number of small regions and undertake matching for each region to locate it in the database. The small spectrum components are quantized by vector quantization (VQ), and the matching operation is executed by looking up a similarity table among the VQ codes and scanning index lists.

The outline of the DAL method is as follows. First, time-frequency power spectra are extracted for reference signals. Each spectrum is then decomposed into a number of small time-frequency components of uniform size and normalized by its average power, as in Fig. 4(a), where G_{t,w_m} denotes the component at time t and the frequency band w_m of the reference signal. Next, the spectrum corresponding to each component is classified by VQ. A VQ codebook is prepared for each frequency band using the LBG algorithm. Then, an index is made for the VQ codes of the reference signal. The index is a list of positions at which each VQ code appears. We perform these processes prior to the search stage.

The four main processing steps are described below.

Step 1 The time-frequency spectra of a query are extracted and decomposed into components [Fig. 4(a)], as done for the reference signals in the pre-processing. Here, let F_{t_i,w_m} be the decomposed component at time t_i of the query, where w_m is the frequency band of F_{t_i,w_m}; let $T_Q = \{t_1, t_2, ...\}$ be the entire set of t_i of decomposed components of the query; and let $W = \{w_1, w_2, ...\}$ be the entire set of frequency bands. These decomposed components are normalized by power and classified by VQ in each frequency band using the same VQ codebook as that used for the stored signals.

Step 2 As shown in Fig. 4(b), components similar to F_{t_i,w_m} are detected. This is easily accomplished by using a look-up table of the similarities between the VQ codes and the index constructed during the pre-processing.


Frequency
Spectrogram of query signal segment
w_m
w_2
w_1
t_1 t_2 t_i
Time
F_{t_i,w_m}
Spectrogram of reference signal
G_{t,w_m}
t
Similar VQ codes
$c \in \mathrm{Sim}(f_{VQ}(F_{t_i,w_m}))$
$\mathrm{Idx}(c)$
Index made in advance for reference signal (for each VQ code, make list of positions of components with that VQ code)
detect components similar to F_{t_i,w_m} just by referring to index
$S(t)$
Detected components

(a) Extract spectrograms and decompose spectrogram into small components.

(b) Detect every similar component for each F_{t_i,w_m}.

(c) Calculate total similarity at t from local similarities of detected components.

Figure 4: Overview of DAL search method

```
$S(T_R) := 0$ /* initialization */
for all $w_m \in W$ do
    for all $t_i \in T_Q$ do
        for all $c \in \mathrm{Sim}(f_{VQ}(F_{t_i,w_m}))$ do
            for all $t_r \in \mathrm{Idx}(c)$ do
                /* voting on the time of reference signal
                   corresponds to the head of query segment */
                $S(t_r - t_i) := S(t_r - t_i) + s(F_{t_i,w_m}, G_{t_r,w_m})$
            end for
        end for
    end for
end for
$S(T_R) := S(T_R)/(|T_Q||W|)$ /* normalization */
```

Figure 5: Calculation of similarity $S(t)$ by voting

Step 3 As in Fig. 4(c), the similarities with respect to each component detected in Step 2 are integrated, and the total similarity $S(t)$ for each segment of the reference signal is calculated as

$$S(t) = \frac{1}{|T_Q||W|} \sum_{w_m \in W} \sum_{t_i \in T_Q} s(F_{t_i,w_m}, G_{t+t_i,w_m}), \tag{7}$$

where $s(F_{t_i,w_m}, G_{t+t_i,w_m})$ is the similarity between F_{t_i,w_m} and G_{t+t_i,w_m}. In the $S(t)$ calculation, $s(F_{t_i,w_m}, G_{t+t_i,w_m})$ is evaluated only if G_{t+t_i,w_m} is detected as a component similar to F_{t_i,w_m} in Step 2.

Step 4 Segments whose total similarities exceed a certain threshold are determined to contain the (original) copy of the query.

Figure 5 shows an algorithm for calculating $S(t)$. Here, $f_{VQ}(\cdot)$ is a function for mapping a spectrum segment to a VQ code, $\mathrm{Sim}(\cdot)$ is a look-up table that returns a set of VQ codes similar to an input VQ code, $\mathrm{Idx}(\cdot)$ is a list of positions at which each VQ code appears, and $T_R = \{t_{R1}, t_{R2}, ...\}$ denotes all the time frames of the reference signal. As an example of a simple implementation, we can adopt the definitions $\mathrm{Sim}(c) \equiv \{c\}$ and $s(F, G) \equiv 1$.

As

$$\sum_{c \in \mathrm{Sim}(f_{VQ}(F_{t_i,w_m}))} |\mathrm{Idx}(c)| \ll |T_R|, \tag{8}$$

the above procedure is much more efficient than an exhaustive search, which requires a calculation cost of order $O(|T_Q||W||T_R|)$.

5 TV2010 submissions and results

This section describes our submitted runs for the TRECVID 2010 CBCD task.

5.1 Audio+Video results

In 2010, audio only and video only results were not tested, but audio + video results were required to be submitted. We merged the audio and video results using the following procedure for each query. We empirically prioritized the audio result when the audio and video results conflicted.

Table 1: Search algorithms and settings

	media	algorithm	configuration and pre-processing operations
a1	audio	DAL	High density, small window width
v1	video	CAM	normal + fixed crop + anti-frame drop + flip
v2	video	CAM	automated crop + normalized intensity + monochrome, internal threshold =0.384 for balanced profile
v3	video	CAM	same as v2 except for the internal threshold=0.533 for no false alarm profile

1. If there are overlapping audio and video results with same reference ID, the audio results are accepted.

2. If there is no overlapping result, the audio results are accepted.

3. If there is no audio result, the video results are accepted.

When multiple results overlap on the same query segment, we accept only the top result regarding the length of the detected segment to avoid false alarms.

5.2 Submitted results

We submitted four runs with varying combinations of search engine settings. The configuration for the audio copy detection was the same among all runs (a1). For the video copy detection, we used three configurations (v1, v2 and v3). These are summarized in Table 1.

The labels and combinations of the settings of the submitted runs are as follows.

NTT-CSL.m.nofa.0 : a1 + v1

NTT-CSL.m.balanced.1 : a1 + v1 + v2

NTT-CSL.m.balanced.2 : a1 + v1 + v3

NTT-CSL.m.nofa.3 : a1 + v1 + v3

NTT-CSL.m.balanced.2 and NTT-CSL.m.nofa.3 are identical except for the application profile.

5.3 Evaluation results

Figure 6 shows the evaluation results of the submitted "balanced" profile runs. The configuration v3 is a strict version of the configuration of v2. The difference between the two configurations was small, we found no significant difference between NTT-CSL.m.balanced.1 and NTT-CSL.m.balanced.2.

Figure 7 shows the evaluation results of the submitted "no false alarm" profile runs. We can see that the NDCR score of NTT-CSL.m.nofa.3 is slightly better than that of NTT-CSL.m.nofa.3. This means that pre-processing before the video feature extraction is working well.

We analyzed the detailed results and found that the queries that contain short segments (e.g. less than 10 seconds) to be detected tend to be missed and result in false negatives for all the runs in this implementation.

6 Conclusions

In this paper, we described our approaches and results in the TRECVID 2010 CBCD task. The transformations in the TRECVID CBCD task include more complex patterns than the transformations frequently found in the real world, and so we introduced a pre-processing stage in the video feature extraction. The evaluation results proved a good accuracy and robustness of our methods against various transformations. Our future tasks will include improving detection accuracy for very short segments.

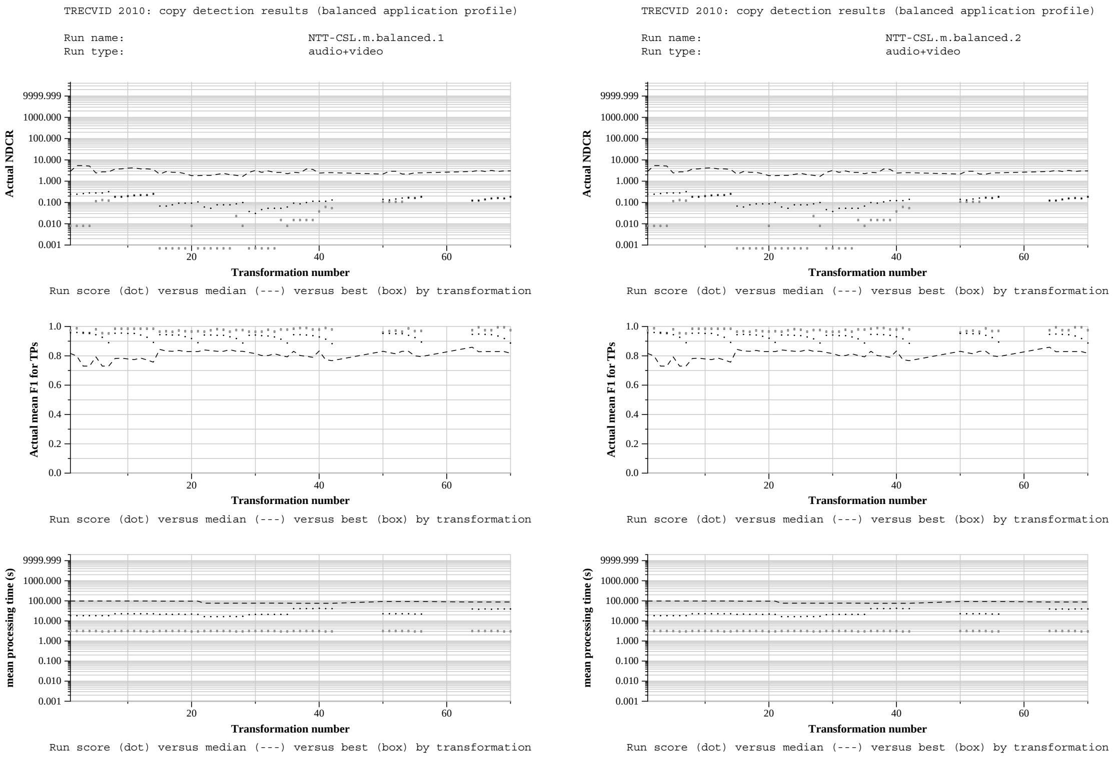


Figure 6: Evaluation results for balanced profile runs

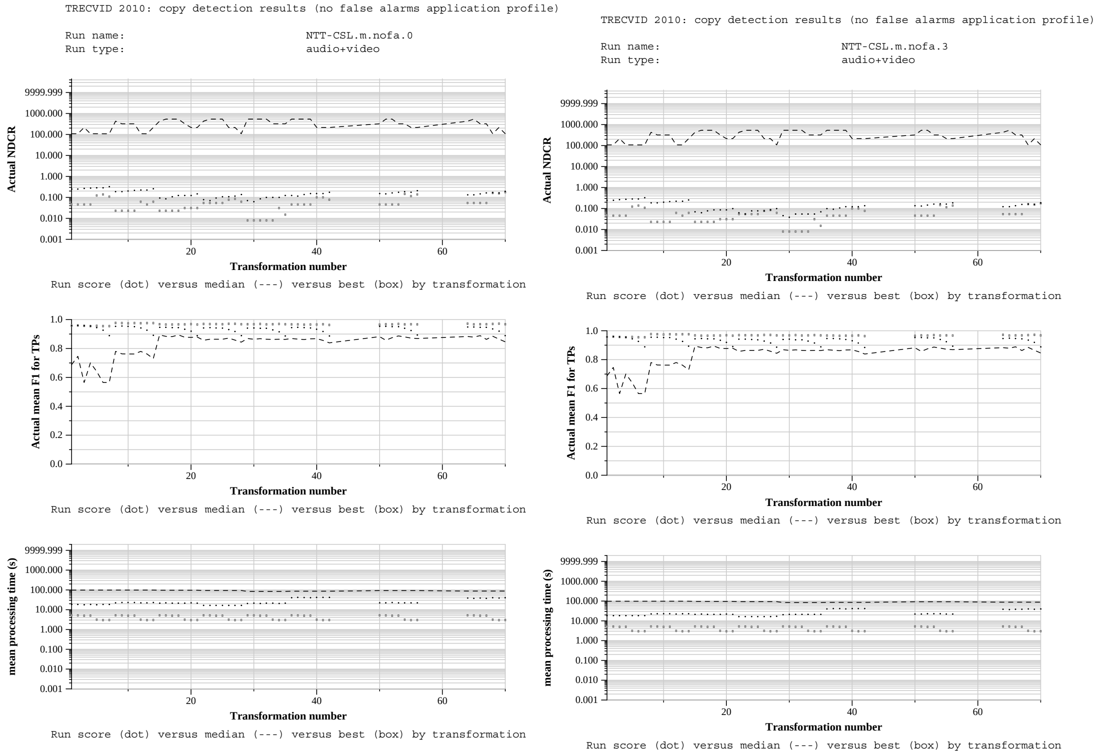


Figure 7: Evaluation results for no false alarm profile runs

References

[1] NTT Data Corporation Press Release, “New content monitoring service identifies audio and video on the Internet quickly and accurately,” 2009, http://www.nttdata.co.jp/en/media/2008/120100.html.

[2] T. Kurozumi, H. Nagano, and K. Kashino, “A robust video search method for video signal queries captured in the real world,” *IEICE Trans. Inf.& Syst.(Japanese Edition)*, vol. J90-D, no. 8, pp. 2223–2231, 2007, in Japanese.

[3] K. Kashino, A. Kimura, H. Nagano, and T. Kurozumi, “Robust search methods for music signals based on simple representation,” in *ICASSP 2007: Proceedings of 2007 International Conference on Acoustics, Speech, and Signal Processing*, 2007, pp. IV–1421–1424.

[4] H. Nagano, K. Kashino, and H. Murase, “A fast search algorithm for background music signals based on the search for numerous small signal components,” in *ICASSP 2003: Proceedings of 2003 International Conference on Acoustics, Speech, and Signal Processing*, 2003, pp. V–796–799.