Rushes Video Summarization by Object and Event Understanding

Feng Wang
Dept. of Computer Science
City University of Hong Kong
fwang@cs.cityu.edu.hk

Chong-Wah Ngo
Dept. of Computer Science
City University of Hong Kong
cwngo@cs.cityu.edu.hk

ABSTRACT

This paper explores a variety of visual and audio analysis techniques in selecting the most representative video clips for rushes summarization at TRECVID 2007. These techniques include object detection, camera motion estimation, keypoint matching and tracking, audio classification and speech recognition. Our system is composed of two major steps. First, based on video structuring, we filter undesirable shots and minimize the inter-shot redundancy by repetitive shot detection. Second, a representability measure is proposed to model the presence of objects and four audio-visual events: motion activity of objects, camera motion, scene changes, and speech content, in a video clip. The video clips with the highest representability scores are selected for summarization. The evaluation at TRECVID shows that our experimental results are highly encouraging, where we rank first in EA (easy to understand), second in RE (little redundancy) and third in IN (inclusion of objects and events).

Categories and Subject Descriptors

I.2.10 [**Computing Methodologies**]: Artificial Intelligence-Vision and Scene Understanding; J.m [**Computer Applications**]: Miscellaneous

General Terms

Algorithms, Design, Performance

Keywords

Rushes Video Summarization, Object Detection, Event Understanding

1. INTRODUCTION

The aim of rushes summarization at TRECVID 2007 is to generate short summaries of raw videos from movie production. The summaries should include as many objects and events as possible so that the professional editors can quickly browse the content and decide their utilities.

TVS'07, September 28, 2007, Augsburg, Bavaria, Germany.

The way of video summarization is to produce a simplified version of the given video by reducing redundant information and the content that can be easily predicted by watching just a portion of the video. Compared with the final version of movie product, there are two types of undesirable or redundant information in rushes videos. The first one is the undesirable content captured by the camera together with the play or inserted during video recording, such as some preparation work for movie capture, discussions between the actors, the director and the cameraman, color bars, and black or grey screen shots. This kind of information is not really related to the storytelling of the movie, and not of value for the editors. Including these shots not only causes the summary lengthier than expected, but also reduces its comprehensiveness. Prior to summary generation, an essential step is to detect and filter these less useful shots. The second type of redundant content is the repetitions of each shot. During movie production, the same scene is usually taken for many times due to some unexpected errors, *e.g.* an actor forgets his lines, or someone else passes by before the camera. Usually, twenty to forty times as much material may be shot as actually becomes part of the finished product. In this case, all the retake shots need to be detected. Only the most satisfactory one is kept for video summarization while all the other repetitions are removed.

After filtering the undesirable and redundant shots, we summarize the video by reducing the intra-shot redundancy of the remaining shots. Video summarization has been studied for decades and different kinds of approaches are proposed based on frame clustering [1], speech transcript [11], and multiple information streams [3]. Video summarization is a challenging task due to the requirement of making decisions automatically according to the semantics of the given video. The evaluation is also a problem as the quantitative measure of summary quality is not easily derived. In the TRECVID guidelines for rushes summarization task, several criterions are used for summary evaluation, including the fraction of the objects and events included by the summary (**IN**), ease of video understanding (**EA**), time needed for subjective judgment (**TT, VT**), and compactness of the summary (**DU, XD**). (Please refer to Table 1 for detailed explanations.)

Our approach focuses on the understanding of video content, including objects and events. A similar piece of work is in [3]. In order to capture the objects and people in the summary, we detect and track objects and human faces in each shot. Both visual and audio events are considered. Visual events include motion activities of objects, camera motion

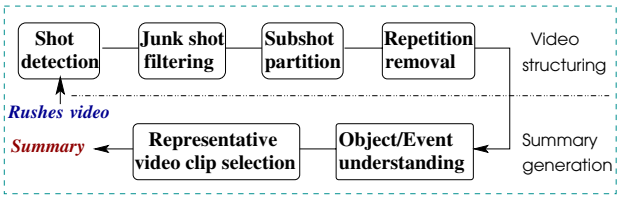

Figure 1: Our framework for rushes summarization.

and scene changes, while audio events include the dialogue or speech of people. All these features are combined to calculate a representability score for each video clip along the timeline. The most representative clips are selected to generate a video summary. Compared with the statistical methods such as frame clustering [3], our approach can get more semantic knowledge of the threading and storytelling of the videos by feature extraction, and make better decisions for video summarization.

Figure 1 illustrates our framework for rushes video summarization. Our approach is composed of two steps. First, a given rushes video is structured by shot boundary detection and subshot partition (Section 2). Based on video structuring, we filter the junk shots and reduce the inter-shot redundancy by repetition removal (Section 3). Second, different visual and audio features are extracted for object and event understanding. A representability measure is proposed to model these features for summary generation (Section 4).

2. VIDEO STRUCTURING

Given the unstructured raw data of rushes video, the first step of our approach is video structuring. Based on shot boundary detection, we select keyframes from each shot and then semantically partition each shot into subshots.

2.1 Shot Detection and Keyframe Selection

We employ the shot detection algorithm in [4] to partition a rushes video into shots. This algorithm detects different shot transitions based on the analysis of spatio-temporal slices extracted from compressed domain. Since all rushes videos are unedited, there are only *cut* transitions. In our experiments, most shots can be detected successfully and efficiently.

An algorithm of keyframe selection for video summarization is proposed in [2]. Three different features including color histogram, edge histogram and multiresolution wavelet analysis are employed to measure the difference between neighboring frames. In our approach, in addition to global color feature, we split each frame into 5×7 grids and calculate the mean and variance of YUV colors in each grid. Euclidean distance is used to measure the difference between neighboring frames. The grid-based approach can capture local motion activities and visual changes. High curvature points are then detected within the curve of the cumulative frame difference, and the keyframes are selected at the midpoints between two consecutive high curvature points.

2.2 Junk Shot Filtering

Some shots in rushes video are not useful for editors, such as color bars, black or gray screens, and very short shots. Before further video analysis, we detect and filter these junk shots.

The short shots with less than 10 frames can be easily filtered. To detect the color bars and black or gray screens, we employ an algorithm similar to [10] by comparing the color histograms of the keyframes with the example color bar and gray screen frames.

2.3 Subshot Partition

In rushes videos, each shot usually contains not only the movie play, but also some other materials that are not related to the storytelling, such as camera adjustment and scene arrangement before movie shooting, discussions between the director and the actors, and unintentional camera motion. In video summarization, we intend to include only the video segments of movie play and remove all the other elements. For this purpose, we partition each shot into subshots corresponding to different phases during video capture. Several features are employed to detect story-irrelevant subshots and possible breakpoints for subshot partition, including unintentional camera motion, retake scene (see examples in Figure 2), the director's commands, and audio scene changes.

We employ the algorithms in our previous work [5, 6] for rushes video structuring based on camera motion estimation. Three semantic categories are considered, namely *stock*, *outtake* and *shaky*. The concept *stock* represents the clips with intentional camera motion which have the potential for reuse, such as capturing an event with still camera and rotating the camera for a panoramic view. In contrast, those clips with intermediate camera motion, which are very likely to be discarded in the final production, are denoted as *outtake*. Besides these two extreme cases, the category *shaky* represents the shaky artifacts which may be discarded or used for special effects such as to show an emergent situation. The hierarchical hidden markov model proposed in [6] is used to classify these three categories and partition a shot into subshots correspondingly.

The second feature is the retake scene, where someone shows a card indicating the sequence of the current shot before the camera. Although correctly recognizing the text and shot numbers is difficult most time, this kind of scenes indicate the starting of a new take or shot, and can be used as breakpoints for subshot partition. We employ the algorithm for Near-Duplicate Keyframe (NDK) detection in [7, 12] to detect retake scenes. A set of 50 example keyframes of the retake scenes are extracted from the development set as shown in Figure 2. The regions of the cards are manually annotated. Among the keyframes of each shot in the given rushes video, we detect the keypoints and match them with the example retake scenes. Figure 2 shows some matching lines between keyframes and the matched example retake scenes. If enough matching lines are found that lie in the annotated regions, the keyframe is detected as a retake scene and the shot is partitioned at that point.

In most cases, the director controls the progress of movie capture by calling out keywords such as "standby", "action", "cut", "take xx", and "shot yy" (xx, yy are the sequence number of the current take and shot). These keywords can be used as indications of the starting and ending of movie play. We employ an ASR (Automatic Speech Recognition) engine for speech recognition and then detect these keywords in the output transcripts, which are used as breakpoints between subshots. A pair of "action" and "cut" consists of the movie play. If there are more than one such pairs, the video clip marked by the last pair is selected and all the other parts are ignored by supposing the last take is the most satisfactory

Figure 2: Detection of retake scenes. First row: keyframes from test video set; Second row: example retake scenes extracted from development set.

one. From our experiments, this assumption works most time. Since the director usually calls out these keywords aloud, the speech recognition rate is pretty high. In case there is no such keyword detected, the whole shot will be used for further processing.

The last feature is audio. In some cases, although the camera is faced to the actors, but they might be discussing with the directors instead of playing. These kinds of subshots cannot be detected by only visual cues. However, we can find that in audio track there is quite obvious boundary between movie play and unintentional subshots, since the source of audio, manner of speaking and background noise are different in these two scenes. The audio scene changes can thus serve as breakpoints for subshot partition. We extract features from audio track and classify the corresponding segments into three classes: silence, actor's lines, and direction or talks from the director or other people. The audio signal is segmented into frames. Each frame is 50 milliseconds (ms) long and overlaps with the previous one with 25 ms. In training stage, we manually select and annotate different video clips from development video set. Different features are extracted from audio track, including cepstral-flux, multi-channel cochlear decomposition, cepstral vector, low energy fraction, volume standard deviation, non-silence ratio, standard deviations of pitch and zero crossing rate, and smooth pitch ratio [9]. An SVM is employed for classification. The neighboring frames of the same category are then merged into subshots.

3. REPETITIVE SUBSHOT REMOVAL

The resulting subshots consist of the basic story units for video summarization. However, besides the actual movie play, there are some other kinds of subshots which are less useful for editors, such as retake scenes, unintentional camera motion, and discussions between actors and the director. Furthermore, due to the mistakes of the actors and in order to achieve better effects, each shot is usually taken many times, which results in many repetitive subshots in rushes videos. Before summary generation, we detect and remove the redundant and less useful subshots in the videos.

We begin with matching subshots in different shots. The similarity of two subshots are calculated based on keyframe and speech transcript comparison. The keyframe similarity is measured by using color histogram, edge histogram, multiresolution wavelet analysis and color moment. The

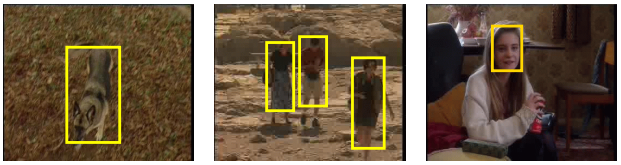

Figure 3: Object and face detection.

transcript similarity is measured based on string matching. Given two subshots s_i and s_j, two different cases are considered: i) s_i and s_j are repetitions of each other; ii) s_i is a subshot of s_j. For the second case, s_i is an incomplete version of s_j and removed from the subshot list. For the first case, all shots are complete. They are different takes of the same shot until the director gets an satisfactory one. In this case, we choose the last version and remove all the other repetitions.

By subshot matching, we construct a directed graph $G(S, E)$ with vertex set S and edge set E. S contains all subshots in different shots. There is no edge between any two subshots in the same shot. Given two subshots s_i, s_j from two different shots, a directed edge $s_i \rightarrow s_j$ is added if s_j is a repetition of s_i. The repetitive subshot detection is solved by searching for maximum complete subgraph. The larger the detected complete subgraph is, the more confident we are that the corresponding subshots are repetitions of an intentional one. For all repetitive subshots, the last one is selected and all the others are removed. Note that we remove the less useful subshots here instead of in Section 2.3 according to the classification results. This avoids some useful subshots being falsely classified and deleted. The usefulness of a subshot can be supported by its repetitive versions.

4. VIDEO SUMMARIZATION

By junk shot filtering and repetitive subshot detection, we remove the less useful information in the video and reduce inter-shot redundancy. Most of the remaining subshots are movie playing parts which the editors would concern. To help the editors quickly browse the content of the video, a short summary is generated which should be concise (the summary duration is less than 4% of the original video), compact (there is little redundant information), and comprehensive (as many objects and events as possible should be included). In this section, we extract visual and audio features from the video, and produce a summary by object and event understanding.

4.1 Feature Extraction

To capture the objects in videos, we employ the algorithm in our previous work [8] for object detection and tracking. Some examples are shown in Figure 3. Each object is represented as $o(o_{id}, t_s, t_e, p_o(t))$, where o_{id} is the identity of the object, t_s and t_e are the time when the object appears and vanishes before the camera, and $p_o(t)$ includes a list of locations of the object at time t ($t_s \leq t \leq t_e$).

We consider visual and audio events in the videos. Visual events include motion activities of objects, camera motion and changes of scenes. By object detection and tracking, we get the location of each object at any given moment. To further estimate the movement of objects and changes of scenes, we employ the algorithm in [12, 7] to track the key-

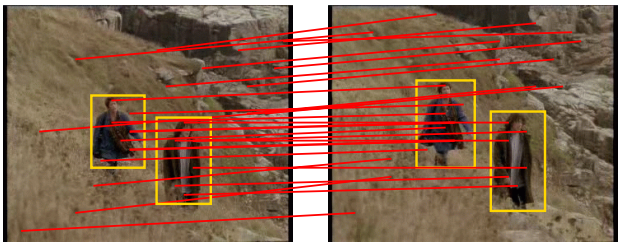

Figure 4: Keypoint tracking in sampled frames.

points in sampled frames. By combining the results of object detection, keypoints are classified as belonging to objects or background scenes.

Basically we evenly select 3 frames every second. For each sampled frame, keypoints are detected and matched with the subsequent 5 frames by employing the algorithm in [12, 7]. An example of keypoint tracking is shown in Figure 4. Based on the results of object detection, a matching keypoint in two frames is assigned to an object if it lies on the object tracked in both frames. If it does not lie on any one of the object, it is assigned to be a background point. Given a set of keypoints $P = \{p_1, p_2, \ldots, p_m\}$ that belong to an object or the background, the movement of the object or the background between two frames is calculated as the average movement of all the keypoints in P.

For audio events, we consider people's speech and dialogue based on speech recognition. An audio event is detected if the density of meaningful words is high enough in a video clip. Audio event is useful for video summarization since there is usually little visual changes during a long dialogue or speech. Such events are important for the semantic completeness of the summary, but cannot be captured by visual detector.

By feature extraction, we get:

a) A set of objects $O = \{o_i\}$. Each object is associated with the existence period and location information;

b) A set of object motion activities $\Phi = \{\phi_j\}$. Each element is associated with an object and its movement along video sequence;

c) A list of camera motion $\Gamma = \{\gamma_k\}$. Each element is associated with the camera motion parameters.

d) The scene changes $\Delta = \{\delta_l\}$ between neighboring frames.

e) Dialogue or speech clips $\Omega = \{\omega_m\}$. Each clip is associated with the speech transcripts.

4.2 Summarization

Each subshot is segmented into 1-second video clips. Each clip overlaps with the previous one by 300ms. Based on extracted features, we select a minimum set of video clips $V = \{v_1, v_2, \ldots, v_d\}$ that are most representative.

Given a video clip v, five scores are defined to measure the representability of v for the five feature sets $(O, \Phi, \Gamma, \Delta, \Omega)$ respectively as follows:

$$R_v(O) = \sum_{o \in O} \int_{t_1}^{t_2} (1 - \frac{|t - (t_{so} + t_{eo})/2|}{t_{eo} - t_{so}}) dt$$

$$R_v(\Phi) = \frac{\sum_{\phi \in \Phi} \int_{t_1}^{t_2} f(t) dt}{\sum_{\phi \in \Phi} \int_{t_{s\phi}}^{t_{e\phi}} f(t) dt}$$

$$R_v(\Gamma) = \sum_{\gamma \in \Gamma} \int_{t_1}^{t_2} (1 - \frac{|t - (t_{s\gamma} + t_{e\gamma})/2|}{t_{e\gamma} - t_{s\gamma}}) dt$$

$$R_v(\Delta) = \frac{\int_{t_1}^{t_2} \delta(t) dt}{\int_{t_{s\delta}}^{t_{e\delta}} \delta(t) dt}$$

$$R_v(\Omega) = \frac{\|W(v) \bigcap W(\Omega)\|}{\|W(\Omega)\|}$$

where (t_1, t_2) denotes the temporal intersection of v and the corresponding object or event, (t_{sa}, t_{ea}) denotes the existence period of object or event a,$f(\cdot)$ is the motion intensity function, $W(\cdot)$ is the set of words in speech transcript, and $\|\cdot\|$ denotes the set cardinality.

First, we calculate the inclusion of each video clip v_i as $Inc(v_i) =< O_{v_i}, \Phi_{v_i}, \Gamma_{v_i}, \Delta_{v_i}, \Omega_{v_i} >$, where $O_{v_i} \subseteq O$,$\Phi_{v_i} \subseteq \Phi$, $\Gamma_{v_i} \subseteq \Gamma$, $\Delta_{v_i} \subseteq \Delta$,$\Omega_{v_i} \subseteq \Omega$ are the sets of objects, motion activities, camera motion, scene changes, and people talking clips respectively that lie in v_i. A representability score of a video clip v_i for v_j is defined as

$$\begin{aligned} Rep(v_i, v_j) &= \frac{1}{\sqrt[4]{d(v_i, v_j)}} \cdot (w_O R_{v_i}(O_{v_j}) + w_\Phi R_{v_i}(\Phi_{v_j}) \\ &+ w_\Gamma R_{v_i}(\Gamma_{v_j}) + w_\Delta R_{v_i}(\Delta_{v_j}) + w_\Omega R_{v_i}(\Omega_{v_j})) \end{aligned}$$

where $d(v_i, v_j)$ is the temporal distance between the midpoints of v_i and v_j, $w_O = w_\Phi = w_\Gamma = w_\Delta = w_\Omega = 0.2$ are the weights for the five different features respectively. $Rep(v_i, v_j)$ indicates to what extent v_i can represent v_j. For each video clip, its representability scores for the neighboring clips (at most 150 nearest clips are considered when the score is above a threshold) are calculated. Figure 5 shows the representability curves of two video clips along the timeline. The overall representability of a video clip is calculated as the area below the corresponding curve in Figure 5.

For video summarization, we select a set of video clips Sum that include all detected features and have the highest ratio of representability and total duration. The algorithm works as follows.

1) Initialize $Sum = \{\}$, and C is the set of all video clips ordered by the representability.

2) Select the clip $c_i \in C$ with the highest representability score. $Sum = Sum \bigcup \{c_i\}$.

3) Remove all clips $c_p \in C$ if $Rep(c_i, c_p)$ is larger than a threshold.

4) Goto 2) if C is not empty.

With the above greedy algorithm, the result is usually not optimal. Few features may also be missed. Next, we update Sum to improve the overall representability. For each video clip c_r that is not selected, we attempt two operations: inserting c_r to Sum or using c_r to replace one of its neighboring clip in Sum. The clips that can improve the representability the most are inserted or used to replace another one until the ratio of representability score and total summary duration reaches a defined minimum value. All video clips in Sum are combined to generate a video summary. Although the result is sub-optimal, in our experiments, most detected features can be included successfully.

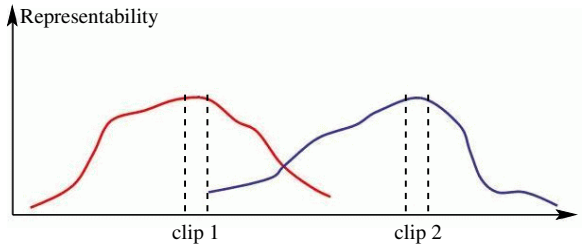

Figure 5: Representability scores of two video clips.

5. RESULTS

Table 1 shows the evaluation results of TRECVID 2007, we get encouraging results for the criterions IN (rank 1), EA (rank 2) and RE (rank 3). Meanwhile, compared with other teams and the baselines, our algorithm can better balance the conciseness and semantic enrichment of the summary. By employing object detection and tracking, our algorithm is good at describing object movement in most cases. However, the summary result is still not satisfactory for some videos, such as "MRS042548.mpg" with many fast camera motion and abrupt scene changes, which results in many short shots.

Feature extraction stage, especially object and face detection, keypoint matching and tracking, consume most of the computational time. The system time is about three to four times of the original video duration.

Table 1: Our results on TRECVID 2007 rushes summarization task.
EA - Easy to understand: 1 strongly disagree - 5 strongly agree;
RE - Little duplicate video: 1 strongly disagree - 5 strong agree;
IN - Fraction of inclusions found in the summary (0 - 1);
DU - Duration of the summary (sec);
XD - Difference between target and actual summary size (sec);
TT - Total time spent judging the inclusions (sec);
VT - Video play time (vs. pause) to judge the inclusions (sec).

Criterion	EA	RE	IN	DU	XD	TT	VT
Baseline 1	3.44	3.52	0.59	62.11	-2.25	105.66	60.88
Baseline 2	3.41	3.59	0.58	60.84	-0.97	100.48	58.68
Mean of all 22 teams	3.16	3.66	0.48	49.54	10.33	92.27	51.14
Our result	**3.60**	**3.94**	**0.64**	**41.11**	**18.75**	**90.58**	**44.83**
Our Ranking	1	2	3	5	5	9	6

6. DISCUSSION AND CONCLUSIONS

Compared with home videos, rushes videos are captured by professional cameramen for movie production. On the other hand, as an unedited version, they are intertwined with many junk shots and intermediate camera motion. The structure of the video and threading of the story are not directly available. However, the mixture of these two elements (home video and movie product), suggests a way of partitioning and classifying rushes video segments. In this work, we employ camera motion, visual feature, speech and audio for video structuring and inter-shot redundancy removal.

To summarize the video content, we explore a variety of visual and audio features for object detection and event understanding to depict the storytelling of a given video. Based on these features, we propose a representability measure for each video clip and select the most representative ones for summary generation. Compared with statistical methods, the extracted features can provide more semantic information, especially the thread of the story, which help producing comprehensive summaries.

From the results of all submissions, we observe that only 68% of all objects and events at most can be included. The summarization problem remains challenging due to the difficulty of video understanding. Although many efforts have been made, most submissions cannot significantly outperform the two baselines, which are simply based on fixed-length shot selection and visual clustering. From our experience, object and event detection shows certain strength for semantic-based video summarization. However, there is plenty of space for improvement, and particularly, event understanding is still an open problem.

Acknowledgement

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (CityU 118905).

7. REFERENCES

[1] A. M. Ferman and A. M. Tekalp, "Two-stage hierarchical video summary extraction to match low-level user browsing preerences", *IEEE Trans. on Multimedia*, vol. 5, no. 2, pp. 244-256, 2003.

[2] C. Gianluigi and S. Raimondo, "An Innovative Algorithm for Keyframe Extraction in Video Summarization", *Journal of Real-Time Image Processing*, vol. 1, no. 1, pp. 69-88, 2006.

[3] Y. F. Ma, L. Lu, H. J. Zhang, and M. Li, "A User Attention Model for Video Summarization", *ACM Multimedia Conf.*, 2002.

[4] C. W. Ngo, T. C. Pong, and R. T. Chin, "Video Partitioning through Temporal Slices Analysis", *IEEE Trans. on Circuits and Systems for Video Technology*, 11(8), pp. 941-953, 2001.

[5] C. W. Ngo, Z. Pan, X. Wei, X. Wu, H. K. Tan, and W. Zhao, "Motion Driven Approaches to Shot Boundary Detection, Low-Level Feature Extraction and BBC Rushes Characterization at TRECVID 2005", *TRECVID Workshop*, 2005.

[6] C. W. Ngo, Z. Pan, and X. Y. Wei, "Hierarchical Hidden Markov Model for Rushes Structuring and Indexing ", *Int. Conf. on Image and Video Retrieval*, 2006.

[7] C. W. Ngo, W. L. Zhao, and Y. G. Jiang, "Fast Tracking of Near-Duplicate Keyframes in Broadcast Domain with Transitivity Propagation ", *ACM Multimedia Conference*, Oct 2006.

[8] Z. Pan and C. W. Ngo, "Moving Object Detection, Association and Selection in Home Videos", *IEEE Trans. on Multimedia*, vol. 9, no. 2, Feb 2007.

[9] S. Srinivasan, D. Petkovic, and D. Ponceleon, "Towards Robust Features for Classifying Audio in the CueVideo System", *ACM Multimedia Conference*, 1999.

[10] S. Tang, Y. Zhang, J. Li, X. Pan, T. Xia, and M. Li, "Rushes Exploitation 2006 By CAS MCG", *TRECVID Workshop*, 2006.

[11] C. M. Taskiran, Z. Pizlo, A. Amir, D. Ponceleon, and E. Delp, "Automated Video Program Summarization Using Speech Transcripts", *IEEE Trans. on Multimedia*, vol. 8, no. 4, pp.775-791, 2006.

[12] W. L. Zhao, C. W. Ngo, H. K. Tan, and X. Wu, "Near-Duplicate Keyframe Identification with Interest Point Matching and Pattern Learning ", *IEEE Trans. on Multimedia*, to appear.