Motion Driven Approaches to Shot Boundary Detection, Low-Level Feature Extraction and BBC Rushes Characterization at TRECVID 2005

Chong-Wah Ngo, Zailiang Pan, Xiaoyong Wei, Xiao Wu, Hung-Khoon Tan, Wanlei Zhao
Department of Computer Science
City University of Hong Kong
Email: {cwngo, zerin, xiaoyong, wuxiao, hktan, wzhao2}@cs.cityu.edu.hk

Abstract

This paper describes our experimental results on shot boundary detection (SB), low-level feature extraction (LLF), and BBC Rushes exploration (BR) at TRECVID 2005. The approaches presented in this paper are mostly based on our previous works [1, 2, 3] grounded on motion analysis with spatio-temporal slices, optical flows and tensor representation. This year, our aim is to explore and investigate the role of motion in various fundamental tasks including video structuring and characterization for both the edited (in SB and LLF) and unedited (in BR) videos.

In SB (system C), we exploit the coherence and patterns of motion texture in spatio-temporal slices for boundary detection and classification. The cut and wipe detectors are based on our work in [1] which performs color-texture segmentation on three slices extracted from videos to determine boundaries. The dissolve detector is based on our work in [3] which is composed of two steps: multi-resolution cut detection and binary classification with Gabor features. We submit 10 runs, depending on the size of training data, flashlight detection capability, and additional statistical features (in addition to Gabor) for classification. Overall, the runs with additional features get better results. Increasing training size will sometime deteriorate the precision of detection.

In LLF (system A), a global 6-parameter affine model is estimated at each frame with LMedS and tensor representation for camera motion annotation. To characterize the changes of motion parameters over frames, we use hysteresis thresholding and Kalman polyline estimation developed in [2] to segment and determine the types of motion in shots. We submit 7 runs for LLF, depending on several empirical parameters. Overall, there is no significant difference in term of recall and precision for each run.

In BR (system A), we study two problems: how to structure and characterize BBC rushes? We define three types of segments based on motion: intentional motion (IM), intermediate motion (IMM), shaking artifacts (SA) for structuring. Our aim is to partition-and-classify (or classify-and-partition) the videos into segments corresponding to their motion characteristics.

We employ and experiment three approaches: finite state machine (FSM), support vector machine (SVM), and hidden Markov model (HMM). FSM is unsupervised, while SVM and HMM are supervised. We randomly select and annotate 60 videos (about 337K frames) from the development set for training and testing. The results show that the performances of all tested approaches are quite close, with SVM being better for structuring and HMM being slightly better for rushes characterization. Overall, HMM can achieve over 90% of recall and precision (in term of frame numbers) in extracting intentional motion. For structuring, SVM achieves approximately 70% of recall and 30% of precision (with sub-shot as units), compared to 0.05% of recall and 35% of precision with shot boundary (cut only) detector.

1 Introduction

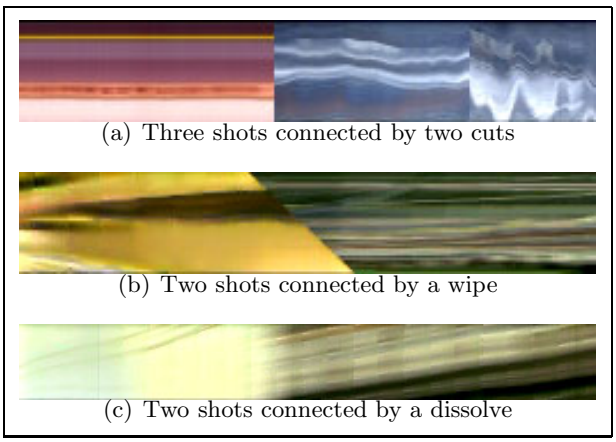
This is the first time we participate in TRECVID. We take part in three tasks, submit 10 runs for shot boundary detection, and 7 runs for low-level feature (camera motion) extraction. In addition, we examine two issues: structuring and characterization of BBC rushes. Our aim at TRECVID 2005 is to investigate the use of motion patterns and features for both edited (news) and unedited (rushes) videos. All works presented in this paper are mostly based on our early works in [1, 2, 3]. Several enhancement, nevertheless, has also been introduced and shown to give improvement over our previous approaches.

2 Shot Boundary Detection

Our approach is based on the segmentation and classification of motion texture patterns in DC-based spatio-temporal (ST) slices [1, 3]. ST slices are $2D$ images extracted from videos with one dimension in space, and the other in time. Figure 1 shows three types of boundaries: cuts, wipes and dissolves on ST slices. We make use of the slice coherence for cut and wipe detection, and the slice pattern for dissolve and non-dissolve classification. Because fade-in and fade-out are special cases of dissolve, we do not consider them separately.

For cut and wipe detectors, we use three slices (center horizontal, vertical and diagonal) and perform color-texture segmentation to locate the boundaries [1]. For dissolve detector, a pyramid of ST slices at different resolutions of time is generated for cut detection. Figure 2 shows the evolution of dissolves to cuts when the resolution of ST slices are temporally reduced. The cuts at low-resolution slices are located with our cut detector and then projected back to the original scale for dissolve verification [3]. We use Gabor features (48 dimensional feature vector) to depict the motion-texture patterns of potential dissolves, and then perform support vector machine for binary classification. In brief, cut and wipe detectors are unsupervised, while dissolve detector is supervised.

On top of our works in [1, 3], we make two improvement: i) flashlight detection and ii) addition features for dissolves. The aim of (i) is to prune false cuts due to sharp lighting changes. We inspect four scans (at $t-2$, $t-1$, $t+1$ and $t+2$) before and after a potential



(a) Three shots connected by two cuts

(b) Two shots connected by a wipe

(c) Two shots connected by a dissolve

Figure 1: Samples of spatio-temporal slices.

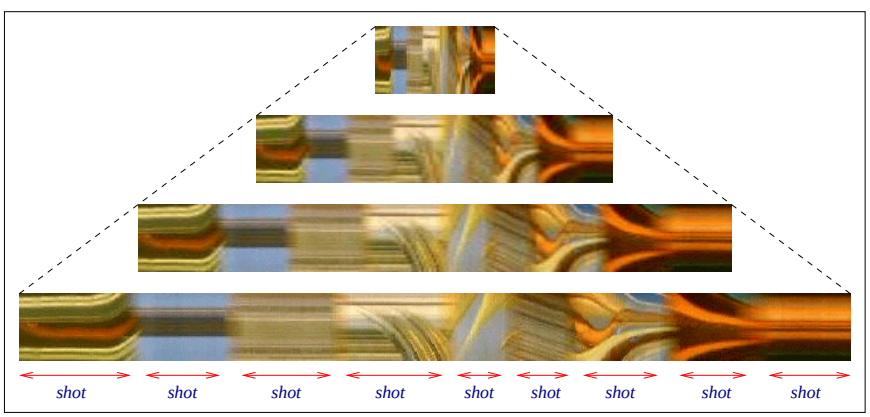


Figure 2: Evolving of *dissolves* to *cuts* in the bottom-up manner along multiple scales in pyramid representation.

cut at time t and then extract the standard deviation of four scans as the feature for decision making. If flashlight happens, the value of this feature often approaches zero since no obvious change between the frames before and after false cuts. The computation is also very efficient since it involves only few scans in ST slices. The aim of (ii) is to improve the precision of dissolve detection, in view that our approach in [3] is not effective enough in discriminating static sequence and dissolve with slight motion. We introduce 9 extra features in addition to Gabor features, add up to a total of 57 dimensional feature vector. These features are extracted by computing the standard deviation of horizontal, vertical and diagonal slices in 3D color space (YCbCr).

2.1 Experiments

We submitted 10 runs according to three aspects: (1) whether flashlight detection is used for cut detection, (2) size of training set, and (3) different features for gradual transition (GT). Table 1 summarizes the characteristics of different runs, and Table 2 shows the results of each run. The small, medium and large data sets contain 864, 1180, 1662 dissolves respectively. The training data is collected from our videos in [1, 3] (so, our system should belong to type C). These data sets are overlapped, i.e., large data set includes all data from the small and medium data sets, while medium data set includes all data from small data set. As shown in Table 1, run-10 gives the best results with 0.870 of recall and 0.796 of precision, compared to one of the best performance (recall=0.927 and precision=0.845) in this year TRECVID. Overall, the performance of our cut detector is quite competitive, but GT detector is not so good probably because we limit the length of dissolve to at least 15 frames during training.

Table 1: Different runs for SB

Run ID	Flashlight Detection for Cut	Training Size	Additional Features for GT
1	×	Medium	×
2	√	Medium	×
3	×	Medium	√
4	√	Medium	√
5	×	Small	×
6	√	Small	×
7	×	Large	×
8	×	Large	√
9	√	Large	×
10	√	Large	√

For cut detection, precision is improved by flashlight detection which indicates this strategy successfully prunes some false cuts due to sharp lighting changes. However, few real cuts are also removed which downgrade the value of recall. The missed cuts generally have brightness preframe and postframe, and their contents are very similar. In our approach, false cuts happen

Table 2: Experimental Results of 10 runs in SB

	ALL		CUT		GT		GT (Frame)	
Run ID	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.
1	0.860	0.733	0.952	0.818	0.591	0.493	0.724	0.589
2	0.860	0.749	0.951	0.842	0.591	0.493	0.724	0.589
3	0.873	0.766	0.952	0.818	0.643	0.600	**0.762**	**0.614**
4	0.873	0.783	0.951	0.842	0.643	0.600	0.762	0.614
5	0.856	0.770	0.952	0.818	0.574	0.598	0.701	0.628
6	0.855	0.787	0.951	0.842	0.574	0.598	0.701	0.628
7	0.858	0.750	0.952	0.818	0.584	0.537	0.736	0.591
8	0.870	0.778	0.952	0.818	0.632	0.640	0.761	0.612
9	0.858	0.767	0.951	0.842	0.584	0.537	0.736	0.591
10	**0.870**	**0.796**	**0.951**	**0.842**	**0.632**	**0.640**	0.761	0.612

when captions in large size come into screen with fast speed, and when there is fade-in or fade-out which happens in less than five frames. The best result we get for cut is 0.951 of recall and 0.842 of precision, compared to one the best run (recall=0.941 and precision=0.928) in TRECVID.

For the GT detection, the additional statistical features, in addition to Gabor, improve both the recall and precision. The size of training data has no obvious impact, and it actually deteriorates performance sometime when its size increase. We are still investigating the possible reasons behind this. Perhaps the training samples itself have noises, or the features we use are not discriminative enough which make the decision boundary even confusing when more training data comes in. In the experiments, false GTs are basically caused by two cases. The first case happens when the scene brightness gradually changes, which generate motion patterns resemble the ones by fade-out. The second case is due to fast camera motion (e.g., zoom-in/out) which results in motion blur and simply generates patterns similar to dissolve. There are also two cases which cause the recall of our GT detection not satisfactory. First, we assume each GT has length of at least 15 frames during training. As a result, we miss a lot of short dissolves. Secondly, some wipe GTs are missed simply because the wipe patterns are too complicated to be detected by our current color-texture image segmentation approach.

In term of speed, our cut and wipe detectors operates in real time. Together, they can run as fast as 90 frame/sec on a Pentium-4 machine. Dissolve detector is not real-time, since significant amount of time is spent in extracting Gabor features.

3 Low-Level Feature Extraction

Our work on LLF is mainly based on our previous work in [2]. Basically, we describe a shot as sequences of motion trajectories. Our task of LLF is to characterize these trajectories with hysteresis thresholding and Kalman polyline estimation to either pan/track, zoom/dolly or tilt/boom.

3.1 Motion Feature Extraction

The motion features are extracted from every two adjacent frames. Harris corner detector is applied to extract the image feature points, $\mathbf{x}_t$ of a frame t. The corresponding points, $\mathbf{x}_{t+1}$, at frame $t+1$, are estimated by the singular value decomposition of 3D structural tensor [2]. The matched point pairs in each two frames are assumed to be consistent with the single camera motion model. Since pan/track, zoom/dolly and tilt/boom are respectively belong to the same feature category, 2D camera motion model is sufficient for the representation of the three motion categories. To seek balance between model effectiveness and complexity, we decide to use the 2D 6-parameter affine model described as,

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + \mathbf{v},$$

where $\mathbf{v} = [v_x, v_y]^T$ is the translation, and A is a 2×2 matrix. A and $\mathbf{v}$ are estimated from the matched points in two consecutive frames using the robust estimator LMedS [4]. RANSAC is not used due to the requirement of inlier threshold which cannot be easily set. A can be further represented by other motion features: rotation θ, skew ϕ and zoom (dolly) $[z_x, z_y]^T$ as follows,

$$A = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} z_x & z_x \tan(\phi) \\ 0 & z_y \cos(\phi) \end{bmatrix}.$$

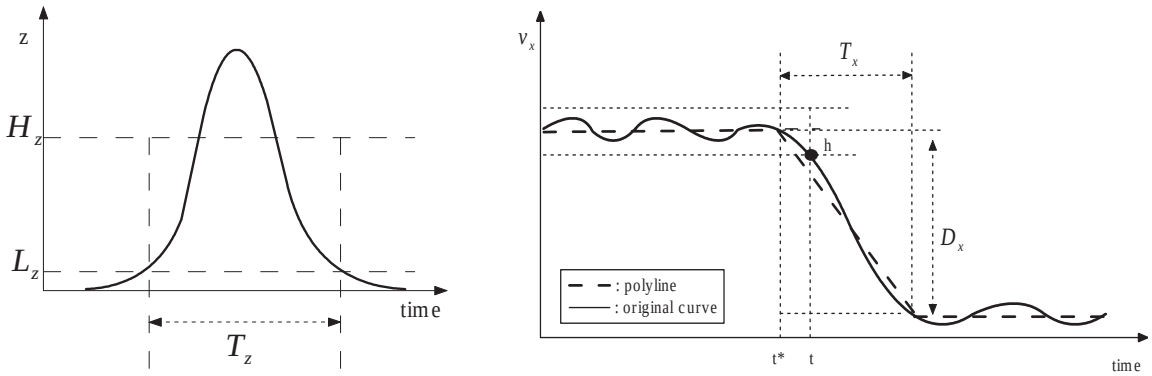
The parameters v_x and v_y characterize pan (track) and tilt (boom) respectively, while z_x and z_y can be used for zoom (dolly) detection. Therefore, we extract a 3-dimensional motion feature vector $\mathbf{f} = [v_x, v_y, z = (z_x - 1) \times (z_y - 1)]$ for each two adjacent frames. Carrying on this procedure along the temporal dimension, we can get a sequence of motion feature vectors $\{\mathbf{f}\}$ for a shot. Grounded on $\{\mathbf{f}\}$, we develop techniques to detect the patterns of camera motion as described in the following section.

3.2 Camera Motion Detection

There are three camera motion categories being considered in TRECVID 2005: pan (track), tilt (boom) and zoom (dolly). The scenes of pan/track and tilt/boom can be approximately regarded as moving parallel to the image plane, whereas the zoom (dolly) moves along the depth direction. This results in different patterns in the feature sequences of zoom (dolly) and the other two categories. Two examples are given in Fig. 3 to illustrate the idea. Based on the sequence patterns, we develop two separate approaches: one to detect zoom (dolly) and the other for pan (dolly) track and tilt (boom) detection.

3.2.1 Zoom (dolly) Detection

Zoom (dolly) detection is relatively challenging for several reasons. First, the geometric relation between the image pixel motion and the zoom (dolly) camera motion is nonlinear. Secondly, this kind of motion, especially zoom, is often un-smooth. These characteristics lead to the diverse patterns of zoom (dolly) when inspecting its motion feature sequence z. To make the pattern



(a) Zoom (dolly) detection

(b) Pan (track) detection

Figure 3: Camera motion detection

detection task realistic, we only capture the patterns that can stay for a long enough duration. One typical example is shown in Fig 3(a). Our approach for zoom (dolly) detection is based on hysteresis thresholding [5]. As shown in Fig 3(a), two thresholds are used: the lower one L_z and the higher one H_z. The advantages of using two thresholds are twofold: i) H_z locates the segments with higher probability of detection while preventing from the selection of faulty zoom segments caused by noise; ii) L_z collects the lost part rejected by H_z and find the duration of the zoom (dolly). If the duration is longer than a predefined threshold T_s, this piece is regarded as zoom (dolly) motion, and the corresponding shot is retrieved accordingly.

3.2.2 Pan (track) and Tilt (boom) Detection

Although the pan and track are different camera motion from 3D point of view, they have similar 2D motion effect. When only few frames are considered, they can be assumed to have constant velocity in 2D. Therefore, their camera trajectories can be approximated by a series of connected lines, so called polylines. So do the tilt and boom. Under this formulation, the task of camera detection is equivalent to the detection of slope lines that are long or significant enough to be regarded as pan (track) or tilt (boom). To better illustrate the idea, we show the cumulative of v_x in Fig. 3(b) as an example. This figure depicts the situation of a static-pan-static transition. The observed trajectory of v_x cumulative is represented by the solid curve. A 3-segment polyline (dashed lines) is estimated to fit the observed curve. This is done by the Kalman polyline estimator as proposed in [2]. Each line is depicted by two factors: the displacement and the duration. Then two threshold D_x and T_x, as shown in Fig 3(b), are used to determine whether if a line is pan (track). If a video shot has a line of pan (track), it is labeled as pan (track) category, and similarly for tilt (boom) category.

3.3 Experiments

There are two thresholds for pan (track) $[D_x, T_x]$ and tilt (boom) $[D_y, T_y]$ respectively, and 3 thresholds for zoom(dolly) $[H_z, L_z, T_z]$. For zoom (dolly) selection, L_z can be easily set with a

Run ID	D_x	T_x	D_y	T_y	H_z	T_z
1	40	30	45	25	500	18
2	40	25	40	25	100	15
3	50	30	45	30	50	18
4	50	25	40	30	50	15
5	40	35	40	20	10	18
6	40	20	45	20	100	12
7	50	35	45	35	10	15

Table 3: Best seven combinations trained with the development set. Each one represents a run.

low value to guarantee high recall. The other 6 thresholds are empirically trained with videos randomly selected from the development set. Basically we give four choices to each threshold: $D_x = [20, 30, 40, 50]$, $D_y = [30, 35, 40, 45]$, $T_x = T_y = [20, 25, 30, 35]$, $H_z = [10, 50, 100, 500] \times 10^{-7}$ and $T_z = [10, 12, 15, 18]$, thus each motion category has 4×4 potential pairs. To determine the best possible combinations, we annotate 6 videos in the development set for training. The seven combinations (shown in Table 3) which give the best results in term of F-measure are selected for testing.

Table 4 shows our experimental results of 7 runs for LLF. There is no significant difference among the runs, which could probably indicate that the trained thresholds are not sensitive to camera motion detection. In our results, false positives are mostly due to foreground object motion, particularly for large objects such as cars and people which can easily generate alarms for pan (track) and zoom (dolly). This result is not surprised since the robust method (LMedS) we use can only deal with dominant and global motion. For false negatives, many are due to slow motion. Based on our observation, even human needs to take time to figure out there are actually camera motion in some of these shots. Since our current approach only samples two adjacent frames at each time instance, we indeed cannot effectively handle slow motion. Several false negatives are due to the case that camera motion only exists in a small window of a video frame. Since our robust estimation is based on least median, if the number of feature points in the window is not larger than the remaining regions, the camera motion exists in the window will be rejected.

Despite the difficulties, overall we still obtain encouraging results with F-measures of 0.9089, 0.7782, 0.8729 for pan/track, tilt/boom, zoom/dolly respectively in our best run. Figure 4 compares our best run (CityU-Max) with the best run (TRECVID05-Max) and median run (TRECVID05-Median) in TRECVID.

4 BBC Rushes Exploration

BBC rushes are unedited videos, without caption and clean speech. Basically they are featured with long shot, shaking artifacts and fast motion characteristics. Traditional shot boundary

	Pan (track)		Tilt (boom)		Zoom (dolly)		Mean	
Run ID	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall
1	0.959	0.833	0.970	0.619	0.968	0.767	0.966	0.740
2	0.955	0.867	0.964	0.633	0.941	0.814	0.953	0.772
3	**0.956**	**0.860**	**0.965**	**0.652**	**0.968**	**0.767**	**0.963**	**0.760**
4	0.957	0.828	0.965	0.648	0.982	0.728	0.968	0.735
5	0.956	0.842	0.964	0.643	0.941	0.814	0.954	0.766
6	0.957	0.830	0.969	0.605	0.968	0.767	0.965	0.734
7	0.975	0.806	0.965	0.652	0.982	0.728	0.974	0.729

Table 4: Experimental results of 7 runs in LLF.

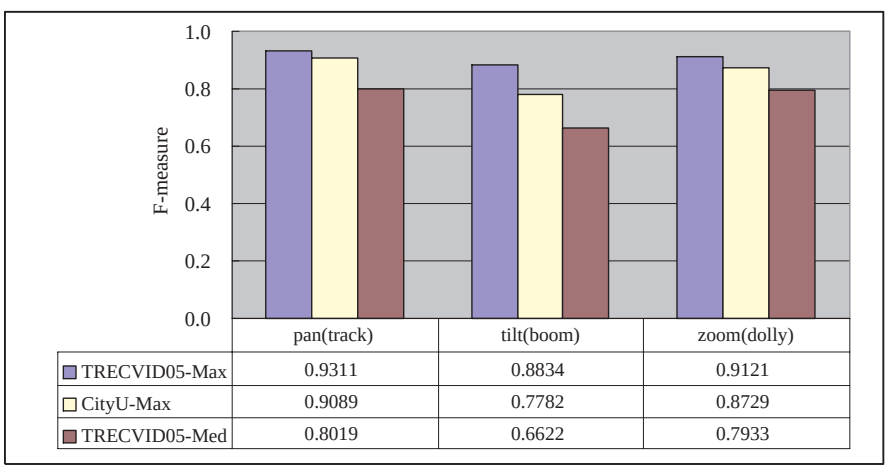


Figure 4: LLF: Comparison to other approaches (best and median runs) in TRECVID'05.

detection cannot effectively structure BBC rushes since false alarms can easily happen due to camera shaking, and most importantly it cannot separate intermediate motions from the real intentions of cameramen. In BBC rushes, a typical long shot can involve camera motion like: "pan to get a larger and wider view of scene", "pan to search for objects-of-interest", and "zoom-and-hold to freeze the impression". One straightforward way to structure BBC rushes, or broadly unedited videos, is to predict and extract the "cameraman's intention" to facilitate browsing and searching.

This year, we study two problems: how to structure and characterize BBC rushes? The primary feature we use is motion (means and variances of affine parameters) since one of the best way in determining cameraman's intention is by analyzing the camera motion trajectories. We define three types of segments: intentional motion (IM), intermediate motion (IMM), shaking artifacts (SA) for structuring. In principle, we look for and index segments with IM, and retain segments with SA for potential use, while removing segments with IMM from indexing. To achieve the aforementioned task, technically the aim is to partition-and-classify (or classify-and-partition) the videos into segments (or sub-shots) corresponding to their motion characteristics.

We employ and experiment three approaches: finite state machine (FSM) [2], support Vector machine (SVM) [6], and hidden Markov model (HMM) [7]. FSM is unsupervised, while SVM and HMM are supervised.

4.1 Finite State Machine (FSM)

FSM is an unsupervised approach and we use it to partition-and-classify sub-shots in BBC rushes. There are three steps in FSM. The first step is motion feature extraction: a motion feature vector $\mathbf{f} = [v_x, v_y, z]^T$ is extracted for each two adjacent frames. The details have been described in Section 3.1. The second step is to temporally separate the video into a series of consecutive segments based on the motion feature vectors. This step is basically similar to the description in Section 3.2. Each partitioned segment is characterized as either one of the following types: *static*, *move (pan or tilt)*, *zoom*. The details of algorithms can be found in our early work [2]. However, unlike edited videos, the segments obtained from BBC rushes are noisy and trivial with effects such as motion blur, flashing and camera shaking. Thus, we further characterize these segments as either: shot static (S_s), long static (L_s), shot move (S_m), long move (L_m), or zoom (Z). In addition, a segment is labeled as "short" if its duration is less than half of a second.

Based on these segments, FSM performs classification to label the segments as IM, IMM or SA. Fig 5 shows the proposed FSM which is composed of four states A, B, C, D. The states are described as follows.

- **State-A** is the initial state.
- **State-B** represents the segments of intentional motion, where camera is in slow and smooth movement. The temporal consecutive segments in this state are merged into a larger segment labeled as IM.
- **State-C** characterizes the segments of intermediate and shaky motions, which usually consists of short move, long move and zoom. The temporally consecutive segments in this state are also merged as a segment which is labeled as SA if the rate of change of camera direction [8] is high, otherwise it is labeled as IMM.
- **State-D** is used to keep the segments of short duration that cannot be classified temporarily, which are merged into the next state.

As presented, both the partitioning and characterization can be done on-the-fly in the finite state machine.

4.2 Support Vector Machines (SVM)

Different from FSM, SVM performs classify-and-partition. We first divide a video into equal length segments, each with one second duration. For each segment, we extract 9 dimensional

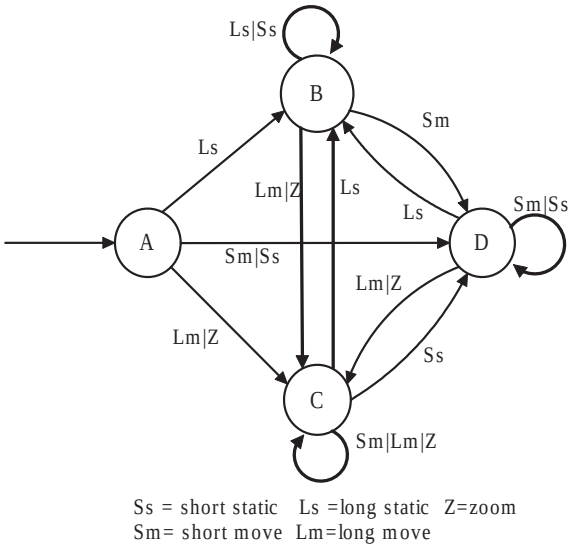


Figure 5: The finite state machine for video segment detection

feature vector $\mathbf{f} = (M_x, M_y, Z_x, Z_y, D_x, D_y, V_x, V_y, S)$, where

$$M_x = \underset{i=1}{\overset{N}{\text{mean}}}(|v_i^x|), \quad M_y = \underset{i=1}{\overset{N}{\text{mean}}}(|v_i^y|)$$

$$Z_x = \underset{i=1}{\overset{N}{\text{mean}}}(|z_i^x|), \quad Z_y = \underset{i=1}{\overset{N}{\text{mean}}}(|z_i^y|)$$

$$D_x = \underset{i=1}{\overset{N-1}{\text{mean}}}(|v_{i+1}^x - v_i^x|), \quad D_y = \underset{i=1}{\overset{N-1}{\text{mean}}}(|v_{i+1}^y - v_i^y|)$$

$$V_x = \underset{i=1}{\overset{N-1}{\text{var}}}(|v_{i+1}^x - v_i^x|), \quad V_y = \underset{i=1}{\overset{N-1}{\text{var}}}(|v_{i+1}^y - v_i^y|)$$

$$S = \underset{i=1}{\overset{N-1}{\text{mean}}}(|\mathbf{v}_{i+1}||\mathbf{v}_i| - \mathbf{v}_{i+1} \cdot \mathbf{v}_i)$$

where $(\mathbf{v}_i = (v_i^x, v_i^y), z_i^x, z_i^y)$ are the translation and zoom factors in both x and y directions. The subscript i denotes the ith frame in a segment. These four motion features are estimated by the method described in Section 3.1. The 9 features, derived from the 4 motion features, represent different motion properties. The parameters M_x, M_y, D_x, D_y, V_x and V_y denote the speed and variation of translational motion, which can help to classify the IM and IMM. The parameters Z_x and Z_y can facilitate the detection of IMM segments caused by zoom operations. The feature S is used to improve the classification of SA. It is actually the mean of the value: $|\mathbf{v}_{i+1}||\mathbf{v}_i|(1 - \cos\theta)$, where θ is the angle between $\mathbf{v}_{i+1}$ and $\mathbf{v}_i$. Therefore, two successive fast translational motions with significant direction change would lead to large S value, which hints the labeling of SA.

We use radial basis function as the kernel of SVM [6]. The SVM is trained to classify three types of motion segments: Intentional (IM), Intermediate (IMM) and Shaky Artifacts (SA). The segments of videos are then labeled by the trained SVM classifier. The adjacent segments with the same label are merged and the sub-shot boundaries are located directly.

4.3 Hidden Markov Model (HMM)

Since our task is to temporally structure and characterize the contents of BBC rushes, it is intuitive to relate this task with HMM. Furthermore, determining intentional and intermediate motions sometime requires observations across several consecutive segments. Second order decision (look at multiple samples to make decision) should be better than first order decision (look at one sample and make decision at a time) in principle. With HMM, video structuring can be modeled with state transition, while video characterization can be regarded as state prediction. The HMM model we develop has three states represent respectively the intentional (IM), intermediate (IMM) motion and shaky artifact (SA). Same as SVM, a video is first divided into equal length segments. The 9 dimensional feature vector (same as SVM) of a segment forms one observation sequence for HMM. After training, the HMM is used to classify the successive segments in videos by the Viterbi algorithm. Similar to SVM, the boundaries are located by merging adjacent segments with same label.

4.4 Experiments

We assess the performance by evaluating the results of rushes structuring and characterization. We randomly select 60 videos (337K frames) from the development set, and manually annotate the sub-shots and their motion categories:[1] intentional (IM), intermediate (IMM) and shaking artifact (SA). We use 30 videos for SVM and HMM training, and another 30 videos for testing. For training purpose, each video is divided into equal length (one sec) segments with label of either IM, IMM or SA. For HMM, the labeled segments of a video form an observation sequence. Thus, there are 30 observation sequences in total for training HMM. For SVM, in contrast, we select equal amount of training segments for each motion category to guarantee unbiased training.

4.4.1 Structuring

Compared to shot boundaries, the sub-shot boundaries are fuzzy and the exact location (in term of frame) is not easy to identify even with careful human inspection. In this experiment, a sub-shot boundary is counted as correct as long as we can find a matched boundary in the ground-truth within 1 second time frame. In our ground-truth, there are 83.4% of boundaries for transitions from IM to IMM, 16.3% from IM to SA, and 0.3% from IMM to SA. As a baseline comparison, we also implement a HMM using shot duration (frames) as input, denoted by SHMM. The video shots are detected by our cut detector described in Section 2. The intuitive idea of SHMM is that long shot should normally confirm to intentional motion, while short shots which may be caused by false cuts may correspond to segments with shaking and fast camera motion. To show that other approaches (FSM, SVM, HMM) are valid for this task, they should have better capability in locating sub-shot boundaries comparing to traditional cut detector.

[1] In reality, we should also consider two other categories: motion blur and illumination changes which also happen frequently in BBC rushes (or any other unedited video).

	Training		Testing	
	Recall	Prec.	Recall	Prec.
FSM	0.614	0.282	0.593	0.279
SVM	**0.769**	0.281	**0.763**	0.289
HMM	0.461	**0.419**	0.395	**0.379**
SHMM	0.060	0.355	0.056	0.322

Table 5: Results of structuring BBC rushes.

	IM		IMM		SA	
	Recall	Prec.	Recall	Prec.	Recall	Prec.
FSM	0.815	0.981	**0.802**	0.118	0.011	0.050
SVM	0.827	**0.990**	0.701	**0.162**	**0.715**	0.239
HMM	**0.927**	0.970	0.329	0.137	0.311	**0.339**

Table 6: Results of characterizing BBC rushes (training videos).

Table 5 shows the experimental results for four approaches. Notice that FSM is unsupervised while the rests are supervised approaches. Overall, SVM give the best recall (above 75%) in locating boundaries, followed by FSM (about 60%). On the other hand, the performance of HMM of using either motion (HMM) or duration (SHMM) is basically poor. Particularly, SHMM simply fails in locating most boundaries, especially for the boundaries connecting the intentional and intermediate segments. Based on the results in Table 5, structuring of BBC rushes is still quite difficult, considering the low value of precision. Of course, with the aid of context information (e.g., time logs from DV camera), this tasks should not be so difficult although we still need methods to separate segments of intentional and intermediate motions to facilitate browse and search.

4.4.2 Characterization

Table 6 and Table 7 show the results of characterizing BBC rushes. The results are assessed based on the number of frames, not sub-shots, being correctly or wrongly classified. Since the amount of IM segments is more than the other two, and IM, in principle, represents the samples useful for search, we should concentrate more on the results for IM. The experimental results show that HMM has best results (90% of recall and precision) in extracting segments with IM. The unsupervised approach (FSM) is close to the performance of SVM in both training and testing data sets. SVM has better results in discriminating IMM and SA perhaps because of the use of extra motion features (e.g., angle change). From the experiments, basically we find that IMM and SA is difficult to be separate, although we thought IMM are segments with fast camera motions but relatively much stable compared with SA segments.

	IM		IMM		SA	
	Recall	Prec.	Recall	Prec.	Recall	Prec.
FSM	0.756	0.968	**0.844**	0.128	0.000	0.000
SVM	0.778	**0.975**	0.456	0.120	**0.362**	**0.182**
HMM	**0.909**	0.929	0.375	**0.196**	0.043	0.067

Table 7: Results of characterizing BBC rushes (testing videos).

5 Future Works

We have conducted experiments for three tasks in TRECVID 2005, with motion patterns and features as primary cues. For dissolve detection, we make a wrong assumption which results in relatively low recall. Whether to use patterns in ST slices as cues for discriminating dissolve/non-dissolves is still under our investigation. Basically we need more powerful and fast-to-extract features for classification. In LLF camera motion annotation, we achieve good results, but still face problem since we only sample motion parameters from two adjacent frames. We believe computing motion parameters along a large temporal scale can solve some difficulties due to slow camera motion and large foreground objects. For BBC rushes, there are still lot of works remain to be investigated. This year, we only concentrate on structuring, characterizing, and extracting potentially useful segments that good for browsing and retrieval.

Acknowledgment

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 1072/02E).

References

[1] C.-W. Ngo, T.-C. Pong, and R. T. Chin, "Video partitioning by temporal slice coherency," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 8, pp. 941– 953, 2001.

[2] Z. Pan and C.-W. Ngo, "Structuring home video by snippet detection and pattern parsing," in *ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR)*, 2004, pp. 69–76.

[3] C.-W. Ngo, "A robust dissolve detector by support vector machine," in *ACM Conference on Multimedia (MM)*, 2003.

[4] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection.* Wiley New York, 1987.

[5] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach.* Prentice Hall, 2003.

[6] V. N. Vapnik, *The nature of statistical learning theory*, 2nd ed. New York: Springer, 2000.

[7] L. Rabiner and B.-H. Juang, *Fundamentals of speech recognition.* Prentice Hall, 1993.

[8] S. Wu, Y.-F. Ma, and H.-J. Zhang, "Video quality classsification based home video segmentation," in *IEEE Int'l Conf. on Multimedia and Expo*, jul 2005, pp. 217–220.