Video Copy Detection Based on Bag of Trajectory and Two-Level Approximate Sequence Matching

Wei-Ta Chu
Department of Computer Science and Information Engineering,
National Chung Cheng University
Chiayi, Taiwan
wtchu@cs.ccu.edu.tw

Po-Chi Chuang
Department of Computer Science and Information Engineering,
National Chung Cheng University
Chiayi, Taiwan
b22585699@gmail.com

Jen-Yu Yu
Information and Communication Research Labs,
Industrial Technology Research Institute
Hsinchu, Taiwan
KenvinYu@itri.org.tw

***Abstract*—We present a video copy detection system that detects video copy segments based on the task settings and dataset in TRECVID 2010. Contributions of this work are two-fold. First, we extract feature-based trajectories from videos, and then model trajectories by a bag of word model. This representation effectively describes information of object movement, and is robust to various visual transformations. Second, to find locally optimal matching between the query video and videos in database, we conduct approximate sequence matching by first finding the longest common subsequence, followed by localizing the max-sum segment. We compare our method with a watershed-based approach and demonstrate the effectiveness and robustness of the proposed method.**

Keywords-video copy detection; bag of trajectory; the maximum-sum segment algorithm; approximate sequence matching

I. INTRODUCTION

Tremendous amounts of videos have been created, edited, and shared on the internet. Recently, the video sharing site YouTube [1] has become the second largest search engine due to its extremely easy video sharing and search functionalities. Everyone can produce their own videos and disseminate them quickly via such kind of Web 2.0 platform. Although users can easily distribute and retrieve videos from the web, convenience of video sharing deteriorates the problem of copyright infringement and video counterfeits. Therefore, videos of similar content flood on the web, and we can easily obtain many copies with little variations, which may be illegally edited by unknown users.

Recently, various video copy detection systems [2] have been designed to detect videos with duplicate content. With effective video copy detection, video copies can be clustered together to facilitate efficient browsing. On the other hand, detecting similar videos forms the foundation of video retrieval systems. From the perspective of copyright protection, finding video copies can be used to monitor dissemination situations and examine whether a copyright protected video is illegally edited or used. Because the importance of video copy detection is well recognized around the research community, a task force has been established in TRECVID [3] since year 2008, in which a common video database is built to prompt development of various techniques.

Different video copies of the same content often have variations caused by various transformations, such as change of gamma, strong re-encoding, flip, and pattern insertion. To robustly detect video copies with these transformations, many works extract global descriptors like color histogram, and local descriptors like SIFT features [4], to represent video frames. Spatiotemporal relationships between video frames are considered to find duplicate video segments [15].

Motivated by the work proposed in [5], we extract motion trajectories from video shots and encode trajectories by a bag of word model. After appropriate weightings on different bag of words, video shots are transformed into feature vectors, by which shot-based video matching is conducted. Different from the dataset used in [5], we follow the content-based copy detection task in TRECVID 2010, in which a video query may be significantly shorter than the targeted videos, and the video query may be concatenated with irrelevant video segments of arbitrary lengths at the beginning or at the end. Therefore, we introduce the maximum-sum segment algorithm to find locally optimal matching between video sequences.

Contributions of this paper are summarized as follows.

- We extend the idea of bag of word to model motion trajectories in video shots. We improve the work proposed in [5] by introducing faster feature extraction/matching, and investigating the effectiveness of the method in a more realistic situation described in TRECVID 2010.
- We applying the maximum-sum algorithm to find locally optimal matching between two video sequences. This algorithm is originally designed to solve constrained sequence matching problem in bioinformatics. With this algorithm, we are able to well handle the characteristics of TRECVID 2010 benchmark, and provide superior detection performance in this challenging dataset.

The rest of this paper is organized as follows. Section II provides a literature survey and a brief introduction of content-based copy detection in TRECVID. Section III gives

the system overview. In Section IV, we describe the idea of bag of trajectory, and transform videos into a series of feature vectors. Video copy detection based on approximate sequence matching is described in Section V. Section VI provides the experimental results based on TRECVID datasets, followed by the concluding remarks in Section VII.

II. Related Work

A. Literature Survey

With the development of Muscle benchmark [12] in year 2007, researchers have the common platform to compare their research results on video copy detection, and therefore it inspires many studies. Law-To et al. [2] gives a comparative study on video copy detection proposed in early years (~2007). Chiu et al. [13] transform video copy detection as a partial matching problem in a probabilistic model. They are devoted to develop a framework robust against spatial and temporal variations, and report relatively fewer experimental results. Yeh and Cheng [14] view video copy detection as a sequence matching problem, which is the most popular viewpoint in this research. As large amounts of sequence matching should be performed, they propose a two-level filtration approach to accelerate the matching process. Douze et al. [15] match individual frames and then verify their spatio-temporal consistency. Local feature indexing method is proposed to make video copy detection robust to video transformations and efficient in terms of memory usage and computation time. Wu et al. [5] propose the idea of representing videos by motion trajectories. The bag of word model is used to characterize basic trajectory elements. Finally, the watershed algorithm is used to find partial matching between the query video and videos in the database.

Motivated by the work proposed in [5], we develop an efficient way to construct trajectories. Based on the bag of trajectory representation, we propose an approximate sequence matching method that first finds the longest common subsequence between two sequences and then finds the maximum-sum segment to localize the best matching. Performance comparison is conducted in the experiment section.

B. TRECVID Benchmark

Starting from year 2008, TRECVID [3] pays attention to video copy detection, or more generally content-based copy detection, due to potential applications of copyright control, business intelligence, advertisement tracking, etc. In TRECVID 2010, there are totally about 12000 videos with totally 400 hours in the reference dataset. Video content in this task is mainly from TV shows or news.

A query video is a segment of video derived from a video in the database, by means of various transformations. It would be concatenated with irrelevant video segments that are actually not in the database. Visual transformations that may be applied to derive a query video include: simulated camcording, picture in picture, insertions of pattern, change of gamma, strong reencoding, decrease in quality, post production, or mix of three transformations described above.

From the experience in years 2008 and 2009, multimodal queries often achieve better copy detection performance, and thus audio+video queries are adopted in TRECVID 2010. An audio query is also derived from the sound track of a video in the database, by means of acoustic transformations such as bandwidth limitation, subband quantization noise, variable mixing with unrelated audio content.

In this paper, we still focus on video only queries, but specially investigate how to accurately find copy segments by the queries derived from various transformations and mixed with irrelevant content. Settings of TRECVID 2010 content-based copy detection task will be used in experiments.

III. System Overview

Figure 1 shows the flowchart of the proposed video copy detection method. For both videos in the database and the query video, we transform them into an efficient representation, and then conduct approximate matching between two sequences to achieve video copy detection. Videos are first segmented into shots, and appropriate numbers of keyframes are extracted from each video shot according to a dynamic clustering scheme. Starting from each keyframe, trajectories based on distinct features are constructed. Moving directions of trajectories are coded as an orientation histogram that efficiently represents how objects move in each video shot and serves as the novel representation different from conventional color-based or texture-based representation.

Because evolutions of trajectories may fluctuate due to transformations such as shift, cropping, and strong re-encoding, we exploit the bag of word model to describe orientation histograms. This representation is then called *bag of trajectory* (BoT), in which a BoT word conceptually represents a type of trajectory evolution that commonly appears in the training corpus. With this representation, we view videos as documents described by BoT words. Motivated by techniques from natural language processing, we give different weightings to different BoT words based on their local and global statistical information. Finally, we transform each video into a sequence consisting of BoT feature vectors.

To compare a query video with a video in the database, we construct a similarity matrix, in which the (i, j)-th entry indicates the similarity between the ith shot in the query video and the jth shot in the targeted video. Based on this matrix, we find the best match between the query video and a targeted video by a dynamic programming strategy, followed by finding the maximum-sum segment [6] in the matching. As described above, a query video in TRECVID 2010 may arbitrarily contain irrelevant content at the beginning or at the end. We therefore have to determine a locally optimal matching between videos. Given a query video, we compare it with each video in the database, and evaluate the found matched segments. The video copy detection results are finally returned by ranking the matched segments.

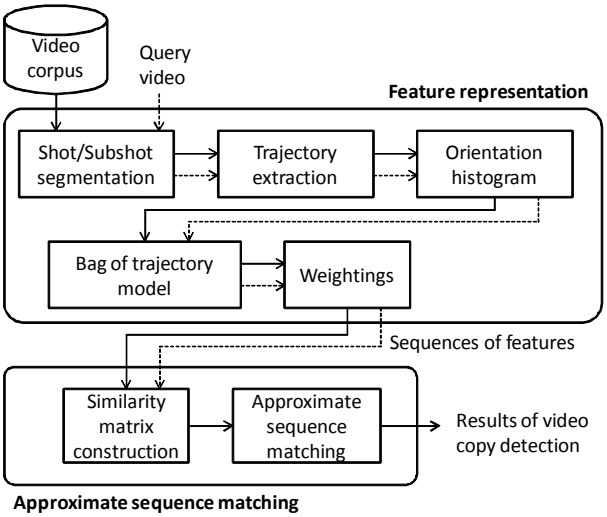

Figure 1. Flowchart of video copy detection.

IV. BOT REPRESENTATION

Based on examining color histogram difference between consecutive frames, we first segment a video into video shots. The HSV (hue, saturation, value) color space is used, and the HSV color histogram for each frame consists of sixteen dimensions, with eight dimensions for hue and four dimensions for saturation and value, respectively.

For each video shot, we would like to further segment it into smaller units, called *subshot*, so that object movement in the same subshot is consistent. To adaptively determine keyframes for each shot, we adopt the global k-means algorithm [7] to cluster videos frames and find the most appropriate number of clusters. The frame that is closest to its clustering centroid is selected as the keyframe. The video segment from a keyframe to its temporally adjacent keyframe forms a subshot. This procedure is illustrated in Figure 2.

We are able to apply the optical flow algorithm to estimate object motion in each subshot. However, high computation cost and tremendous amounts of videos make this approach infeasible. We instead extract distinct feature points at the start of each subshot, and then just track these features to construct motion trajectories. Because the features points are often located on the corners of objects, how they move appropriately describe how objects move.

In this work, we extract the SURF [8] (Speech Up Robust Features) feature points from keyframes, followed by feature tracking with the KLT [9] (Kanade-Lucas-Tomasi) algorithm. SURF features can be efficiently detected and are invariant to scaling, rotation, and some degree of illumination changes and viewpoint changes. Based on these features, time cost for motion estimation is largely reduced.

For a subshot, a large number of trajectories with different lengths (frame number) may be extracted. To efficiently represent a trajectory, we collect statistics of moving directions between two consecutive frames. Moving direction is categorized into five classes and each of which is denoted by a number from 0 to 4: moving toward up-right (denoted by 1), moving toward up-left (denoted by 2), moving toward left-bottom (denoted by 3), moving toward right-bottom (denoted by 4), and no movement (denoted by 0). We calculate the probability of each moving direction and form a 5-dimensional vector to describe a trajectory. For example, if moving directions of a trajectory of four frames are (4, 1, 2, 2), it is transformed as the vector (0:0.0, 1:0.25, 2:0.5, 3:0.0, 4:0.25), in which (*m*:*n*) indicates the probability of moving toward direction *m* is *n*. With this representation, trajectories of various lengths are described in the same way.

Motivated by the bag of word model that is originally proposed in natural language processing, we try to view trajectories as the basic elements to describe videos [5]. We conceptually map a video into a document, and map trajectories into visual words for constituting the document [10]. Given the training corpus, we extract trajectories from each subshot and transform them into 5-dim orientation histograms. Feature vectors collected from the training corpus are then clustered by the k-means algorithm. Feature vectors that are grouped into the same cluster are claimed to represent the same bag of trajectory (BoT) word. A BoT word conceptually represents a set of trajectories that are similar in moving evolution. A video shot d that consists of many trajectories, therefore, is transformed into a BoT word histogram $\boldsymbol{h} = \{n_{1,d}, n_{2,d}, ..., n_{K,d}\}$, in which $n_{i,d}$ denotes the number of trajectories corresponding to the ith BoT word b_i. The value K is the number of different BoT words, i.e. number of clusters.

Different BoT words have different influences on describing documents. From the study of natural language processing, we can measure the importance of a BoT word by TF-IDF (term frequency – inverse document frequency):

$$w_i = \frac{n_{i,d}}{n_d} \log \frac{D}{n_i}, \tag{1}$$

where n_d denotes the number of BoT words (number of trajectories) in the document (video) d, n_i denotes the number of documents that contain b_i, and D denotes the number of document in the training corpus. If b_i occurs frequently in the document d but rarely occurs in other documents, it's a more important BoT word to describe the document d.

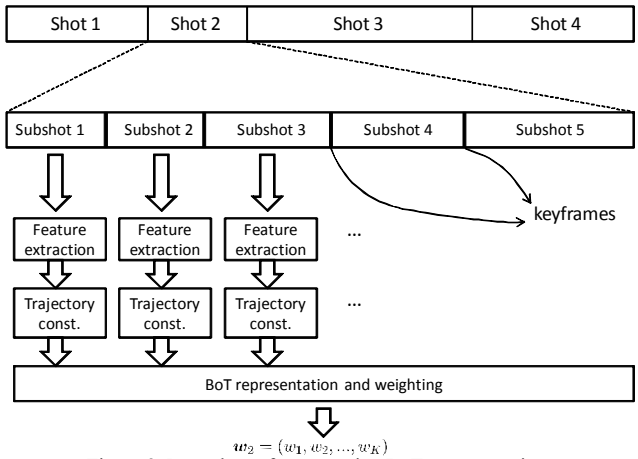

Figure 2. Procedure of constructing BoT representation.

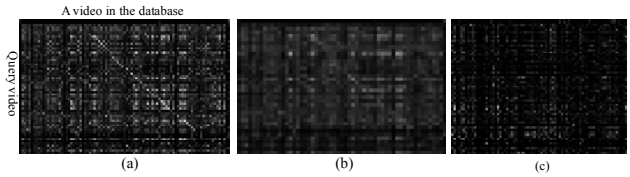

Figure 3. Examples of similarity matrices.

After the processes described above, we transform each video shot as a K-dim vector $\boldsymbol{w} = (w_1, w_2, .., w_K)$, in which w_i denotes the weighting corresponding to the ith BoT word b_i. Figure 2 shows an example of transforming the second video shot into a TF-IDF vector.

V. VIDEO COPY DETECTION

A. Finding Video Copy Candidates

With the procedure illustrated in Figure 2, we represent a video as a sequence of tf-idf feature vectors. Therefore, the problem of video matching has been transformed into comparing sequences of feature vectors. Given a query video in the representation of $\boldsymbol{Q} = (\boldsymbol{w}_1^q, \boldsymbol{w}_2^q, ..., \boldsymbol{w}_M^q)$ that consists of M shots, we would like to compare it with a video in the database with the representation of $\boldsymbol{T} = (\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, ..., \boldsymbol{w}_N^t)$. We assume that the number of shots in the video in corpus is always larger than that in the query video, i.e. $N > M$. By comparing any two video shots in $\boldsymbol{Q}$ and $\boldsymbol{T}$, we can construct an $M \times N$ similarity matrix S, in which the (i, j)-th entry is defined as

$$S(i, j) = sim(\boldsymbol{w}_i^q, \boldsymbol{w}_j^t), \tag{2}$$

where $sim()$ denotes the cosine similarity between two vectors.

Figure 3 shows examples of similarity matrices between two videos, in which pixels with higher intensity mean larger similarity values between two shots. The matrix in Figure 3(a) is constructed by comparing a query video with the video that is really a video copy to the query. We can see a diagonally straight line in this matrix. Figure 3(b) shows a matrix constructed by comparing a query video with the video that is really a video copy, but the query video is suffered from gamma change. Although we still can roughly recognize the diagonally straight line in this matrix, finding correspondence between these two videos is much more challenging than that in Figure 3(a). Figure 3(c) shows the matrix constructed by comparing a query video with the video that doesn't contain video copy at all. Noise-like patterns can be seen in this case.

As we describe in Section 2, queries in the TRECVID 2010 content-based copy detection task may contain irrelevant video segments at the beginning or at the end. Therefore, we have to be able to find partial matching between the query and any video in the database. Based on the similarity matrix, the wanted partial matching corresponds to a block that has the maximum sum of intensity values. A straightforward method to solve this problem is to view the similarity matrix as an image, and adopt an image segmentation algorithm such as watershed [5] to find the partial matching. However, due to visual transformation or interference caused by noise, the blocking effect is often not clear, as shown in Figure 3(b).

To improve reliability of video copy detection, we propose an approach that first finds globally optimal matching between two videos, and then localize the best matching in the found candidates. Based on the similarity matrix constructed from two sequences of feature vectors, we can formulate approximate sequence matching as finding the longest common subsequence (LCS) between them. The longest common subsequence between two subsequences $\boldsymbol{Q}_m$ and $\boldsymbol{T}_n$ is described as follows.

$$LCS(\boldsymbol{Q}_m, \boldsymbol{T}_n) = \begin{cases} LCS(\boldsymbol{Q}_{m-1}, \boldsymbol{T}_{n-1}) + sim(\boldsymbol{w}_m^q, \boldsymbol{w}_n^t), & \text{if } \boldsymbol{w}_m^q \sim \boldsymbol{w}_n^t, \\ \max(LCS(\boldsymbol{Q}_{m-1}, \boldsymbol{T}_n), LCS(\boldsymbol{Q}_m, \boldsymbol{T}_{n-1})), & \text{otherwise,} \end{cases} \tag{3}$$

where $\boldsymbol{Q}_i$ denotes the i th prefix of $\boldsymbol{Q}$, i.e. $\boldsymbol{Q}_i = (\boldsymbol{w}_1^q \boldsymbol{w}_2^q ... \boldsymbol{w}_i^q)$. The notation $LCS(\boldsymbol{Q}_i, \boldsymbol{T}_j)$ denotes sum of similarity of the longest common subsequence between $\boldsymbol{Q}_i$ and $\boldsymbol{T}_j$. The notation $\boldsymbol{w}_m^q \sim \boldsymbol{w}_n^t$ means that the similarity value between the mth shot of the query video and the nth shot of the targeted video is larger than a threshold. This recursive structure facilitates usage of the dynamic programming strategy to find the global optimal solution.

In the conventional LCS algorithm, the LCS between $\boldsymbol{Q}_M$ and $\boldsymbol{T}_N$ is found by backtracking from the most right-bottom entry of the similarity matrix $S(M, N)$. Note that $LCS(\boldsymbol{Q}_M, \boldsymbol{T}_N)$ represents the globally optimal matching between two sequences. However, we know that the query video may be partially copied from a video in the database, and thus finding globally optimal matching between two sequences that are actually partially overlapped would lead to misleading situations. In this work, we find all possible global matchings by backtracking from $S(M, N)$, $S(M, N-1)$, ..., $S(M, M+1)$, respectively. By each backtracking, we are able to find a path that indicates a possible correspondence between the query and the video in database. All these paths are viewed as candidates that may consist of video copy segments. According to our experiments, the real video copy segment would be embedded in at least one of these possible matchings. In the next subsection, we develop a process to examine these candidates, and find the locally optimal matching that indicates the real video copy segments.

B. Localize Video Copy Segments

For a path $\boldsymbol{\xi} = (\xi_1, \xi_2, ..., \xi_M)$ determined by the LCS algorithm, we would like to determine a segment in it that conveys the optimal local matching between videos. Note that the length of this path is always equal to the length of the query video due to $N > M$, and any entry ξ_i in this path indicates an entry in the similarity matrix, i.e. the similarity value between two video shots. This problem can be formulated a variation of the maximum-sum segment problem [6]. The goal is to find a segment $\boldsymbol{\xi}(i, j) = (\xi_i, \xi_{i+1}, ..., \xi_j)$ from $\boldsymbol{\xi}$ such that the segment $\boldsymbol{\xi}(i, j)$ of the longest length conveys the largest average

similarity value, where $i = 1, ..., M-1$, $j = 2, ..., M$, and $i < j$.

To find the segment $\boldsymbol{\xi}(i, j)$ corresponding to the video copy segment, we first transform the sequence $\boldsymbol{\xi} = (\xi_1, \xi_2, ..., \xi_M)$ into a real number sequence $\boldsymbol{Z} = (z_1, z_2, ...z_M)$ as follows. The mean similarity of this path is calculated:

$$\rho = \frac{1}{M} \sum_{i=1}^{M} \xi_i. \quad (4)$$

After mean removing, we obtain

$$z_i = \xi_i - \rho. \quad (5)$$

Note that the sequence $\boldsymbol{Z}$ may contain both negative and positive real numbers.

We would like to find an interval $[i, j]$ in $\boldsymbol{Z}$, $1 \le i \le j \le M$, such that $\boldsymbol{Z}(i, j) = (z_i, ..., z_j)$ is the maximum-sum segment of $\boldsymbol{Z}$, i.e. $\sum_{h=i}^{j} z_h$ is maximal in all possible substrings in $\boldsymbol{Z}$.

The aforementioned problem can be viewed as a range maximum-sum segment query (RMSQ) problem [6], which is able to be solved by a linear time algorithm. In this work, we apply the algorithm proposed by Chen and Chao [6] to find the segment in $\boldsymbol{Z}$, which conceptually indicates the most similar segment between the query video and the video in database, along the current LCS (the current matching situation).

With the processes above, we can find a maximum-sum segment for each possible matching (each LCS). Assume that the maximum-sum segments $\boldsymbol{Z}_1(i_1, j_1)$, $\boldsymbol{Z}_2(i_2, j_2)$, …, $\boldsymbol{Z}_k(i_k, j_k)$ are respectively found from the sequence matchings backtracking from $S(M, N)$, $S(M, N-1)$, …, $S(M, M+1)$. We determine the best local matching between the query video and the video in database by finding the maximum-sum segment $Z_{\ell^*}(i_{\ell^*}, j_{\ell^*})$ that has the largest average similarity μ_{ℓ^*}:

$$\ell^* = \arg\max_{\ell=1,2,...,k} \mu_\ell, \quad (6)$$

$$\mu_\ell = \frac{1}{j_\ell - i_\ell}(z_{i_\ell} + z_{i_\ell+1} + ... + z_{j_\ell}). \quad (7)$$

With this decision, we finally find the optimal local matching between the query video and a video in the database. We respectively find best local matching between the query video and all videos in database, and then rank the retrieval results by average similarity values of corresponding maximum-sum segments.

Figure 4 shows the overall scheme for video copy detection. Given the query video that would consist of only a segment of video copy, we first compare it with every video in the database and respectively construct a similarity matrix. Based on a similarity matrix, we find all possible matchings between two videos, and then find the maximum-sum segment in each matching. The best local matching corresponds to the maximum-sum segment that has the largest average similarity value. After obtaining the best local matchings between the query video and all videos in database, the retrieval results are ranked according to average similarity values.

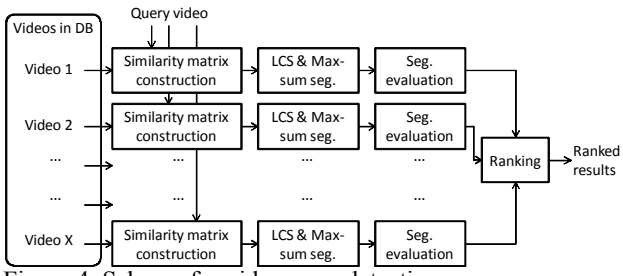

Figure 4. Scheme for video copy detection.

Figure 5. Samples of query videos with different transformations.

VI. Performance Evaluation

To evaluate the proposed video copy detection method, we generate query videos according to the transformations defined in TRECVID 2010, including change of gamma, insertions of pattern, picture in picture, cropping, and shift. We randomly select nine videos from the TRECVID 2010 database, randomly select a segment from each video, followed by applying a transformation on it. In this work, Corel Digital Studio [11] is used to implement these transformations. At the beginning and the end of the transformed video segment, we concatenate it with irrelevant video segments (not in the TRECVID dataset) of arbitrary lengths. By respectively applying five transformations to nine videos, we finally generate 45 query videos that have partial video copies. Figure 5 gives examples of query videos generated from the same video but with different transformations.

There are totally 3173 videos in the database. Length of each video ranges from 3.6 minutes to 4.1 minutes, and totally more than 200 hours of videos are in the database.

A. Performance Variations with Different Numbers of BoT Words

We evaluate the influence of the number of BoT words on detection accuracy. Recall that a BoT word represents a kind of trajectory. In this experiment, we respectively evaluate detection performance based on feature vectors derived from 50, 100, 150, 200, and 250 BoT words, with the 45 query videos. Figure 6 shows the performance variations. The vertical axis shows the percentage of queries that successfully retrieve the correct video copy in the top k

retrieved results. From Figure 6, we clearly see that representing videos with the dictionary consisting of 100 BoT words achieves the best performance, and thus we use this setting in the following experiments. With this setting, about 70% of queries can successfully retrieve the correct video copy in the top 3 results.

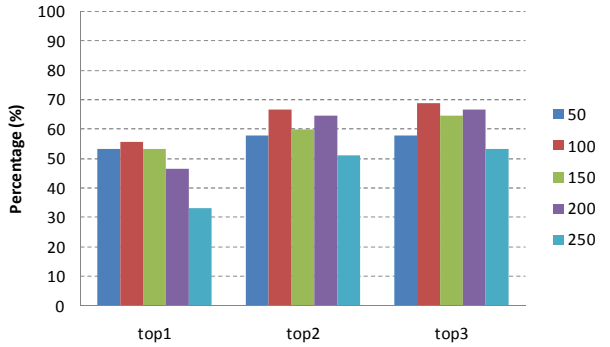

Figure 6. Performance variations with different numbers of BoT words in different ranking situations.

B. *Performance Variations for Different Types of Queries*

Different transformations may differently deteriorate the representation of BoT. Figure 7 shows the percentage of queries with different transformations that retrieve correct results in the top 1 and top 2 results. We obtain the best performance in cropping. Cropping in this work is implemented as removing the surrounding region but remaining the center region of video frames. Because there are fewer feature points (SURF features) in background, removing surrounding region just slightly degrades detection performance.

On the other hand, detection by queries with shift achieves the worst performance. We implement shift transformation by horizontally or vertically shifting the query video. Because the contents in "shift out" region and "shift in" region are missing, we scale the frame as the original size (see Figure 5). The detected features would be different or the locations of features would change, which further change constructed trajectories and degrade detection performance.

Overall, Figure 7 shows that the detection performance is satisfactory by queries with different transformations, especially when we consider the top 2 retrieved results.

C. *Performance of Copy Segment Detection*

In previous two experiments, we claim that it's a correct detection if the retrieved result really contains the video copy segment. In this subsection, we further calculate precision and recall to show how accurately we can achieve to find the video copy segment.

$$\text{precision} = \frac{N_c}{N_r} \text{ and } \text{recall} = \frac{N_c}{N_v}, \quad (8)$$

where N_r denotes the number of frames of the retrieved video copy segments, N_c denotes the number of frames that are in the retrieved results and are really in the truth video copy segment, and N_v denotes the number of frames that are in the truth video copy segment. Precision and recall for nine queries derived from the same transformation are averaged, respectively.

Figure 8 shows average precision and recall of different query types. We compare our method with the one proposed in [5], in which the watershed algorithm is used to find locally optimal matching between the query video and the videos in database. Although the results reported in [5] are promising, we found that the watershed method is not robust to transformations defined in TRECVID 2010. When we apply queries with transformations, correspondence embedded in similarity matrices is not clear, and thus the watershed method that is originally designed to segment images with clear object boundaries doesn't work well. Overall, our method achieves 0.79 in precision and 0.80 in recall, while the method in [5] achieves 0.23 in precision and 0.55 in recall.

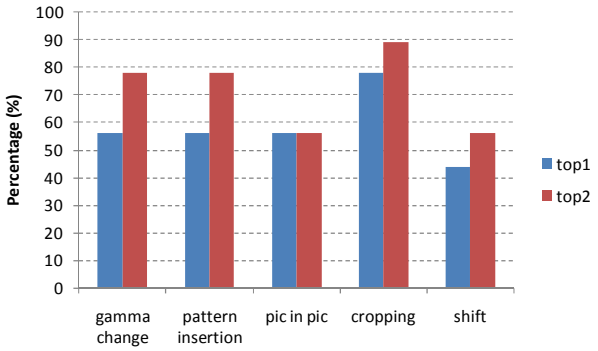

Figure 7. Detection accuracies by queries derived from different transformations.

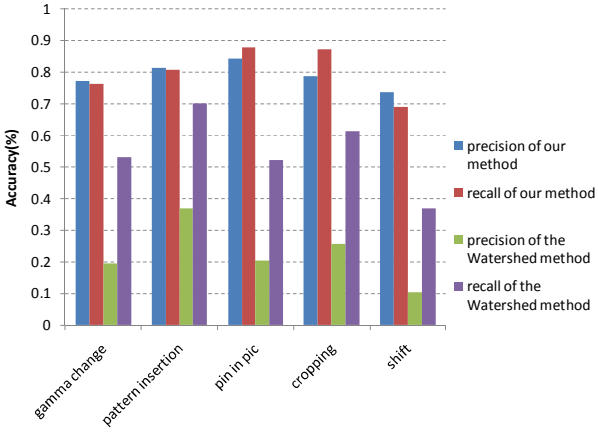

Figure 8. Accuracies of copy segment detection by queries with different transformations.

VII. CONCLUSION

We present a video copy detection system that represents videos by a bag of word model and conduct detection by approximate sequence matching. After extracting feature-based trajectories in videos, we view each video shot as a document constituted by bag of trajectory words. This

representation effectively describes the information of object movement. We compare videos based on this representation, and transform video copy detection as an approximate sequence matching problem. In addition to finding the longest common subsequence between two sequences, we further find the locally optimal matching by the maximum-sum segment algorithm. Different types of queries are evaluated in experiments, based on the TRECVID 2010 benchmark, and the experimental results demonstrate the effectiveness and superiority of the proposed method.

In the future, queries with more visual transformations will be studied, and audio information will be considered as well to conduct multimodal video copy detection.

ACKNOWLEDGMENT

This work was partially supported by the National Science Council of ROC under NSC 98-2221-E-194-056.

REFERENCES

[1] YouTube, http://www.youtube.com/

[2] 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," Proceedings of ACM International Conference on Image and Video Retrieval, pp. 371-378, 2007.

[3] TREC Video Retrieval Evaluation, http://trecvid.nist.gov/

[4] D. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[5] X. Wu, Y. Zhang, Y. Wu, J. Guo, and J. Li, "Invariant visual patterns for video copy detection," Proceedings of International Conference on Pattern Recognition, 2008.

[6] K.-Y. Chen and K.-M. Chao. On the range maximum-sum segment query problem. Discrete Applied Mathematics, vol. 155, no. 16, pp. 2043-2052, 2007.

[7] A. Likas, N. Vlassis, and J.J. Verbeek, "The global k-means clustering algorithm," Pattern Recognition, vol. 36, pp. 451-461, 2003.

[8] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, "SURF: speeded up robust features," Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346-359, 2008.

[9] J. Shi and C. Tomasi, "Good features to track," Proceedings of IEEE Conference on Computer Vision and Pattern Recongition, pp. 593-600, 1994.

[10] J. Sivic and A. Zisserman, "Efficient video search for objects in videos," Proceedings of the IEEE, vol. 96, no. 4, pp. 548-566, 2008.

[11] Corel Corporation, http://www.corel.com

[12] J. Law-To, A. Joly, and N. Boujemaa. "Muscle-VCD-2007: a live benchmark for video copy detection," http://www-rocq.inria.fr/imedia/civr-bench, 2007.

[13] C.-Y. Chiu, C.-S. Chen, and L.-F. Chien, "A framework for handling spatiotemporal variations in video copy detection," IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 3, pp. 412-417, 2008.

[14] M.-C. Yeh and K.-T. Cheng, "Video copy detection by fast sequence matching," Proceedings of ACM International Conference on Image and Video Retrieval, 2009.

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