Fast Content-Based Mining of Web2.0 Videos

Sébastien Poullot[12], Michel Crucianu[1], and Olivier Buisson[2]

[1] Vertigo - CNAM, 292 rue St Martin, 75141 Paris cedex 03, France
Michel.Crucianu@cnam.fr
[2] Institut National de l'Audiovisuel, 94366 Bry-sur-Marne, France
SPoullot@ina.fr, OBuisson@ina.fr

Abstract. The accumulation of many transformed versions of the same original videos on Web2.0 sites has a negative impact on the quality of the results presented to the users and on the management of content by the provider. An automatic identification of such *content links* between video sequences can address these difficulties. We put forward a fast solution to this video mining problem, relying on a compact keyframe descriptor and an adapted indexing solution. Two versions are developed, an off-line one for mining large databases and an online one to quickly post-process the results of keyword-based interactive queries. After demonstrating the reliability of the method on a ground truth, the scalability on a database of 10,000 hours of video and the speed on 3 interactive queries, some results obtained on Web2.0 content are illustrated.

1 Introduction

A very large share of the videos currently hosted by Web2.0 sites is issued from professional original video content that is modified or repurposed and eventually uploaded by many different users. In many cases, the number of different, transformed versions of a same content that are stored on a site is high and, given the wide availability of easy to use video editing software, can be expected to keep growing. The frequent transformations include resizing, compression, colorimetric changes, insertion of logos or various artifacts, modification of the time line and mixing. The accumulation of very many transformed versions of the same content has a strong negative impact on the quality of the results presented to the users and on the management of the available content. Frequently, consecutive results to a query are versions of a same original video, forcing the user to skim through several pages of results in order to find the next truly different video. Also, a very large share of the videos that a Web2.0 site store is actually useless (and even detrimental) because it consists of such duplicates.

These examples bring out the need for identifying the *content links* between the stored videos, and for using them to unclutter and structure the content of the video databases, with the ultimate aim to improve user experience with Web2.0 sites. The textual annotations currently available for the videos can very seldom bring to light the links between the many versions of a same original video. The tool of choice for finding such links is content-based video copy detection (CBVCD), where the relation of "copy" holds between two videos if they are transformed variants of the same original video. We consider that CBVCD can be used for this purpose in two complementary ways. First, an *off-line process* can be periodically launched on large but specific shares of the database in order to identify near-duplicates and videos that share a common sub-sequence. A good reference is the volume of videos returned by a broad query; e.g., on a popular Web2.0 site the query "Madonna" returns 116,000 answers (10,000 hours of video). The results of this process can be employed to remove e.g. the lower quality versions of the videos, thus reducing storage requirements and improving the diversity of the answers to future user queries. They also allow to establish navigation links based on common sub-sequences between videos that do not share keywords, or to improve the quality of the keywords associated to a video by exploiting the keywords of its different versions.

Second, when the system receives a keyword-based query, it can employ a *query-dependent online process* to structure the preliminary results before returning them to the user. This process can remove recently uploaded duplicates and group the results in a query-specific way or modify the "default" ranking by exploiting the content links. Such an online process should be able to answer almost immediately when given about 1,000 videos (websites usually return the top 1,000 answers even if more relevant videos are available) of a few minutes each (most videos are less than 5 minutes long), i.e. around 80 hours of video.

Several problems have to be solved in order to achieve such fast online mining and scalable off-line processing. The diversity and the amplitude of the transformations that can be encountered (regarding both the individual frames and the temporal arrangement of video sub-sequences) rule out simple descriptions of the videos. But robust descriptions typically require expensive matching operations that can make CBVCD-based mining hopelessly slow. To address this challenge, we suggest to represent videos as sequences of keyframes and propose a compact keyframe signature based on robust local descriptions; then, we develop an indexing method that can significantly speed up the identification of the sets of similar keyframe signatures and further adapt it to the online context.

After reviewing some recent work on video mining by copy detection in Sect. 2 we put forward our mining solution in Sect. 3. Subsection 3.1 presents a compact keyframe signature, called Glocal, subsection 3.2 describes a new redundant indexing scheme that allows to find the sets of similar signatures fast, and subsection 3.3 shows how the links between video sequences are established. In Sect. 4 we first demonstrate the reliability of the method on a ground truth, the scalability on a database of 10,000 hours of video and the speed on three interactive queries. Then, results obtained on Web2.0 content are illustrated and some applications are discussed.

2 Content-Based Video Copy Detection for Video Mining

There are several recent developments in CBVCD methods (see e.g. [6] and references therein), but few attempt to apply CBVCD to video mining. The early approaches to video mining in [9] and [16] focus on news shows and develop CB-

VCD solutions. The databases are not very large and the descriptors employed are rather simple, limiting the range of transformations that still allow the detection of copies, but these proposals demonstrate the relevance of CBCD for video mining. In [15], near-duplicate frames are identified in the TRECVID 2004 video corpus, but 16 seconds are required for searching 150 frames in 10 minutes of video (600 frames). An application of CBVCD to the elimination of video duplicates in Web search is proposed in [14]; global descriptors help separating the least similar videos, then local descriptors allow to refine duplicate detection. However, several minutes are required for returning the top 10 answers (among the 600 preliminary results) to a keyword-based query, which is too long for performing the query-dependent online processing we need. The scope of the video mining stage in [11] is broader than copy detection, since it attempts to identify content links between e.g. same or similar objects in different scenes; this requirement and the nature of the descriptions employed significantly reinforce the scalability and the speed challenges. While some generic CBVCD methods successfully address the scalability issue for monitoring a video stream against a database of original content, their direct application to video mining would be unacceptably slow. For example, the method in [8] needs about 20 days to identify the links between keyframes in a database of 10,000 hours of video.

Some interesting methods video mining do not attempt to identify copies. The occurrences of the same scenes (from different viewpoints) are detected by using the discontinuities in the trajectories of interest points in [10] or by employing flash patterns in [12]. News subjects are tracked by using both video keyframe similarity and automatic audio transcriptions in [17], [13].

3 Proposed Video Mining Solution

To reliably identify the versions of videos in spite of the transformations that modify the individual frames and their temporal arrangement, we represent videos as sequences of keyframes and propose a compact keyframe signature based on robust local descriptions. The mining process first discovers the pairs of similar keyframes in the database and then employs them to find similar video sequences. To reduce the number of similarity computations required for discovering the pairs of keyframes, we develop an indexing scheme based on dividing the database of keyframe signatures into segments such that, in each segment, the similarity between any two signatures is above a threshold; the search for similar keyframes is then only performed within each segment.

3.1 Keyframe Description

Given the diversity and the amplitude of the transformations that can be encountered in the different versions of a video, a reliable identification of these versions requires local descriptions for the keyframes. But the set of local signatures describing a keyframe is not very compact, so more frequent access to mass storage can be needed during mining, which slows down the process. Moreover, a costly matching step is required for going from the level of individual local signatures to the level of keyframes. We have defined a keyframe descriptor, called *Glocal*, that attempts to find a good compromise between the robustness of copy detection with local signatures and the compactness of keyframe-level signatures. To obtain the Glocal signature of a keyframe, a maximum of 20 interest points are found using the improved Harris detector and for each point a 20-dimensional spatio-temporal differential description (local signature) is computed as in [8]. The 20-dimensional description space containing these local signatures is partitioned at a limited depth h, which produces 2^h cells that are numbered. The Glocal signature of the keyframe is the binary vector where the bit i is set to 1 only if the local signature of at least one interest point of the keyframe falls within cell i. For the example in Fig. 1, the Glocal signature is **1 0 0 0 0 1 0 0 0 0 1 0 0 1 0 1**.

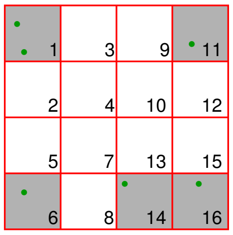

Fig. 1. Synthesis of a *Glocal* signature from the descriptions of 5 interest points; in this simplified example the description space is 2-dimensional and partitioned at $h = 4$

The analysis of a large database shows that for a depth of $h = 8$ the average number of bits set to 1 in a Glocal signature is 17.4, while the average number of local descriptors is 19.4 for a keyframe. The local signatures representing a keyframe are well spread, so if only their coarse positions are stored the loss of information is limited. For $h = 8$, a Glocal signature takes $2^h = 256$ bits; to save space, for higher depths only the *positions* of the bits set to 1 are stored.

The similarity between Glocal signatures is given here by the Dice coefficient, $S_{\text{Dice}}(\mathbf{g}_1, \mathbf{g}_2) = \frac{2|\mathcal{G}_1 \cap \mathcal{G}_2|}{|\mathcal{G}_1|+|\mathcal{G}_2|}$, where $\mathcal{G}_i$ is the set of bits set to 1 in the signature $\mathbf{g}_i$ and $|\cdot|$ denotes set cardinality. A study of the impact of the transformations shows that a keyframe can be safely considered a copy of another if their similarity is above $\theta = 0.55$.

3.2 Keyframe Indexing for Off-line or Online Mining

If the similarity was computed for every pair of signatures, the time complexity of the identification of pairs of similar keyframes would be quadratic in the size of the database (e.g. 4500 seconds are needed to mine 100 hours of video, not compatible with online processing). Access to mass storage further slows down mining, especially for the off-line process. An indexing solution is thus required.

The indexing scheme we developed is inspired by redundant indexing methods such as LSH [4], RBV [7], OMEDRANK [3] or PvS [5], and by proposals addressing similarity joins, like [1] and [2]. We divide the database of keyframe signatures into segments (or *buckets*) such that, in each segment, the similarity between any two signatures is above a threshold; the search for similar keyframes is then only performed within each bucket. Time and storage complexity depend on the total number of buckets and on their lengths; if the number and lengths are small, the gain can be significant with respect to the computation of all the similarities. The number of different buckets can be reduced using selection rules as shown below. The size of the buckets is limited for the online version.

A Glocal signature is a set of "words", a word being the position of a bit set to 1. With these words one can build "sentences" of various lengths. A bucket consists of all the Glocal signatures in the database that contain a specific sentence; the bucket can be stored as an inverted list. There are $C_{256}^{3} = 2,763,520$ possible sentences of length 3 for Glocal signatures of 256 bits ($h = 8$) and a signature can contain $C_{20}^{3} = 1120$ of them (3 positions out of 20), i.e. be indexed in at most 1120 buckets. To reduce this bound, we only consider sentences composed of neighboring bits (not separated by any other bit set to 1), 1-out-of-2 bits (separated by 1 bit set to 1), 1-out-of-3 bits and 1-out-of-4 bits. For the Glocal signature **1 0 0 0 0 1 0 0 0 0 1 0 0 1 0 1** the sentences of neighboring bits are 1-6, 6-11, 11-14 and 14-16, those of 1-out-of-2 bits are 1-11, 6-14 and 11-16, while those of 1-out-of-3 bits are 1-14 and 6-16. Using these rules, the average number of sentences per signature is of only 48. To find the pairs of similar keyframes, the similarities between all the signatures are computed within every bucket; this independent mining of the buckets makes this method easy to parallelize. If the similarity is above the θ threshold, the identifiers of both keyframes are stored and later used for finding similar video sequences. But if the number of buckets is reduced, some pairs of similar keyframes might no longer be found together in any remaining bucket and would thus be missing from the results of mining. A collision analysis shows that for a partitioning depth $h = 8$ with sentences of 3 words and the above selection rules, a recall of 0.9 is obtained.

Depending on the video database, some of the buckets can be large and are retrieved from mass storage one after the other for mining. This can make the duration of online mining long and unpredictable. To keep it under control, an upper bound is used in online mining for the size of the buckets, so that all the buckets can hold in main memory. When a bucket reaches the bound, no further Glocal descriptors are added to it; a small upper bound implies faster processing at the expense of lower recall (fewer pairs of similar keyframes are found).

3.3 Linking Video Sequences

The second stage of mining sets up the links between video sequences that are transformed versions of a same content and builds the graphs that summarize the results. To begin with, similar keyframes are grouped together by pairs of video identifiers; for each pair, the keyframes are sorted by increasing time codes. Then, pairs of corresponding sequences are built by the stepwise addition of pairs of linked keyframes that verify temporal consistency conditions. Consider two different sequences of identifiers $\mathrm{ID_x}$ and $\mathrm{ID_y}$. First, the temporal gap between the last keyframe in a sequence (time code $\mathrm{Tc_{x,l}}$) and a candidate keyframe to be added to the same sequence (time code $\mathrm{Tc_{x,c}}$) should be lower than a threshold τ_g: $\mathrm{Tc_{x,c}} - \mathrm{Tc_{x,l}} < \tau_g$. This allows small gaps due to the absence of a few connected keyframes, as a result of post-processing operations (addition or removal of several frames) or of false negatives in the detection of connected keyframes. Second, a limited amount of temporal offset (jitter) τ_j, caused by post-processing operations or by instabilities of the keyframe detector, is also tolerated: $|(\mathrm{Tc_{x,c}} - \mathrm{Tc_{x,l}}) - (\mathrm{Tc_{y,c}} - \mathrm{Tc_{y,l}})| < \tau_j$, where $\mathrm{Tc_{x,l}}$ is the time code of the last keyframe in $\mathrm{ID_x}$, $\mathrm{Tc_{y,l}}$ the time code of the last keyframe in $\mathrm{ID_y}$ and $\mathrm{Tc_{x,c}}$, $\mathrm{Tc_{y,c}}$ are the time codes of the candidate keyframes. Third, sequences should be longer than a minimal value τ_l to be considered valid; this removes very short detections, typically false positives.

4 Evaluation Results and Illustrations

The reliability and the scalability of the proposed mining method should be evaluated first. Then, the online mining process is illustrated on the results returned by a Web2.0 site for two keyword-based queries and applications are discussed. While the method is easy to parallelize, the following results are obtained on a single core 3 GHz CPU with 4 Gb of RAM.

4.1 Experimental Validation

Reliability on a ground truth. The public video copy detection benchmark[3] of CIVR 2007 (63 hours, 156,238 keyframes) was employed to test the off-line and online versions, by adding the queries (ST1 and ST2) to the database. We require a precision of 1, i.e. no wrong link should be established. For both versions, the recall obtained was of 0.7 with a similarity threshold $\theta = 0.55$ and of 0.8 with $\theta = 0.5$. The online process requires 5 seconds to build the index and 13.5 seconds to identify the corresponding video sequences.

Scalability to large databases. To measure the time required for mining a large database (e.g. comparable to the volume of videos annotated with the keyword “Madonna” on a popular Web2.0 site) a 10,000 hours database was created by randomly selecting videos from the INA archive. This volume also corresponds to 416 days of broadcast for one TV channel. The Glocal signatures of the 28.7×10^6 keyframes were obtained using a partitioning depth $h = 9$. Mining this database required 82.5 hours (3.5 days).

Speed in processing the results of interactive queries. To test the online version of our mining method, the top 1,000 answers returned by a Web2.0 site to 3 simple keyword-based queries were employed (the number of videos considered was actually lower because transcoding to MPEG failed for some

[3] `http://www-rocq.inria.fr/imedia/civr-bench/`

of them). For the query “Zidane” (soccer player) 739 videos (35 hours) were processed, for the query “Justin Timberlake” 907 videos (69 hours) and for the query “Madonna” 925 videos (69 hours). Table 1 presents the results of the online version of the method for $\theta = 0.55$ (and $h = 8$). The time required for mining is compatible with interactive retrieval and can be further reduced by simple means (optimized code, progressive display, etc.). Given the limited sizes of these databases, all the data holds in main memory without any significant reduction in bucket size.

Table 1. Online mining of the results returned by a Web2.0 site to 3 queries

Keyword-based query	“Zidane”	“Justin Timberlake”	“Madonna”
Database size (hours)	35	63	69
Number of keyframes	92,452	155,872	182,613
Time for base construction (seconds)	5	7	7
Time for linking keyframes (seconds)	8	17	23
Time for linking sequences (seconds)	1	2	2
Total time required for mining (seconds)	14	26	32

4.2 Illustration on the Results of Two Keyword-Based Queries

Results obtained for the “Zidane” query are shown in Fig. 2 and those for the “Madonna” query in Fig. 3. For both figures, the top left image presents the resulting graph between corresponding video sequences. The layout of the graph, with each connected component visually separated from the others, is obtained in less than 2 seconds. Note that if two sequences are issued from a same broadcast (same ID) and have a non-zero overlap, then they are represented by a single vertex. So, if a vertex X is linked to vertices Y and Z, and these two links correspond to different duplicate sequences having a very small overlap ($< \tau_l$), then there is no direct link between Y and Z. For both queries, no wrong link was found in the results (precision is 1) but recall could not be measured. The metadata on every link includes the length of the detection and the minimal, maximal and mean similarities between the keyframes composing the sequences. This information can be used for modifying the display of a graph.

The “Zidane” graph (Fig. 2) has many small connected components and a few large ones. A single representative vertex (video sequence) could be returned for each component; the number of vertices of the component is relevant for its ranking. The largest connected subgraph A assembles edited versions of several compilations and goal sequences. In order to keep some diversity in the results, this subgraph can be used to select one representative for each of the strongly connected components (a keyframe is shown for each). For the small connected components in region B some vertices are illustrated by *connected* keyframes. Short sequences that are present in very many videos can also be identified on the graph of video sequences; these sequences can be taken out from the full videos and returned as well-ranked answers. The bottom right part of Fig. 2 shows another graph where a link is set between two vertices only if at least one

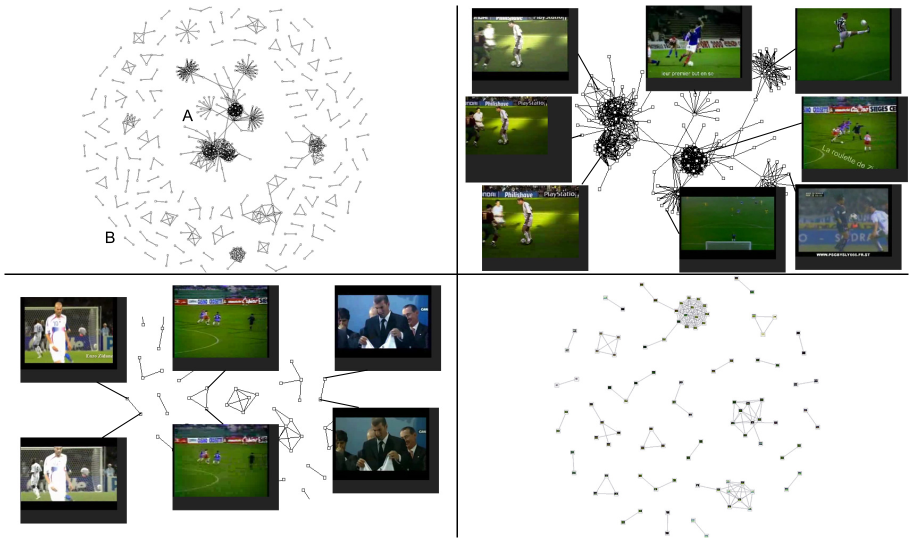

Fig. 2. Graph between sequences for "Zidane" query (top left), illustrations for subgraph A and subgraphs B, and graph restricted to entire copies (bottom right)

of the vertices is an entire video that is included in the other. This graph allows to find entire copies and to identify compilations that contain entire videos; the videos that are included in others and do not have a better quality can be removed from the results and probably also from the database.

On the "Madonna" graph (Fig. 3), the largest connected subgraph A contains sequences from the many different video-clips of the "4 minutes" single; the same videos are cut into small segments (some shorter than $\tau_l = 4$) that are then transformed and assembled together. Some versions are official releases, the others are created by enthusiasts. In the "double star" subgraph B, the centers of the stars are two very different versions of a video-clip (same piece of music) that group together video sequences also found in the outer vertices. The bottom right part of Fig. 3 shows the graph between entire videos, where a link is set between two vertices if they include versions of at least one common subsequence. The central subgraph allows to identify the video C that is a comprehensive user-created compilation of many different video-clips.

The most frequent transformations we found in Web2.0 data are the reassembly of short sequences, the speedup or slowdown (by as much as 20%), strong compression and scaling. They are quite demanding for the CBVCD, so some connections between video sequences can remain undetected. The presence of many versions with various transformations may still allow to find an indirect path between such sequences. The online mining process can be used to filter out some preliminary answers to a query and to re-rank the results returned to the user. Also, an innovative interface could exploit some of the identified links and provide new ways to navigate the content.

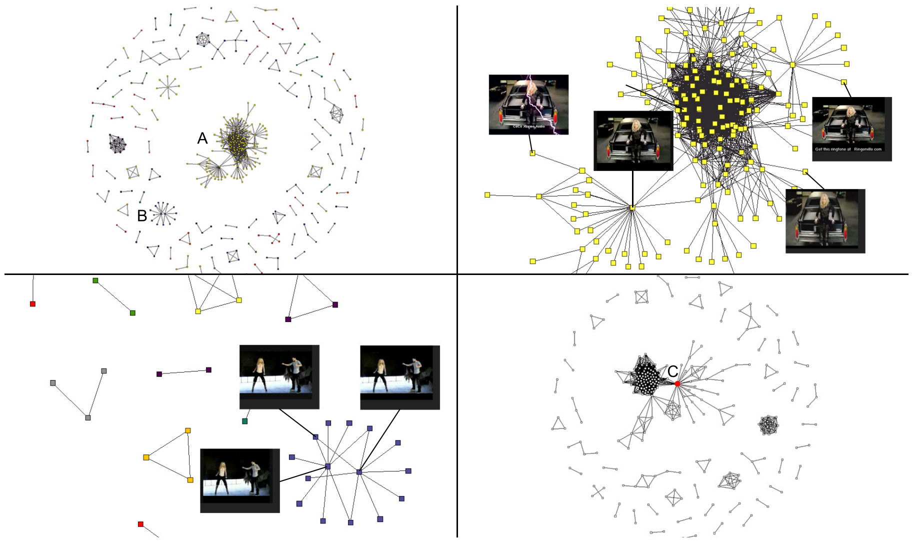

Fig. 3. Graph between sequences for "Madonna" query (top left), illustrations for subgraphs A and B, and graph between entire videos (bottom right)

5 Conclusion

To address the problems posed by the accumulation of very many versions of the same video content on Web2.0 sites, we suggest to mine the video database using content-based video copy detection. But the reliable identification of the transformed videos requires robust local image descriptions and associated matching operations that can make the process hopelessly slow. We propose a compact keyframe signature based on local descriptions and develop an indexing method that can significantly speed up the identification of the sets of similar keyframe signatures. The reliability of this mining method is demonstrated on a public ground truth and its scalability on a database of 10,000 hours of video. Then, we show that the online version of the method can indeed be used during interactive retrieval. The results obtained on the answers returned by a Web2.0 site to two interactive queries are illustrated and ways to exploit them are discussed. All these results show that the compact keyframe description preserves the main information required for copy detection and that the indexing scheme allows to achieve a significant speed-up. The possibilities of exploiting this mining solution for Web2.0 content should be further explored.

6 Acknowledgments

This work is part of the Sigmund project financed by the ANR (France). Many thanks to Jérôme Thièvre (INA) for the graph visualization interface.

References

1. A. Arasu, V. Ganti, and R. Kaushik. Efficient exact set-similarity joins. In *Proc. 32nd intl. conf. on Very Large Data Bases*, pp. 918–929. VLDB Endowment, 2006.
2. R. J. Bayardo, Y. Ma, and R. Srikant. Scaling up all pairs similarity search. In *Proc. 16th intl. conf. on World Wide Web*, pp. 131–140, New York, NY, USA, 2007. ACM.
3. R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proc. ACM SIGMOD intl. conf. on Management of Data*, pp. 301–312, New York, NY, USA, 2003. ACM.
4. A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. 25th intl. conf. on Very Large Data Bases*, pp. 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Inc.
5. H. Lejsek, F. H. Ásmundsson, B. T. Jónsson, and L. Amsaleg. Scalability of local image descriptors: a comparative study. In *Proc. 14th ACM intl. conf. on Multimedia*, pp. 589–598, New York, NY, USA, 2006. ACM.
6. J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *Proc. 6th ACM intl. Conf. on Image and Video Retrieval*, pp. 371–378, New York, NY, USA, 2007. ACM.
7. J. Oostveen, T. Kalker, and J. Haitsma. Feature extraction and a database strategy for video fingerprinting. In *Proc. 5th Intl. Conf. on Recent Advances in Visual Information Systems*, pp. 117–128, London, UK, 2002. Springer-Verlag.
8. S. Poullot, O. Buisson, and M. Crucianu. Z-grid-based probabilistic retrieval for scaling up content-based copy detection. In *Proc. ACM intl. Conf. on Image and Video Retrieval*, pp. 348–355, Amsterdam, The Netherlands, July 2007.
9. S. Satoh. News video analysis based on identical shot detection. In *Proc. IEEE Intl. Conf. on Multimedia and Expo*, pp. 69–72, 2002.
10. S. Satoh, M. Takimoto, and J. Adachi. Scene duplicate detection from videos based on trajectories of feature points. In *Proc. intl. workshop on Multimedia Information Retrieval*, pp. 237–244, New York, NY, USA, 2007. ACM.
11. J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. 9th IEEE Intl. Conf. on Computer Vision*, pp. 1470–1477, Washington, DC, USA, 2003. IEEE Computer Society.
12. M. Takimoto, S. Satoh, and M. Sakauchi. Identification and detection of the same scene based on flash light patterns. In *Proc. IEEE Intl. Conf. on Multimedia and Expo*, pp. 9–12, Los Alamitos, CA, USA, 2006. IEEE Computer Society.
13. X. Wu, A. G. Hauptmann, and C.-W. Ngo. Novelty detection for cross-lingual news stories with visual duplicates and speech transcripts. In *Proc. 15th intl. conf. on Multimedia*, pp. 168–177, New York, NY, USA, 2007. ACM.
14. X. Wu, A. G. Hauptmann, and C.-W. Ngo. Practical elimination of near-duplicates from web video search. In *Proc. 15th intl. conf. on Multimedia*, pp. 218–227, New York, NY, USA, 2007. ACM.
15. X. Wu, W.-L. Zhao, and C.-W. Ngo. Near-duplicate keyframe retrieval with visual keywords and semantic context. In *Proc. 6th ACM intl. Conf. on Image and Video Retrieval*, pp. 162–169, New York, NY, USA, 2007. ACM.
16. F. Yamagishi, S. Satoh, and M. Sakauchi. A news video browser using identical video segment detection. In K. Aizawa, Y. Nakamura, and S. Satoh, eds., *PCM (2)*, vol. 3332 of *Lecture Notes in Computer Science*, pp. 205–212. Springer, 2004.
17. Y. Zhai and M. Shah. Tracking news stories across different sources. In *Proc. 13th ACM intl. conf. on Multimedia*, pp. 2–10, New York, NY, USA, 2005. ACM.