Video Copy-Detection and Localization with a Scalable Cascading Framework

Yonghong Tian, Tiejun Huang, Menglin Jiang, and Wen Gao
Peking University

A novel video copy-detection and localization approach with scalable cascading of complementary detectors and multiscale sequence matching can improve copy-detection accuracy and localization precision for most audio-visual transformations.

The Internet continues to grow into a global platform for multimedia content distribution. According to YouTube, users uploaded a staggering 72 hours of video to the site every minute in April 2012. The explosive growth of videos on the Internet aggravates the proliferation of near-duplicate copies. Also in April 2012, a German court ordered Google to install filtering software on YouTube in the country to prevent users from uploading copyrighted material. These developments highlight the necessity of technical solutions for video copy detection. Similarly, by detecting near-duplicate copies, video search engines can return results with semantically coherent but visually diversified content.

Technologically, however, copy detection and localization in a Web-scale video database is a challenging task. This is mainly because Web video copies have often undergone various complex transformations on the audio and video components in a video file or stream. In addition, researchers have discovered that no single audio-visual feature, or single detector based on several features, can help detect all transformations. To address this problem, we propose a novel video copy-detection and localization approach with scalable cascading of complementary detectors and multiscale sequence matching. In this cascade framework, a soft threshold learning algorithm is utilized to estimate the optimal decision thresholds for detectors, and a multiscale sequence-matching method is employed to precisely locate copies using a 2D Hough transform and multi-granularities similarity evaluation.

Extensive experiments were performed on the TRECVID Content-Based Copy Detection (TRECVID-CBCD) 2011 benchmark dataset. The results show that our approach achieves the best copy-detection accuracies and localization precisions for most transformations on this dataset.

Audio and Video Transformations

Figure 1 and Table 1 show examples of transformations used for the TRECVID-CBCD tasks organized by the National Institute of Standards and Technology (NIST). We can see that compared with audio transformations that contain relatively few variations (such as compression, companding, and mixing with speech), visual transformations are much more diverse and complicated. Usually, video content is largely preserved after spatial or temporal content-preserving operations such as format conversion, quality reduction (such as noise addition, resolution change, and re-encoding), and frame-rate change. In contrast, video is notably modified after spatial or temporal content-altering operations such as cropping, picture-in-picture (PiP), pattern insertion, temporally replacing, or reordering.

After years of practice, researchers have widely recognized that no single feature, or single detector based on several features, can be both robust and discriminative for copy-detection tasks under various transformations. That is, if we utilize a set of audio-visual features to construct several detectors, some of which may be robust against certain types of transformations but vulnerable to other types, other detectors may be the other way around. Thus, it is beneficial to combine several detectors to enhance the robustness and discriminability of a copy-detection system. (See the ''Related Work in Video Copy Detection''

1070-986X/13/$31.00 © 2013 IEEE

Published by the IEEE Computer Society

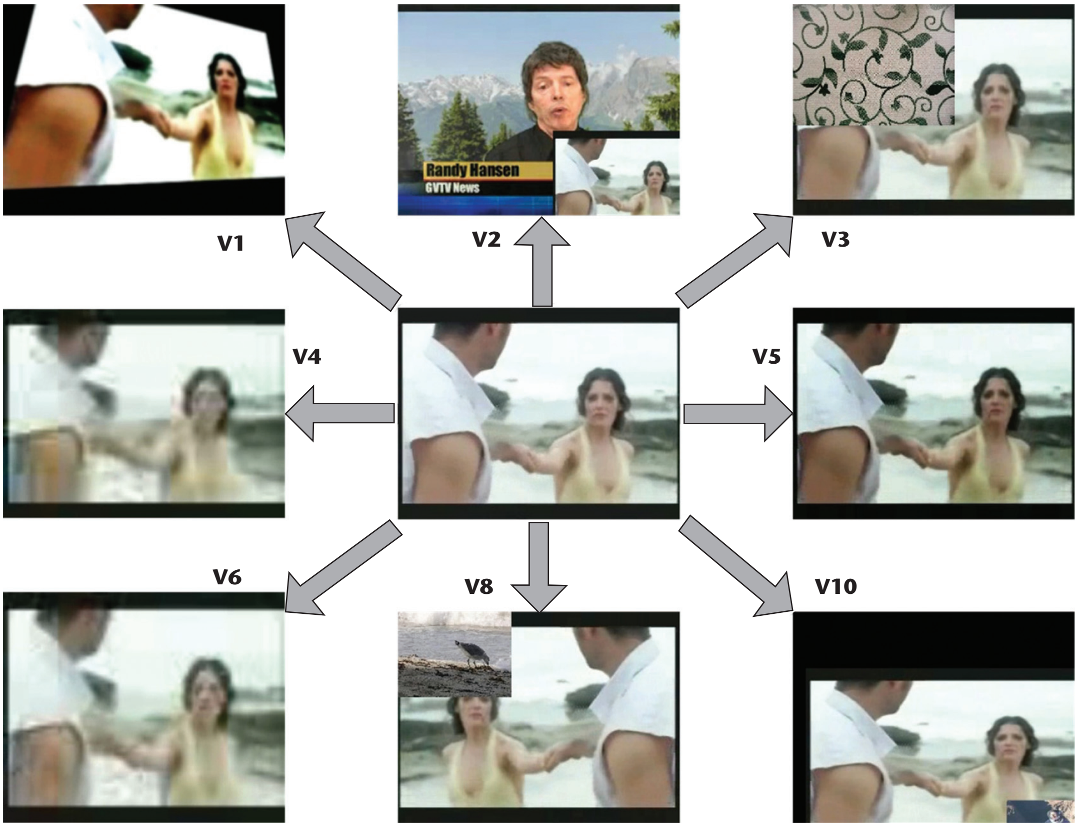


Figure 1. Examples of visual transformations on a Web video. See Table 1 for a description of the various labels.

Table 1. Transformations used for the TRECVID-CBCD tasks.

Category	Label	Type
Visual transformation	V1	Camcorder
	V2	Picture in picture
	V3	Pattern insertion
	V4	Re-encoding
	V5	Gamma change
	V6	Quality decrease
	V8	Postproduction
	V10	Combination of three randomly chosen transformations
Audio transformation	A1	Do nothing
	A2	MP3 compression
	A3	MP3 compression and multiband companding
	A4	Bandwidth limit and single-band companding
	A5	Mix with speech
	A6	Mix with speech and multiband compression
	A7	Bandwidth filter, mix with speech, and compression
Mixed transformation*	Mn	$\mathrm{V}x + \mathrm{A}y \Rightarrow \mathrm{M}[(x - 1) * 7 + y]$

* There were a total of 56 mixed transformations, each of which were generated by applying a combination of eight visual transformations and seven audio transformations.

Related Work in Video Copy Detection

Video copy detection mainly involves two key techniques: feature representation and video matching. According to their intrinsic characteristics, the features used in existing work can be classified into two categories: global and local features. Based on the statistics of the entire frame or the whole clip, global features such as the spatial or temporal ordinal signatures[1,2] or image/video fingerprints[3,4] have advantages of compactness and low computational complexity and, most importantly, are largely invariant to content-preserving transformations. However, they cannot deal well with complex transformations such as postproduction. Instead, local features are by nature resistant to such content-altering operations because a part of original content always remains in the copy.

Toward this end, local features are mostly based on the interest-point detection and local-descriptor calculation.[5–7] Among them, the scale-invariant feature transform (SIFT) could be the most suitable local visual feature for copy detection. To accelerate feature matching, the bag-of-words (BoW) technique is used frequently by building a visual vocabulary for local features and constructing a visual word histogram to represent each frame.[5] To make use of both visual and audio features in video copy detection, some audio features originally designed for retrieval are also used for copy detection, such as Mel-frequency cepstral coefficients (MFCCs), mean energy, normalized spectral sub-band moments (NSSMs), and audio spectrum flatness (ASF).[8]

More generally, it is beneficial to combine several different features, or several detectors based on some features, to improve the detection performance. For example, a multiple-feature hashing algorithm was proposed in earlier work[9] to fuse multiple features for copy detection, by learning a group of hash functions for video keyframes and generating a series of binary codes to represent each video. The combination is formulated as a reranking problem,[8] which recalculates the similarity scores for all the individual detection results and then employs four strategies (such as average, max, multiply, and logistic) to choose the best match. In another approach,[10] detection results using audio and visual features are fused by selecting video matches with the highest similarity score. Obviously, these approaches could achieve better detection accuracy than a single feature/detector, yet at a cost of much longer overall processing time. Therefore, this work proposes a scalable cascading framework to organize complementary detectors in a cascade structure, consequently reducing the processing time for most copies.

Video matching generally refers to the process of similarity evaluation and copy assertion between two video clips based on extracted features. Existing matching methods can be roughly classified into sliding-window-based and frame-fusion-based categories. Assuming that two videos are matched directly by frame-to-frame matching, a sliding window with the same size as the query clip is moved frame by frame along the reference video, in which the similarity is calculated.[1,2] The major drawbacks were high computational complexity and vulnerability to temporal transformations such as frame dropping. Thus, a more flexible and widely used way is to search the reference database, obtain a list of similar reference frames for each query frame, and then determine whether it is a copy by fusing these reference frames.[11,12] However, without a proper temporal fusing mechanism, copies are difficult to be precisely located. Therefore, a spatiotemporal post-filtering method was presented to keep only the frame matches that are consistent with a spatiotemporal model.[5] Another approach applied a 2D Hough

sidebar for more details.) This trend has also been validated by recent practices in the TRECVID-CBCD contest,[1,2] in which most of the participating approaches compute several detection results through individual features and then fuse them to obtain the final result. Although such approaches could achieve good detection accuracy, they have an obvious drawback. That is, the processing time will be at least the sum of the time required by all detectors, if not optimized with multithread or multiprocess programming.

To address this problem, this article presents a novel scalable cascading framework to organize multiple complementary detectors in a cascade framework. In this framework, a query video is sequentially processed until one detector asserts it as a copy. So the number of used detectors is largely dependent on the transformations that the query video is subject to. In this way, the processing time can be significantly reduced for most copies because they can be correctly detected through at most the first several detectors. Moreover, instead of manually tuning decision thresholds for detectors, a soft threshold learning algorithm is also proposed to estimate the optimal decision thresholds. In this algorithm, none of dataset-related priors is specified, and thus it exhibits a good generalization on various databases.

In the frame-based detection paradigm,[3] without a proper temporal fusing mechanism transform on frame matches to locate the copy segment.[10] Following this, we propose a multiscale sequence matching method using a 2D Hough transform and multigranularities similarity evaluation to achieve a better trade-off between high localization preciseness and fast matching speed.

From 2008 to 2011, TRECVID organized a content-based copy detection (CBCD) task. Given a test collection of videos and a set of queries, the task is to determine for each query whether it is a copy and if so, where some part of the query occurs in the test collection. All submitted results are evaluated for two profiles: NOFA, which aims to reduce the false alarm rate to 0, and BALANCED, which sets an equal cost for false alarms and misses. The benchmark dataset contains 394 hours of high-quality professional videos from 2008 and 2009 and 425 hours of 11,524 poor-quality Web videos from 2010 and 2011. Each year, more than 20 teams participated in this contest.[13] It is widely recognized that the TRECVID-CBCD 2010/2011 dataset is one of the largest and most complex benchmarks for Web-scale video copy detection.

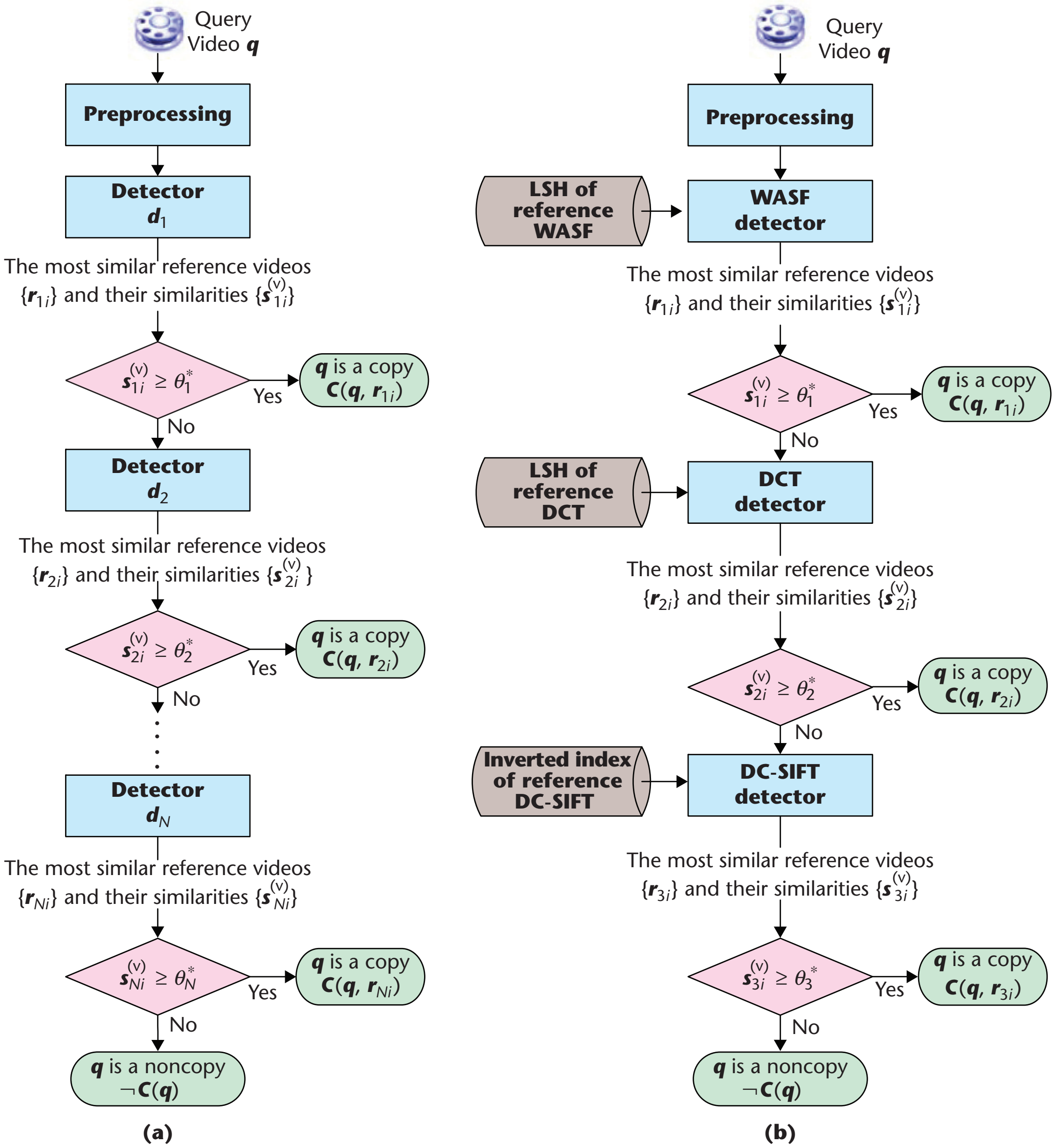
References

1. C.Y. Chiu, C.S. Chen, and L.F. Chien, ''A Framework for Handling Spatio Temporal Variations in Video Copy Detection,'' *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 3, 2008, pp. 412–417.
2. C. Kim and B. Vasudev, ''Spatiotemporal Sequence Matching for Efficient Video Copy Detection,'' *IEEE Trans. Circuits and Systems for Video Technology,* vol. 15, no. 1, 2005, pp. 127–132.
3. J. Oostveen, T. Kalker, and J. Haitsma, ''Feature Extraction and a Database Strategy for Video Fingerprinting,'' *Proc. Int'l Conf. Recent Advances in Visual Information Systems,* Springer-Verlag, 2002, pp. 117–128.
4. C. De Roover et al., ''Robust Video Hashing Based on Radial Projections of Key Frames,'' *IEEE Trans. Signal Processing,* vol. 53, no. 10, 2005, pp. 4020–4037.
5. M. Douze, H. Jegou, and C. Schmid, ''An Image-Based Approach to Video Copy Detection with Spatio-Temporal Post-filtering,'' *IEEE Trans. Multimedia,* vol. 12, no. 4, 2010, pp. 257–266.
6. A. Joly, O. Buisson, and C. Frélicot, ''Content-Based Copy Retrieval Using Distortion-Based Probabilistic Similarity Search,'' *IEEE Trans. Multimedia,* vol. 9, no. 2, 2007, pp. 293–306.
7. X. Wu et al., ''Real-Time Near-Duplicate Elimination for Web Video Search with Content and Context,'' *IEEE Trans. Multimedia,* vol. 11, no. 2, 2009, pp. 196–207.
8. T.J. Huang et al., ''Mediaprinting: Identifying Multimedia Content for Digital Rights Management,'' *Computer,* vol. 43, no. 12, 2010, pp. 28–35.
9. J.K. Song et al., ''Multiple Feature Hashing for Real-Time Large Scale Near-Duplicate Video Retrieval,'' *Proc. ACM Multimedia,* ACM, 2011, pp. 423–432.
10. Y. Liu et al., ''Coherent Bag-Of Audio Words Model for Efficient Large-Scale Video Copy Detection,'' *Proc. ACM Multimedia,* ACM, 2010, pp. 89–96.
11. N. Gengembre and S.A. Berrani, ''A Probabilistic Framework for Fusing Frame-Based Searches within a Video Copy Detection System,'' *Proc. ACM CIVR,* ACM, 2008, pp. 211–220.
12. S.K. Wei et al., ''Frame Fusion for Video Copy Detection,'' *IEEE Trans. Circuits and Systems for Video Technology,* vol. 21, no. 1, 2011, pp. 15–28.
13. W. Kraaij and G. Awad, *TRECVID 2011 Content-Based Copy Detection: Task Overview,* Nov. 2011; www-nlpir.nist.gov/projects/tvpubs/tv11.slides/tv11.ccd.slides.pdf.

on frame-level detection results, copies are difficult to locate precisely. To solve this problem, a multiscale sequence matching method is presented to assemble frame-level similarity search results into video-level matches using a 2D Hough transform and multigranularities similarity evaluation. As such, video matching is performed in a pyramid structure, making it computationally efficient.

Detection with a Scalable Cascading Framework

Given a query video q and reference videos $\mathbb{R} = \{r_i\}$ $(1 \leq i \leq R)$, the task of copy detection is to examine whether $\exists r_i \in \mathbb{R}$ such that $C(q, r_i)$ holds, where $C(x, y)$ means x is a copy of y. Toward this end, this article proposes a scalable cascading framework to organize detectors that complement each other (see Figure 2a). Inspired by the well-known classifier cascade,[4] our cascading framework places a series of detectors in simple-to-complex order. Namely, efficient but relatively simple detectors should be placed in the front, while effective but complex detectors should be located in the rear. In an N-stage cascade of detectors, $\mathbb{D}_N = \langle d_1, d_2, \ldots, d_N \rangle$, a query q is processed by each detector successively until one determines it as a copy or all determine it as a noncopy. That is, q is first processed by d_1, where a set of positive results $\{r_{1i}\}$ are returned. Then for each r_{1i}, if its similarity $s_{1i}^{(\mathrm{V})}$ is greater than or equal to a threshold θ_1, q will be immediately accepted as a copy



Query Video q
Preprocessing
Detector d1
The most similar reference videos {r1i} and their similarities {s(v)1i}
s(v)1i ≥ θ*1
Yes
q is a copy C(q, r1i)
No
Detector d2
The most similar reference videos {r2i} and their similarities {s(v)2i}
s(v)2i ≥ θ*2
Yes
q is a copy C(q, r2i)
No
Detector dN
The most similar reference videos {rNi} and their similarities {s(v)Ni}
s(v)Ni ≥ θ*N
Yes
q is a copy C(q, rNi)
No
q is a noncopy ¬C(q)
(a)
Query Video q
Preprocessing
LSH of reference WASF
WASF detector
The most similar reference videos {r1i} and their similarities {s(v)1i}
s(v)1i ≥ θ*1
Yes
q is a copy C(q, r1i)
No
LSH of reference DCT
DCT detector
The most similar reference videos {r2i} and their similarities {s(v)2i}
s(v)2i ≥ θ*2
Yes
q is a copy C(q, r2i)
No
Inverted index of reference DC-SIFT
DC-SIFT detector
The most similar reference videos {r3i} and their similarities {s(v)3i}
s(v)3i ≥ θ*3
Yes
q is a copy C(q, r3i)
No
q is a noncopy ¬C(q)
(b)

Figure 2. Schematic illustration of (a) the proposed cascading framework and (b) one of its implementations.

of r_{1i}. Otherwise, the evaluation of d_2 on q will be triggered and so forth. Only if q is asserted as a noncopy by all the detectors will it be accepted as a noncopy. In practice, most copies can be detected through the first several detectors, thus saving a great deal of processing time.

Two key issues should be addressed in our framework: how to set complementary detectors d_1, d_2, …, d_N and how to determine the decision threshold θ_n for d_n. Note that if $\Theta = \{\theta_1, \theta_2, \ldots, \theta_N\}$ are set manually, they are called hard thresholds; otherwise, soft thresholds.

Complementary Detectors

Figure 2b shows one implementation of the proposed cascading framework. The basic motivation is to exploit several audio-visual features that complement each other on different transformations. To meet the complementarity requirement, three features are used in our implementation, including an audio feature named weighted audio spectrum flatness (WASF),[5] a global visual feature based on a discrete cosine transform (DCT), and a local visual feature of dense-color scale-invariant feature transform (DC-SIFT).[6] The complementarity of visual and audio features is obvious. DC-SIFT and DCT are complementary in that the former can effectively handle spatial content-altering transformations and the latter can resist spatial content-preserving but quality-degrading operations. This can be further illustrated by Table 2, which shows the effectiveness of different features on the audio-visual transformations. We can see that using only WASF is enough to deal with audio transformations A1–A4, no matter which visual transformations exist. However, if the query video is also subject to one of the V3–V6 visual transformations, WASF and DCT are needed. For the remaining cases, three features should be used. Our experimental results also validate the complementarity of individual features in our system.

WASF Detector. As an extension of a MPEG-7 descriptor, we use WASF to cope with audio transformations such as MP3 compression and multiband companding. A descriptor with 72D WASF features is extracted from each 6-second audio frame, and we adopt Euclidean distance to measure the dissimilarity between two descriptors. For reference videos, all the WASF descriptors are indexed by locality sensitive hashing (LSH) for efficient feature matching, where the indexing tables are generated using 16 spherical hashing functions.[7]

DCT Detector. DCT features are robust in content-preserving transformations. Moreover, they are compact and computationally efficient. In our system, a new DCT feature is designed by utilizing the relationship between low-frequency DCT coefficients of adjacent image blocks. It differs from the original DCT feature in that subband energy is used as an alternative to DCT coefficient. Then the invariance of its relative magnitude can be used to improve the feature's robustness.

In this detector, Hamming distance is used as the distance metric, and LSH is used to index all the reference DCT features. Here the binary hashing functions are used, by randomly selecting one dimension in the DCT feature each time to generate a hashing indexing table.

DC-SIFT Detector. In our system, DC-SIFT is adopted to cope with content-altering visual transformations (such as camcording, PiP, and postproduction). We use it to replace the SIFT and speeded up robust feature (SURF) in our TRECVID-CBCD 2010 system, which can obtain high detection accuracy at the cost of a long processing time. Furthermore, the bag-of-words (BoW) technique is applied to convert each DC-SIFT descriptor into a visual word (800 words generated from 10 million DC-SIFT descriptors).

Table 2. The effectiveness of different features on the audio-visual transformations.

	A1	A2	A3	A4	A5	A6	A7
V1	Case 1: WASF only				Case 3: WASF + DCT + DC-SIFT		
V2							
V3					Case 2: WASF + DCT		
V4							
V5							
V6							
V8					Case 3: WASF + DCT + DC-SIFT		
V10							

However, BoW representation might lead to a loss of discriminability for the descriptors. Therefore, the position, orientation, and scale information for each keypoint is taken into account so that only keypoints mapped to the same visual word and roughly with the same position, orientation, and scale will be regarded as matches. In particular, a keyframe's spatial region is divided into 2 × 2 cells, so the position of each keypoint is quantized into one integer (0–3). Similarly, the orientation and scale of each keypoint are quantized into 16 and 2 bins, respectively—thus, actually having $800 \times 4 \times 16 \times 2 = 100{,}000$ words in the extended vocabulary. For all reference videos, DC-SIFT BoWs are stored into an inverted index.

Preprocessing

Two preprocessing issues should also be addressed here. The first is to extract audio and video frames. In our system, visual keyframes are obtained by uniformly sampling the video component at a rate of 3 frames per second (fps), while 6-second audio frames are constructed from every 198 audio words with a 5.4-second overlap between adjacent frames, where each audio word is a short segment of 90 ms with a 60-ms overlap between consecutive words.

The second issue is PiP detection. PiP frames are detected using a Hough transform that detects two pairs of parallel lines to locate the inserted foreground videos. For queries with PiP, the foreground and non-foreground keyframes will be processed respectively to check whether the corresponding videos are a copy. In addition, queries asserted as noncopies will be flipped and matched again to identify flip transformations.

Detection Using Frame Fusion

To maintain robustness to various temporal transformations, all detectors in our system follow the frame-based copy detection paradigm,[3] where the final result is obtained by assembling frame-level similarity search results into video-level matches. Given a query video q, each detector $d_n \in \mathbb{D}_N$ picks up the top K_1 similar reference keyframes (audio frames) for each query keyframe (audio frame) ($K_1 = 20$ in this work), obtaining a collection $\mathbb{M}_n^{(\mathrm{F})}(q)$ of frame-level matches. Each match is expressed as

$$\mathbf{m}_n^{(\mathrm{F})} = (t(q), t(r)) = \left\langle q, t(q), r, t(r), s_n^{(\mathrm{F})} \right\rangle \tag{1}$$

which means the reference frame of r at timestamp $t(r)$ is a match to that of q at timestamp $t(q)$, with a similarity $s_n^{(\mathrm{F})}$. Then through an appropriate sequence-matching method described in the next section, a set of video-level matches can be obtained, each denoted by

$$\mathbf{m}_n^{(\mathrm{V})}(q,r) = \left\langle q, t^{(\mathrm{B})}(q), t^{(\mathrm{E})}(q), r, t^{(\mathrm{B})}(r), t^{(\mathrm{E})}(r), s_n^{(\mathrm{V})} \right\rangle \tag{2}$$

where $[t^{(\mathrm{B})}(q), t^{(\mathrm{E})}(q)]$ and $[t^{(\mathrm{B})}(r), t^{(\mathrm{E})}(r)]$ are timestamps for the beginning and ending of the copy segment in Q and r, and $s_n^{(\mathrm{V})}$ denotes the video-level similarity calculated by d_n.

Soft Threshold Learning

For each detector, it is obvious that artificial adjustment of a hard threshold is burdensome and, more importantly, lacks generalization. Therefore, we design a learning algorithm to automatically determine the optimal soft thresholds $\Theta^* = \{\theta_1^*, \theta_2^*, \ldots, \theta_N^*\}$.

Problem Formulation

Given a query video q, the detector $d_n \in \mathbb{D}_N$ with a threshold θ_n is assumed to output several video-level matches $\{\mathbf{m}_n^{(\mathrm{V})}(q,r)\}$. Then the cost of q, with respect to θ_n, denoted by $c(q, \theta_n)$, is calculated as follows:

1. If q is actually a copy and asserted as a copy by $d_n\left(s_n^{(\mathrm{V})} \geq \theta_n\right)$, then if $\mathbf{m}_n^{(\mathrm{V})}(q,r)$ is the correct reference—that is, the reference video r is right and the copy duration $[t^{(\mathrm{B})}(r), t^{(\mathrm{E})}(r)]$ largely overlaps with the ground truth—then it is a true positive (TP) and $c(q, \theta_n)$ is set to zero. However, if $\mathbf{m}_n^{(\mathrm{V})}(q,r)$ is identified as a wrong reference clip, then it generates a false positive (FP) and a false negative (FN) simultaneously, and $c(q, \theta_n)$ is set to the sum of c_{FP} and c_{FN}, which represent the penalty for a FP and a FN, respectively.

2. If q is a copy but asserted as a non-copy by $d_n\left(s_n^{(\mathrm{V})} \geq \theta_n\right)$, then $\mathbf{m}_n^{(\mathrm{V})}(q,r)$ is a FN and $c(q, \theta_n)$ is set to c_{FN}.

3. If q is a noncopy but asserted as a copy, then $\mathbf{m}_n^{(\mathrm{V})}(q,r)$ is a FP and $c(q, \theta_n)$ is set to c_{FP}.

4. If q is a noncopy and asserted as a noncopy, then $\mathbf{m}_n^{(\mathrm{V})}(q,r)$ is a true negative (TN) and $c(q, \theta_n)$ is set to zero.

By summarizing the four cases, we can get

$$c(q,\theta_n) = \begin{cases} 0 & \text{if } \left(\bar{C}(q) \wedge s_n^{(\mathrm{V})} \geq \theta_n \wedge T\left(\mathbf{m}_n^{(\mathrm{V})}(q,r)\right)\right) \\ & \quad \vee \left(\neg\bar{C}(q) \wedge s_n^{(\mathrm{V})} \geq \theta_n\right) \\ c_{\mathrm{FP}} + c_{\mathrm{FN}}, & \text{if } \left(\bar{C}(q) \wedge s_n^{(\mathrm{V})} \geq \theta_n \wedge \neg T\left(\mathbf{m}_n^{(\mathrm{V})}(q,r)\right)\right) \\ c_{\mathrm{FN}}, & \text{if } \left(\bar{C}(q) \wedge s_n^{(\mathrm{V})} < \theta_n\right) \\ c_{\mathrm{FP}}, & \text{if } \left(\neg\bar{C}(q) \wedge s_n^{(\mathrm{V})} \geq \theta_n\right) \end{cases} \tag{3}$$

where $\bar{C}(q)$ denotes the statement that q is indeed a copy in the ground truth (in contrast, $C(q)$ denotes q is asserted as a copy by the system), $T\left(\mathbf{m}_n^{(\mathrm{V})}(q,r)\right)$ denotes that $\mathbf{m}_n^{(\mathrm{V})}(q,r)$ recognizes the correct reference, and the operator $\neg$ denotes the logical negation. Often, $<c_{\mathrm{FP}}, c_{\mathrm{FN}}>$ can be set empirically. In our study, it is set to $<2, 0.2>$ because a FP is much worse than a FN in many applications, such as copyright protection.

Thus given a training set $\mathbb{Q} = \{q_1, q_2, \ldots, q_n\}$, the error rate of d_n with respect to θ_n can be defined as the weighted sum of the costs for all queries:

$$\varepsilon(\mathbb{Q}, \theta_n) = \sum_{j=1}^{J} w_{n,j} \times c(q_j, \theta_n) \tag{4}$$

where $\mathbf{W}_n = \{w_{n,1}, w_{n,2}, \ldots, w_{n,J}]$ is the weight vector of all training queries for d_n. In the cascade, different detectors may correspond to different training weight vectors. Thus, the objective of threshold learning is to minimize the overall error rate $\varepsilon(\mathbb{Q}, \theta_n)$ θ for all detectors in $\mathbb{D}_N$ over the training set $\mathbb{Q}$.

Learning Algorithm

Ideally, the optimal thresholds $\Theta^* = \{\theta_1^*, \theta_2^*, \ldots, \theta_N^*\}$ should make a good trade-off between FPs and FNs, consequently leading to the minimum error rate. For $d_n \in \mathbb{D}_N$, $\varepsilon(\mathbb{Q}, \theta_n)$ should be calculated at a range of candidate thresholds, sweeping from the minimal video similarity (denoted by $\breve{s}_n^{(V)}$) to the maximal similarity (denoted by $\widehat{s}_n^{(V)}$) so that the similarity score associated with the minimal error rate ε_n^* is chosen as the optimal threshold θ_n^*. That is,

$$\theta_n^* = \mathop{\arg\min}_{\theta_n \in \left[\breve{s}_n^{(V)}, \widehat{s}_n^{(V)}\right]} \varepsilon(\mathbb{Q}, \theta_n) \tag{5}$$

To ensure posterior detectors only focus on the queries that are incorrectly detected by their antecessors, in the training, weights of those correctly judged queries should be abated for these posterior detectors. Motivated by the Adaboost algorithm in object detection,[4] this weight update can be expressed as

$$w_{n+1,j} = \begin{cases} w_{n,j} \frac{\varepsilon_n^*}{1-\varepsilon_n^*}, & \text{if } c(q_j, \theta_n^*) = 0 \\ w_{n,j}, & \text{otherwise} \end{cases} \tag{6}$$

where $w_{n,j}$ and $w_{n+1,j}$ denote the training weights of q_j for two consecutive detectors d_n and d_{n+1}. That is, if q_j is detected correctly by d_n (that is, $c(q_j, \theta_n^*) = 0$), its training weight should become smaller for d_{n+1}. Otherwise, the training weight remains unchanged.

The learning procedure is summarized in Algorithm 1 (see Figure 3).

```
Algorithm 1
Input: A cascade of meta-detectors $\mathbb{D}_N = \langle d_1, d_2 \ldots, d_N \rangle$, a training set $\mathbb{Q} = \{q_j\}_J$, and a reference database $\mathbb{R} = \{r_i\}_R$.
Output: Optimal thresholds $\Theta^* = \{\theta_1^*, \theta_2^*, \ldots, \theta_N^*\}$
Procedure:
1. Initialize weights $w_{1,J} = 1/J$ for $j = 1, 2, \ldots, J$.
2. For $n = 1, 2, \ldots, N$
    2.1. Normalize the weights:
        $w_{n,j} \leftarrow \frac{w_{n,j}}{\sum_{k-1}^{J} w_{n,k}}$ for $j = 1, 2, \ldots, J$
        so that $\mathbf{W}_N$ is a probability distribution.
    2.2. Evaluate $d_n$ on $\mathbb{Q}$, and record the detection results:
        $\mathbb{M}_n^{(V)} = \left\{\mathbf{m}_n^{(V)}(q_j, r_i) | r_i \in \mathbb{R}, j = 1, 2, \ldots, J\right\}$
        Then calculate the maximal and minimal video similarities:
        $\widehat{s}_n^{(V)} = \mathrm{Max}\left(\left\{s_n^{(V)} \mid \mathbf{m}_n^{(V)}(q_j, r_i) \in \mathbb{M}_n^{(V)}\right\}\right)$
        $\breve{s}_n^{(V)} = \mathrm{Min}\left(\left\{s_n^{(V)} \mid \mathbf{m}_n^{(V)}(q_j, r_i) \in \mathbb{M}_n^{(V)}\right\}\right)$
    2.3. Find the optimal threshold for $d_n$,
        $\theta_n^* = \mathop{\arg\min}_{\theta_n \in \left[\breve{s}_n^{(V)}, \widehat{s}_n^{(V)}\right]} \varepsilon(\mathbb{Q}, \theta_n)$
        and record the minimum error rate: $\varepsilon_n^* = \varepsilon(\mathbb{Q}, \theta_n^*)$.
    2.4. Update the weights for $d_{n+1}$:
3. Return $\Theta^* = \{\theta_1^*, \theta_2^*, \ldots, \theta_N^*\}$.
```

Figure 3. Procedure for learning soft thresholds.

Localization Using Multiscale Sequence Matching

If a query q is asserted as a copy of $r_i \in \mathbb{R}$, then the task of copy localization is to determine the precise beginning and ending timestamps of the copy segments in q and r_i—namely, $\left[t^{(B)}(q), t^{(E)}(q)\right]$ and $\left[t^{(B)}(r_i), t^{(E)}(r_i)\right]$. Instead of treating copy localization as an additional computation of copy detection,[8] our system performs in a localization-by-detection manner. That is, given frame-level matches $\mathbb{M}_n^{(F)}(q) = \left\{\mathbf{m}_n^{(F)}(t(q)), t(r_i)\right\}$ obtained by detector d_n, our system first identifies a set of video-level match hypotheses $\mathbb{H}_n(q)$, each with the corresponding precise locations. It then evaluates the similarity $s_n^{(V)}$ between the copy segments q and r_i for each hypothesis and, finally, picks up the video matches with $s_n^{(V)} \geq \theta_n^*$. In this process, copy localization is performed simultaneously with detection.

One crucial issue is how to achieve high localization preciseness and fast matching speed simultaneously. Toward this end, this article proposes a multiscale sequence matching method using a 2D Hough transform and multigranularities similarity evaluation.

Multiscale Sequence Matching

Inspired by spatial pyramid matching,[9] which conducts a pyramid match kernel in 2D image space, multiscale sequence matching partitions each video into increasingly finer temporal segments and combines similarities over multiple granularities in a principled way. This "multiscale" way offers a certain degree of freedom in video matching and thus remains robust to the possible temporal transformations.

Figure 4 shows two typical paradigms of multiscale sequence matching. In the first paradigm, dynamic time warping (DTW) methods

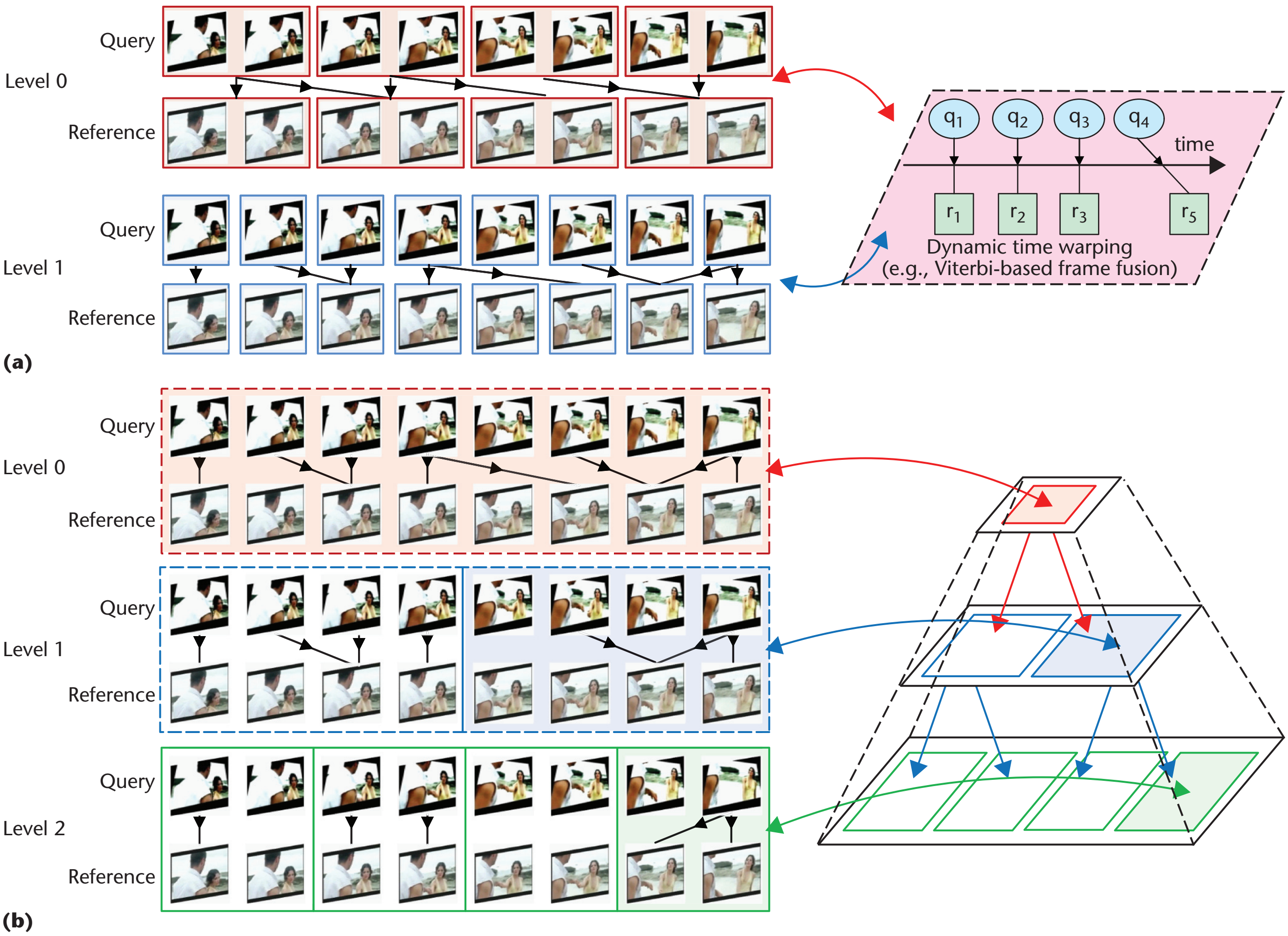


Figure 4. Two typical paradigms of multiscale sequence matching: (a) dynamic time warping (DTW) and (b) temporal pyramid matching (TPM).

(such as the Viterbi-based frame fusion algorithm[3]) are performed on multiple temporal granularities, with different numbers of frames in a node. In each granularity, DTW can find the optimal path that minimizes the accumulated distance between two sequences, independent of certain nonlinear temporal variations. After applying a 2D Hough transform to identify a set of video-level match hypotheses, the second paradigm, temporal pyramid matching (TPM),[2] places a sequence of increasingly coarser grids on each hypothesis and takes a weighted sum of the number of matches that occur at each level. Here two nodes are said to match if they fall into the same cell of the grid. Matches found at finer levels are weighted more highly than matches found at coarser levels. Compared with DTW, TPM is computationally more efficient because it calculates the similarity between two nodes at each level simply by counting the matches.

TPM Algorithm

For frame-fusion based matching, a key issue is to define appropriate temporal constraints on frame-level matches such that two matched sequences have coherent timestamps (that is, $t^{(E)}(r_i) - t^{(B)}(r_i) \approx t^{(E)}(q) - t^{(B)}(q)$). However, it is not easy to define such constraints. Constraints that are too strict could effectively filter out some false alarms (for example, a noncopy reference video with only several visually similar frames), meanwhile inevitably causing some misses because it is hard to find one-one frame correspondence between an original and its transformed copies. However, constraints that are too relaxed may cause the opposite problem.

TPM addresses this contradiction with a 2D Hough transform and multigranularities similarity evaluation. Its procedure can be described as follows:

Given frame-level matches $\mathbb{M}_n^{(\mathrm{F})}(q) = \{\mathbf{m}_n^{(\mathrm{F})}(t(q), t(r_i))\}$ where $\mathbf{m}_n^{(\mathrm{F})}(t(q), t(r_i)) = \langle q, t(q), r_i, t(r_i), s_n^{(\mathrm{F})}\rangle$, a 2D Hough transform is first conducted on $\mathbb{M}_n^{(\mathrm{F})}(q)$ to vote in K_2 video-level hypotheses $\mathbb{H}_n(q)$ ($K_2 = 10$ in this work), where all frame-level matches in each hypothesis have a similar time shift value δt. For $\mathbf{m}_n^{(\mathrm{F})}(t(q), t(r_i))$, for example, the time shift between two frames is calculated by $\delta t = t(q) - t(r_i)$. Then for $\mathbf{h}_n(q) \in \mathbb{H}_n(q)$, the timestamps of a copy in q and r_i—that is, $\left[t^{(\mathrm{B})}(q),\ t^{(\mathrm{E})}(q)\right]$ and $\left[t^{(\mathrm{B})}(r_i),\ t^{(\mathrm{E})}(r_i)\right]$—are identified by picking up the first and last frame-level matches that accord with this hypothesis.

The next step is to evaluate the similarity between $\left[t^{(\mathrm{B})}(q),\ t^{(\mathrm{E})}(q)\right]$ and $\left[t^{(\mathrm{B})}(r_i),\ t^{(\mathrm{E})}(r_i)\right]$ for each $\mathbf{h}_n(q) \in \mathbb{H}_n(q)$. Toward this end, they are partitioned into increasingly finer segments, with a total of $G = 2^l$ segments at level l ($l = 0, 1, \ldots, L$), denoted by $(ts_0(q), \ldots, ts_{G-1}(q))$ and $(ts_0(r_i), \ldots, ts_{G-1}(r_i))$, respectively. Then similarity scores of frame-level matches across the two segments are accumulated to get the segment similarity, namely,

$$s_{n,l,g}^{(\mathrm{S})} = \sum_{\substack{\mathbf{m}_n^{(\mathrm{F})}(t(q),t(r_i)) \in \mathbf{h}_n(q),\\ t(q) \in ts_g(q),\, t(r_i) \in ts_g(r_i)}} s_n^{(\mathrm{F})}(t(q), t(r_i)) \tag{7}$$

Thus, the video-level similarity at level l can be calculated as follows:

$$s_{n,l}^{(\mathrm{V})} = \frac{1}{m} \sum_{g=0}^{G-1} s_{n,l,g'}^{(\mathrm{S})} \tag{8}$$

where m is the number of keyframes in $\left[t^{(\mathrm{B})}(q),\ t^{(\mathrm{E})}(q)\right]$ and is used to eliminate the influence of the sequence length. Finally, the overall video-level similarity score $s_n^{(\mathrm{V})}$ is calculated by accumulating the weighted similarities from multiple levels:

$$s_n^{(\mathrm{V})} = 2^{-L} s_{n,0}^{(\mathrm{V})} + \sum_{l=1}^{L} 2^{l-L-1} s_{n,l}^{(\mathrm{V})} \tag{9}$$

The weight of level l is set to 2^{-L} for $l = 0$, and 2^{l-L-1} for $l = 1, \ldots, L$ to penalize matches in coarser levels. After that, we can obtain a set of candidate matches for q, denoted by $\mathbb{M}_n^{(\mathrm{V})} = \left\{\mathbf{m}_n^{(\mathrm{V})}(q_j, r_i) | r_i \in \mathbb{R}\right\}$, where $\mathbf{m}_n^{(\mathrm{V})}(q, r_i) = \langle q, t^{(\mathrm{B})}(q), t^{(\mathrm{E})}(q), r_i, t^{(\mathrm{B})}(r_i), t^{(\mathrm{E})}(r_i), s_n^{(\mathrm{V})}\rangle$. For each $\mathbf{m}_n^{(\mathrm{V})}(q, r_i) \in \mathbb{M}_n^{(\mathrm{V})}(q)$, we can determine whether it is a copy by comparing $s_n^{(\mathrm{V})}(q, r_i)$ with the threshold θ_n^*.

Because only a set of frame-level matches are needed as its input, TPM is suitable for various detectors using frame-based visual/audio features. It is also computationally efficient: Let R denote the size of the reference database, M be the average number of keyframes in each query video, B be the bin number in the 2D Hough transform, K_1 be the number of similar reference key-frames (audio frames) for each query keyframe (audio frame), and K_2 be the number of voted video-level hypotheses by the Hough transform. Then the computational complexity of TPM for one query can be approximated as

$$R \times B \times \log K_2 + M \times K_1 \times K_2 \times (L + 1) \tag{10}$$

Here the left term is related to the 2D Hough transform and the right one is regarding multigranularities similarity evaluation. Because L, K_1, and K_2 are small numbers in our study and M is also not big for most real-world applications, the overall computational complexity is mainly determined by the 2D Hough transform over a large reference database.

Experiments

To evaluate the performance of our approach, experiments were conducted on the TRECVID-CBCD benchmark dataset used in the 2010 and 2011 contests. In this dataset, the 425-hour-long reference database contains 11,503 videos collected from the Internet that are diverse in content, style, and format and varied in quality. In addition, two databases of query videos, each of which is an average of 73 seconds long, were constructed by applying a combination of eight visual transformations and seven audio transformations (for a total of 56 transformations) to three types of video: reference video only, reference video embedded into a nonreference video, and nonreference video only.[1] Among them, we used the first one, which contains 10,976 videos constructed for the 2010 contest, as the training dataset for soft threshold learning, and we used the other database of 11,256 videos constructed for the 2011 contest as the test set.

In our experiments, we adopt the evaluation metrics used in the TRECVID-CBCD contest.[1] The primary metric, the normalized detection

Table 3. Comparison of individual detectors.

Metric	Detector	V1 (A1)	V2 (A2)	V3 (A3)	V4 (A4)	V5 (A5)	V6 (A6)	V8 (A7)	V10	Average
NDCR	DC-SIFT	0.149	0.075	0.015	0.104	0.030	0.261	0.097	0.202	**0.117**
	DCT	0.970	0.373	0.142	0.097	0.075	0.224	0.522	0.351	**0.344**
	WASF	0.119	0.127	0.127	0.142	0.284	0.276	0.284	–	**0.194**
Mean F1	DC-SIFT	0.943	0.944	0.962	0.952	0.965	0.960	0.957	0.954	**0.955**
	DCT	0.911	0.956	0.963	0.962	0.962	0.954	0.950	0.956	**0.952**
	WASF	0.952	0.951	0.951	0.950	0.948	0.946	0.946	–	**0.949**
MP-Time	DC-SIFT	217	421	209	207	211	204	311	216	**250**
	DCT	6.36	10.20	4.60	4.77	4.60	4.71	8.50	7.31	**6.38**
	WASF	5.49	5.49	5.48	5.48	5.49	5.48	5.49	–	**5.49**

cost rate (NDCR), is used to measure the detection accuracy for each individual transformation. It is calculated as a weighted combination of the probability of a miss error P_{MISS} and the false alarm rate R_{FA}:

$$\begin{aligned}\mathrm{NDCR} &= P_{\mathrm{MISS}} + \beta * R_{\mathrm{FA}} \\ &= \frac{\mathrm{FN}}{N_{\mathrm{Tgt}}} + \frac{c_{\mathrm{FA}}}{c_{\mathrm{MISS}} \times R_{\mathrm{Tgt}}} + \frac{\mathrm{FP}}{T_{\mathrm{Ref}} \times T_{\mathrm{Qry}}}\end{aligned} \qquad (11)$$

where FN and FP stand for false negative (miss) and false positive (false alarm), respectively. c_{FA} and c_{MISS} are the costs of an individual false alarm and an individual miss, respectively ($c_{\mathrm{FA}} = 1$ for BALANCED, $c_{\mathrm{FA}} = 1{,}000$ for NOFA, and $c_{\mathrm{MISS}} = 1$ for both profiles). Without loss of generality, we only report the balanced results. $T_{\mathrm{Tgt}} = 134$ is the total number of copies, and $R_{\mathrm{Tgt}} = 0.005$ is the a priori target rate. $T_{\mathrm{Reg}} = 420$ and $T_{\mathrm{Qry}} = 4.4592$ are the total length (in hours) of the entire reference database and that of the queries for a transformation respectively. Clearly, for a detection system, a smaller NDCR is better.

To measure the localization precision, the second metric, mean F1, is defined as the harmonic mean of precision and recall, where precision is the percentage of the asserted copy that is an actual copy while recall is the percentage of the actual copy that is subsumed in the asserted copy. Thus, if $C(q, r_i)$ holds, F1 represents the extent of the overlap between $\left[t^{(\mathrm{B})}(r_i), t^{(\mathrm{E})}(r_i)\right]$ (the asserted copy) and $\left[t'^{(\mathrm{B})}(r_i), t'^{(\mathrm{E})}(r_i)\right]$ (the actual copy). The third measure, mean processing time (MP-Time), is the mean time (in seconds) to process a query from decoding the video files to yielding the result. All experiments were carried out on a Windows Server 2008 with a 32-Core 2.00-GHz CPU and 32 Gbytes of memory.

Performance of Individual Detectors

The first experiment was to evaluate the performance of the detectors. Table 3 shows the results of detectors based on DC-SIFT, DCT, and WASF.

We can see that the DCT detector performs well on content-preserving transformations such as V4 and V6, but it is vulnerable to content-altering transformations such as V1, V2, V8, and V10. In particular, it is incapable of resisting V1 (camcording), which is indicated by a NDCR of 0.970. Although the NDCR of the DC-SIFT detector is superior to that of the DCT detector on most visual transformations, the copy collections respectively obtained by them do not fully overlap. Generally speaking, the DCT detector has the advantages of excellent mean F1 and relatively short MP-Time. In contrast, the MP-Time of the DC-SIFT detector takes much longer than that of the DCT or WASF detectors.

For audio, the WASF detector can handle simple transformations such as A1–A4 well, while showing relatively poor performance for complex transformations such as A5–A7, which feature mixing-with-speech. Intrinsically, the WASF feature cannot effectively distinguish the mixed speech from audio signal.

The experimental results support our conjecture about the complementarity of detectors based on audio features as well as global and local visual features. That is, none can resist all the transformations; whereas a good overall performance may be achieved by appropriately combining them.

Comparison of DTW and TPM

The second experiment was to compare the performance of different multiscale sequence

Table 4. Comparison between TPM and DTW.

Metric	Detector	V1 (A1)	V2 (A2)	V3 (A3)	V4 (A4)	V5 (A5)	V6 (A6)	V8 (A7)	V10	Average
NDCR	TPM	0.149	0.075	0.015	0.104	0.030	0.261	0.097	0.202	**0.117**
	DTW	0.172	0.090	0.015	0.112	0.045	0.284	0.112	0.224	**0.132**
Mean F1	TPM	0.943	0.944	0.962	0.952	0.965	0.960	0.957	0.954	**0.955**
	DTW	0.901	0.916	0.921	0.917	0.920	0.914	0.913	0.919	**0.915**
MP-Time*	TPM	4	4	4	4	4	4	4	4	**4**
	DTW	84	77	73	79	72	91	77	89	**80**

* MP-Time only includes the time for TPM or DTW, while excluding the time for DC-SIFT feature extraction and searching.

matching methods—that is, DTW using a Viterbi-based frame fusion algorithm[3] and TPM. For simplicity, this experiment is only conducted using the DC-SIFT detector. Parameter validation experiments show that TPM with four levels ($l = 0$, 1 2, 3, thus at most eight temporal segments) can achieve the best matching results. Too few levels (for example, $l = 0$) would lead to including many malposed frame matches, while too many levels (for example, $l = 4$) would miss some short copies due to inadequacy of strictly aligned frame matches.

Table 4 shows that TPM outperforms DTW in terms of all three metrics. Particularly, TPM exhibits better localization precision (with an average mean F1 of 0.955). This is because DTW utilizes more relaxed constraints to cope with temporal transformations (such as frame dropping) and is tolerant to malposed frame matches to some extent, consequently leading to more false alarms in detection and localization. Compared with TPM, DTW is computationally more expensive because it employs a Viterbi-like dynamic programming algorithm on a large amount of frames. The experimental results confirm the effectiveness and efficiency of TPM as a video matching algorithm.

Performance of the Proposed Approach

We used two versions of our implementation in this experiment: one based on three detectors $\mathbb{D}_3^{(S)} = \langle d_{\text{WASF}}^{(S)}, d_{\text{DCT}}^{(S)}, d_{\text{DC-SIFT}}^{(S)} \rangle$ (which we call Soft D3) and one based on two detectors $\mathbb{D}_2^{(S)} = \langle d_{\text{WASF}}^{(S)}, d_{\text{DCT}}^{(S)} \rangle$ (which we call Soft D2). Meanwhile, the hard threshold version $\mathbb{D}_3^{(H)} = \langle d_{\text{WASF}}^{(H)}, d_{\text{DCT}}^{(H)}, d_{\text{DC-SIFT}}^{(H)} \rangle$ (which we call Hard D3), also included in this experiment, achieved the best performance in the TRECVID-CBCD 2011 contest at the cost of a long time for manual threshold tuning. Here Soft D2 and Hard D3 can be viewed as two baseline methods of the proposed implementation (Soft D3). Moreover, we also included the two best approaches from 21 other participants in this contest, CRIM-VISI (which obtained excellent performance using nearest-neighbor mapping) and INRIA-LEAR (which features the early fusion of multiple audio-visual features), and the median performances on each transformation among all approaches in this contest (denoted by "Median").[1] Note that their MP-Times are only used as references because they were executed on different platforms.

Table 5. Comparison between the proposed approach and results from the TRECVID-CBCD 2011 evaluation.

Method		Average NDCR	Average mean F1	Average MP-Time
Proposed approach	Soft D3	**0.054**	**0.951**	163.184
	Soft D2	0.178	0.950	9.752
TRECVID-CBCD 2011 Evaluation	Hard D3	0.055	0.950	172.291
	CRIM-VISI	0.159	0.715	2,792
	INRIA-LEAR	0.271	0.944	2,079
	Median	1.050	0.889	191.535

Table 5 summarizes the average performances for different methods, and Figure 5 shows their performance curves over 56 transformations. From the results, we can see that by integrating complementary detectors, both Soft D3 and Hard D3 can dramatically improve detection accuracy, localization precision, and efficiency (note that their average MP-Time is even shorter than that of the DC-SIFT detector). Meanwhile, by utilizing the soft threshold learning algorithm, SoftD3 can achieve a NDCR (paired t test, $p = 0.774$) and mean F1 ($p = 0.0476$) comparable to

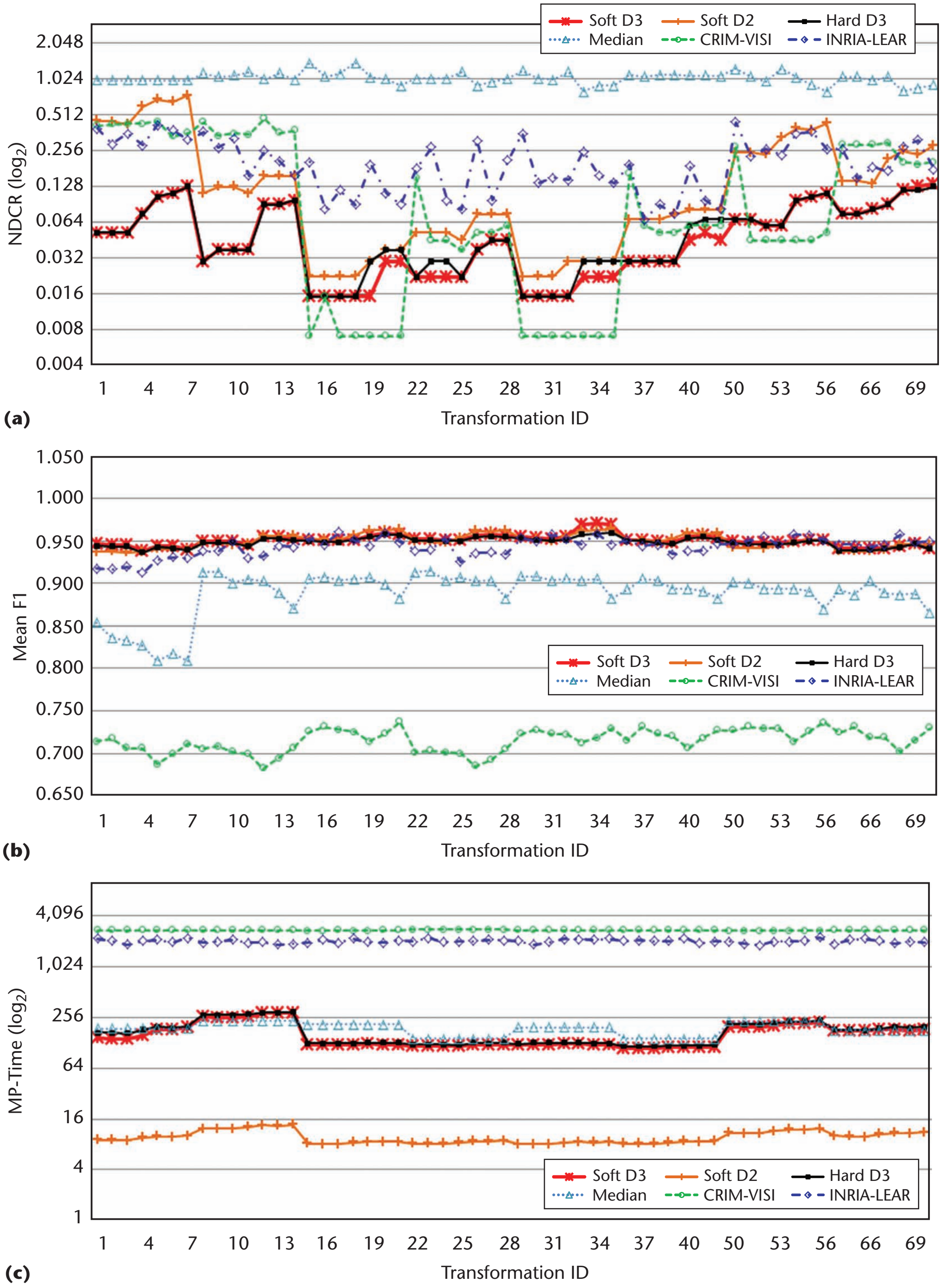


Figure 5. Performance curves of different methods over 56 transformations. (a) NDCR (y-axis in log2 coordinate), (b) mean F1, and (c) MP-Time (y-axis in log coordinate).

Hard D3. As Figure 5 shows, despite being statistically insignificant on the overall detection accuracy, Soft D3 exhibits slightly better detection accuracy on several transformations. More importantly, compared with Hard D3, which utilizes some dataset-related priors to manually tune the thresholds (for example, to set a high confidence for the DC-SIFT detector on V3 and similarly for the DCT detector on V4 and V6), Soft D3 does not specify such priors in the threshold learning process and thus can have a better generalization on various databases.

Among all the methods, Soft D3 achieves a compelling performance; it wins 35 best NDCR and 35 best mean F1, and its MP-Time is better than the median ones on most transformations. Comparatively, Hard S3 wins 32 best NDCR (30 tied for first place with Soft D3) and two best mean F1, while CRIM-VISI wins 20 best NDCR but has poor localization precision, with an average mean F1 of 0.715. By using only two efficient detectors, Soft D2 can also achieve a competitive NDCR and excellent mean F1 within a relatively short MP-Time, demonstrating the scalability of the proposed cascading framework. Overall, this excellent performance shows the effectiveness and efficiency of our approach.

The performance results for the different methods demonstrate the shows the effectiveness and efficiency of our approach.

Conclusion

The experimental results show that, on the TRECVID-CBCD 2011 benchmark dataset, our approach can achieve the best copy detection accuracies and excellent localization precisions for the majority of transformations.

In our approach, all query videos, whichever transformations they are subjected to, are processed by the same chain of detectors with the identical decision thresholds. However, different transformations (such as V1 versus V4 in Figure 1a) may exhibit significantly distinct audio-visual properties. Thus in the future work, we intend to integrate the transformation recognition module in the cascading framework and use different detector chains for different categories of transformations. It is expected to achieve better performance and higher scalability on practical copy-detection applications. **MM**

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grants 61035001 and 60973055, and the National Basic Research Program of China under a grant 2009CB320906.

References

1. W. Kraaij and G. Awad, *TRECVID 2011 Content-Based Copy Detection: Task Overview,* Nov. 2011; www-nlpir.nist.gov/projects/tvpubs/tv11.slides/tv11.ccd.slides.pdf.
2. Y.H. Tian et al., ''A Multimodal Video Copy Detection Approach with Sequential Pyramid Matching,'' *Proc. 18th IEEE Int'l Conf. Image Processing* (ICIP), IEEE CS, 2011, pp. 3629–3632.
3. S.K. Wei et al., ''Frame Fusion for Video Copy Detection,'' *IEEE Trans. Circuits and Systems for Video Technology,* vol. 21, no. 1, 2011, pp. 15–28.
4. P. Viola and M. Jones, ''Rapid Object Detection Using a Boosted Cascade of Simple Features,'' *Proc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition* (CVPR), vol. 1, IEEE CS, 2001, pp. 511–518.
5. J.P. Chen and T.J. Huang, ''A Robust Feature Extraction Algorithm for Audio Fingerprinting,'' *Proc. 9th Pacific Rim Conf. Multimedia: Advances in Multimedia Information Processing* (PCM), Springer-Verlag, 2008, pp. 887–890.
6. A. Bosch, A. Zisserman, and X. Muoz, ''Scene Classification Using a Hybrid Generative/Discriminative Approach,'' *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 30, no. 4, 2008, pp. 712–727.
7. K. Terasawa and Y. Tanaka, ''Spherical LSH for Approximate Nearest Neighbor Search on Unit Hypersphere,'' *Proc. 10th Int'l Conf. Algorithms and Data Structures* (WADS 2007), LNCS 4619, Springer-Verlag, 2007, pp. 27–38.
8. B. Liu et al., ''Real-Time Video Copy-Location Detection in Large-Scale Repositories,'' *IEEE Multimedia,* vol. 18, no. 3, 2011, pp. 22–31.
9. S. Lazebnik, C. Schmid, and J. Ponce, ''Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories,'' *Proc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition* (CVPR), vol. 2, IEEE CS, 2006, pp. 2169–2178.

Yonghong Tian is a professor in the School of Electrical Engineering and Computer Science at Peking University, China. His research interests include

computer vision and multimedia content analysis. Tian has a PhD in computer science from the Institute of Computing Technology at the Chinese Academy of Sciences, China. He is a senior member of IEEE and a member of ACM. Contact him at yhtian@pku.edu.cn.

Tiejun Huang (corresponding author) is a professor in the School of Electrical Engineering and Computer Science at Peking University, China. His research interests include digital library and video coding and understanding. Huang has a PhD in image recognition and artificial intelligence from Huazhong University of Science and Technology, China. He is a senior member of IEEE and a member of ACM. Contact him at tjhuang@pku.edu.cn.

Mengling Jiang is a doctoral student in the Department of Computer Science at Rutgers University. His research interests include computer science. Jiang has an MS from the School of Electrical Engineering and Computer Science at Peking University, China. Contact him at mljiang@pku.edu.cn.

Wen Gao is a professor in the School of Electrical Engineering and Computer Science at Peking University, China. His research interests include image and video communication, computer vision, and artificial intelligence. Gao has a PhD from the University of Tokyo. He is an IEEE fellow and an academician in the Chinese Academy of Engineering. Contact him at wgao@pku.edu.cn.

cn *Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.*