University of Marburg at TRECVID 2006: Shot Boundary Detection and Rushes Task Results

Ralph Ewerth[1,2], Markus Mühling[1,2], Thilo Stadelmann[1,2], Ermir Qeli[2], Björn Agel[2], Dominik Seiler[2], and Bernd Freisleben[1,2]

[1] *SFB/FK615, University of Siegen, D-57068 Siegen, Germany*
[2] *Dept. of Math. and Computer Science, University of Marburg,D-35032 Marburg, Germany*
{ewerth, muehling, stadelmann, ermir, agel, seiler, freisleb}@informatik.uni-marburg.de

Abstract

In this paper, we summarize our results for the shot boundary detection task and the rushes task at TRECVID 2006. The shot boundary detection approach which was evaluated last year at TRECVID 2005 served as a basis for our experiments this year and was modified in several ways. First, we investigated different parameter settings for the unsupervised approach. Second, we experimented with the possibility to create an unsupervised ensemble that consists of several clusterings that have been obtained with different parameter settings. Our prototype for the rushes task consists of a summarization component and a retrieval component. Rushes videos are segmented on a sub-shot basis in order to separate redundant from non-redundant sequences within a shot. The summarization component is based on sub-shots clustering, and an appropriate visualization of clusters is presented to the user. The sub-shots are clustered with respect to a number of low-level and mid-level features, and they are visualized such that the user can navigate through these sub-shots. The retrieval component enables the user to search the rushes material automatically according to several features: camera motion, audio features (silence, speech, music, action, and background), speaker identity and interviews, shot sizes, face appearances, and by queries by example based on color and texture features.

1. Structured Abstract

The paper is structured as follows. In this section, the results of our participation in both tasks are presented in form of the requested structured abstract. The shot boundary detection approach and the related experimental results are presented in section 2. Our system suggested for the exploration of the rushes material is described in Section 3 along with the experimental results. Section 4 concludes the paper.

The following definitions are used in this paper:

$$recall = \frac{\#correctDetectedItems}{\#Items} \quad (1)$$

$$precision = \frac{\#correctDetectedItems}{\#correctDetectedItems + \#falseAlarms} \quad (2)$$

$$f1 = \frac{2 * recall * precision}{recall + precision} \quad (3)$$

Shot Boundary Detection: "What approach or combination of approaches did you test in each of your submitted runs?"
The unsupervised approach investigated this year relies on our TRECVID system from 2005 [6]. In this system, k-means clustering is used for both cut detection and gradual transition detection.
To detect cuts, two different frame dissimilarity measures are applied: Motion-compensated pixel differences of subsequent DC-frames [4] and the histogram dissimilarity of two frames within a pre-defined temporal distance of 2. A sliding window technique similar to [23] is used to measure the relative local height of a peak value. For cut detection, the best sliding window size is estimated by evaluating the clustering quality of "cut clusters" for several window sizes. Thus, the minimum and maximum sliding window sizes serve as parameters for both dissimilarity metrics. Several ranges for this parameter were tested in the experiments for both dissimilarity measures. Also, the unsupervised approach was optionally extended for cut detection by additional unsupervised respectively supervised classifiers to obtain an

ensemble of classifiers. Optionally, a false alarm removal took place.
To detect gradual transitions, dissimilarity measures for different frame distances were applied. Feature vectors were created similar to the cut detection approach using a sliding window technique. K-means was applied to cluster these feature vectors. This approach was extended by a fade detector following the proposal in [21]. Finally, false alarms were removed if the start and end frame of a transition are too similar. Several runs were submitted and tested, and the different parameter settings for each run are described in Table 1. For the purpose of simplification the run ids are renamed to marburg0 – marburg9 as follows:

marburg0: MR_1_2_1_6_1_8_0_11_def:
marburg1: MR_1_2_3_10_6_20_0_removefalsecuts
marburg2: MR_1_2_4_10_4_20_0_removefalsecuts
marburg3: MR_1_2_4_10_4_20_0_refcm_votes2mini
marburg4: MR_1_2_4_10_4_20_2_kmeans_3er_ens.
marburg5: MR_1_2_4_10_4_20_2_kmeans_ensemble
marburg6: MR_1_2_4_10_4_20_4_kmeans_5er_ens.
marburg7: MR_1_2_4_14_5_11_0_remfcuts
marburg8: MR_1_2_5_10_10_20_0_removefalsecuts
marburg9: MR_1_2_5_10_5_20_0_removefalsecuts

Shot Boundary Detection: "What, if any significant differences (in terms of what measures) did you find among the runs?"
The different settings of the range for the minimum and maximum possible sliding window size had very little impact on the detection results in case of cut detection: Recall was about 90% and precision about 80% for all runs. Increasing the maximum sliding window size led to a slightly higher precision rate, while recall was slightly lower, as expected. The extension of the unsupervised basis system to form an ensemble with additional supervised or unsupervised classifiers did not improve the cut detection performance significantly.
The detection of gradual transitions was quite stable as well, achieving in the best case a recall of about 60% and a precision of 67%. Frame-based recall was about 59% and frame-based precision about 74%.

Shot Boundary Detection: "Based on the results, can you estimate the relative contribution of each component of your system/approach to its effectiveness?"
The runs submitted this year ended up with the same performance in nearly all cases.

Rushes Task: "What approach or combination of approaches did you test?"
Our prototype for the rushes task consists of a summarization component and a retrieval component. The summarization component is based on clustering of sub-shots and an appropriate visualization of sub-shot clusters. Rushes videos were segmented on a sub-shot basis with respect to camera motion, face appearances, speech and silence in order to separate redundant from non-redundant sequences within a shot. Those sub-shots were clustered with respect to a number of low-level and mid-level features, and they were visualized such that the user can navigate through these sub-shots. The retrieval component enables the user to search the rushes material according to several features: camera motion (pan, tilt, and zoom), interviews, audio information (silence, music, speech, action, background), speaker identity, shot sizes, face appearance, and a query by example system based on color and texture features.

Shot Boundary Detection and Rushes Task: "Overall, what did you learn about runs/approaches and the research question(s) that motivated them?"
The unsupervised approach to shot boundary detection has reached a mature level of robustness and detection quality, in particular for the task of cut detection. The precision is significantly lower than last year. Analysis showed that in case of cut detection many of the false alarms are not definite detection failures, they could be judged as a cut as well. An example for such a case are picture-in-picture effects where a shot is displayed in a part of the frame region: Though there might be a cut in this frame part, the rest of the frame remains the same – how to judge such cases remains a subjective decision.
The system developed for the rushes task provides a video summarization by a visualization of sub-shot clusters as well as a retrieval component to search several features. Experiments indicate that sub-shots are a reasonable processing unit to explore rushes videos. The precision of retrieved sub-shot lists is promising but it should be further improved for individual features. In the future, it is planned to extend the retrieval component with the possibility to search for several additional high-level features.

2. Shot Boundary Detection

The shot boundary detection approach is split up in two parts in order to detect cuts and gradual transitions appropriately.

2.1 Video Cut Detection

Unsupervised learning is utilized in the cut detection approach which was optionally extended to an ensemble of several classifiers using majority voting. For this purpose, additional classifiers were trained on an appropriate training set respectively variants of the unsupervised approach were used with different parameter settings. The basic unsupervised approach to cut detection works as follows.

Two frame dissimilarities are used for the unsupervised cut detection task. Motion compensated pixel differences of subsequent frames (i.e. their frame distance is 1) and frame histogram differences are computed. The histograms have 512 bins where each bin represents a combination of the Y, Cb and Cr color channel each with 8 quantization levels. A frame distance of 2 is used for a second frame dissimilarity metric which is aimed at detecting very short gradual transitions. Time series that are based on a frame distance $n>1$ are subsampled by a factor of n.

For the dissimilarity values with a frame distance of 1, GoP-oriented (GoP: Group of Pictures) frame difference normalization is applied [3] to remove compression artifacts in the dissimilarity time series. Two features are extracted for both metrics at frame positions where the dissimilarity is the maximum in the middle of a sliding window of size $2*m+1$:

1.) *max*: the ratio of the dissimilarity value divided by the maximum dissimilarity value in this video, and,

2.) *sec*: the ratio of the second largest value divided by the maximum of the sliding window.

Then, k-means (k is known a-priori in case of cut detection: 2) is applied separately for each metric using all feature vectors belonging to the same sliding window size and metric. The cluster whose average feature vector is nearer to the feature space point (1, 1) is considered as the "cuts" cluster. Now, for each cluster and for each sliding window size and metric the silhouette coefficient is computed which describes the compactness of a cluster (more details are described in [4]). The cluster with the highest coefficient represents the best sliding window size for a given metric and is considered as the cut detection result. If a cut is detected using both metrics, only the shorter transition is included in the final result.

2.2 Video Cut Detection Using an Ensemble of Classifiers

Last year, we have extended the unsupervised basic system with two supervised classifiers. This year, we investigated the possibility to also extend the basis system with two or four additional unsupervised classifiers using different settings with respect to minimum and maximum sliding window size.

It has been shown that such an ensemble of classifiers can improve accuracy in recognition tasks [10]. Since most transitions in a video are abrupt (without any transitional frames between the different shots), the ensemble was added to the unsupervised approach for cut detection. The extension of the basis system to an ensemble with supervised classifiers works as follows.

We have chosen Adaboost (e.g. described in [22]) as the first classifier to select the best features for a given training set (in our case the TRECVID 2005 shot boundary test set). The key idea of the Adaboost approach is to combine a number of n "weak classifiers" to build a strong classifier within n rounds of training. For each feature, a threshold is estimated which minimizes the classification error on the (re-weighted) training set. The classification error is computed based on the weights of the training samples. Misclassified training samples are re-weighted such that they have more impact in the next training round for the next "weak classifier". Each "weak" classifier's weight depends on its error rate. The final strong classifier rule checks if the weighted sum of the weak classifiers' positive votes exceeds a threshold. For the task of cut detection, we have defined 42 features for a certain frame distance describing dissimilarity of DC-frames with respect to:

- motion compensated pixel differences,
- histogram differences,
- luminance mean and variance,
- edge histograms of Sobel-filtered (vertically and horizontally) DC-frames,
- local histogram differences, and
- ratio of the second largest dissimilarity value divided by the local maximum for several sliding window sizes.

In this study, we have further used Adaboost for feature selection where the best $m<=n$ features are used to train a SVM on the same test set of 2005. Two frame distances (1 and 2) were investigated resulting in a total feature number of 84. Thus, we finally got three classifiers evaluating each frame (considering the unsupervised approach as a kind of classifier as well). A majority vote is implemented in our approach, i.e. a cut is detected if at least two "experts" vote that a frame belongs to a new shot. In [7], we have modified this system in a manner that an ensemble of classifiers is built adaptively for a given video. In this approach, classifiers are trained with training data that are labeled automatically using the unsupervised basic approach,

employing only the data from the video under consideration.

2.3 Gradual Transition Detection

The main idea of the gradual transition detection approach is to view a gradual shot change as an abrupt shot change at a lower temporal resolution. It is also an unsupervised approach. This basic approach is extended by a fade detector following the approach in [21]. The approach to detect gradual transitions works as follows:

1.) First, frame dissimilarities are computed based on histograms of approximated DC-frames. Those dissimilarities are computed for certain temporal resolutions Δt. To detect gradual transitions, frames are compared at a higher temporal distance, e.g. up to 50 frames. Due to this, a histogram based metric seems to be more suited to compute frame dissimilarities than a motion compensated pixel-based comparison, which is more sensitive to object and camera motion. A subsampled set of frame dissimilarity values $\{d_{0,\Delta t}, d_{1,\Delta t}, ..., d_{i,\Delta t}, ..., d_{n/\Delta t,\Delta t}\}$ is obtained for each of the temporal resolutions Δt, where $\Delta t \in N\backslash\{0\}$, and $d_{i,\Delta t}$ is the dissimilarity value for the frames $i*\Delta t$ and $(i+1)*\Delta t$. As mentioned above, the idea is to view a gradual shot change as a cut at a lower temporal resolution. Therefore, each time series of dissimilarity values for resolution Δt is subsampled by the factor Δt. If a gradual transition of length k (this might be a cut of length 0 as well) starts at position n, in all time series with $\Delta t >= k$ this transition should be represented by a peak in the dissimilarity measurements.

2.) The feature vectors are now created similar to the task of cut detection and consist of the same two components: *max* and *sec*. The value *max* is normalized for each time series Δt using the respective maximum. For each temporal resolution Δt the basic sliding window size of $2*m+1$ is set separately based on the parameter x: $m=max(x/\Delta t, c)$, where c is a constant and controls the minimum size for m, e.g. c=2. The parameter x represents the length of the sliding window at the finest temporal frame resolution. By computing m separately for each Δt, fewer dissimilarity values are taken into account at lower temporal resolutions due to the preceding subsampling. The sliding window is not kept at constant absolute size for the lower temporal resolutions since neighboured cuts and transitions would probably fall in the sliding window and affect the usefulness of the parameter *sec*.

3.) Then, these feature vectors are clustered using k-means (again with 2 clusters). Different strategies are possible to cluster the time series. First, clustering can be conducted separately for each time series Δt, and the clustering result must be merged afterwards. Alternatively, the feature vectors of all time series can be clustered in one clustering process since the feature vectors are normalized accordingly. After k-means clustering, the members of the gradual transition cluster(s) must be processed further. If clustering has been applied for each Δt separately, then several "transition clusters" exist, one for each temporal resolution. In case when two or more feature vectors are finally in one or more "transition cluster" and have a frame overlap, or a cut has been detected in this frame interval before, then, the longer transition(s) are removed. The transition start and end positions are optionally refined by comparing the dissimilarity between pairs of frames in the transition interval. Finally, false alarms are removed if the start frame and the end frame are too similar, i.e. the dissimilarity value between start and end frame is below a threshold.

2.4 Experimental Results

The shot detection approach was tested with the following parameter settings for all runs. The frame distance for the second cut detection metric was set to 2. The frame distances for the gradual transition detection were set to: 6, 10, 20, 30, 40, 50; the parameter x describing the initial sliding window size for the finest temporal resolution was set to 24. The MDC decoder was used for MPEG decoding [11]. Feature selection using Adaboost was performed on the TRECVID 2005 shot boundary test set. Eleven features were selected from the whole feature set to build an Adaboost classifier. Those best features were used to train a SVM [1] on eight of the twelve videos from the last year's test set.

The experimental settings and the results for the different runs are shown in Table 1 and 2. The parameters for the sliding window sizes had very little impact for both cut detection and gradual transition detection, and the results are very similar for all runs. For cut detection, recall is about 90% and precision about 80%, for gradual transition detection recall is about 59% and precision about 67%. Increasing the maximum sliding window size led to slightly better precision values, whereas recall decreased very slightly. This effect can be observed if one compares the run marburg0 with the other runs. For marburg0, a lower range of window sizes was used which yielded an increased recall and a decreased precision. The unsupervised ensemble approaches marburg4-6 achieved the best results in terms of the f1-measure (see Table 3), but the improvement is not significant.

Run	Cuts: MinWin Size Metric1	Cuts: MaxWin Size Metric1	Cuts: MinWin Size Metric2	Cuts: MaxWin Size Metric2	#Classifiers	False Alarm Removal
marburg0	1	6	1	8	1	0
marburg1	3	10	6	20	1	1
marburg2	4	10	4	20	1	1
marburg3	4	10	4	20	3-supervised	0
marburg4	4	10	4	20	3-clusterings	0
marburg5	4	10	4	20	3-clusterings	0
marburg6	4	10	4	20	5-clusterings	0
marburg7	4	14	5	11	1	1
marburg8	5	10	10	20	1	1
marburg9	5	10	5	20	1	1

Table 1: The parameter settings for the different runs: minimum and maximum sliding window sizes for different frame distances, the option for false alarm removal. In the column "#Classifiers", the number of classifiers and the type of the additional classifiers is given (clustering or supervised classification).

Run	Cuts		Gradual Transitions		Gradual Trans. Frame-based		All Transitions	
	Recall	**Prec.**	**Recall**	**Prec.**	**Recall**	**Prec.**	**Recall**	**Prec.**
marburg0	0.923	0.777	0.582	0.675	0.586	0.742	0.831	0.755
marburg1	0.906	0.802	0.587	0.667	0.586	0.745	0.820	0.772
marburg2	0.908	0.804	0.588	0.668	0.586	0.744	0.821	0.773
marburg3	0.893	0.810	0.587	0.664	0.586	0.743	0.810	0.777
marburg4	0.908	0.804	0.588	0.668	0.586	0.744	0.821	0.773
marburg5	0.908	0.804	0.588	0.668	0.586	0.744	0.821	0.773
marburg6	0.905	0.806	0.588	0.667	0.586	0.744	0.820	0.774
marburg7	0.902	0.801	0.583	0.670	0.586	0.742	0.816	0.772
marburg8	0.895	0.807	0.597	0.661	0.586	0.742	0.814	0.773
marburg9	0.902	0.806	0.588	0.666	0.587	0.744	0.817	0.774

Table 2: Recall and precision for the different runs, separated for cuts, gradual transitions, for gradual transitions on a frame basis, and for all transitions.

Run	Cuts	Gradual Transitions	Gradual Transitions Frame-based	All Transitions
marburg0	0.844	0.625	0.655	0.791
marburg1	0.851	0.624	0.656	0.795
Marburg2	0.853	0.625	0.656	0.796
Marburg3	0.849	0.623	0.655	0.793
Marburg4	0.853	0.625	0.656	0.796
Marburg5	0.853	0.625	0.656	0.796
Marburg6	0.853	0.625	0.656	0.796
Marburg7	0.849	0.623	0.655	0.793
Marburg8	0.849	0.627	0.655	0.793
Marburg9	0.851	0.625	0.656	0.795

Table 3: F1-measures for all runs, separated for cuts, gradual transitions, frame-based detection performance of gradual transitions, and all transitions.

3. Rushes Video Exploration

In this section, we describe our system prototype to explore the rushes material. The system consists of a summarization component and a retrieval component. All components have been integrated in our video content analysis software "Videana" which originally is aimed at supporting scientific movie studies.
The rushes videos are segmented on a sub-shot basis. Initially, shot segmentation is applied to the videos. In case of the rushes material, only cut detection is needed, since there are no complex transition effects present in these videos. Then, the shots are segmented further in order to obtain sub-shots with respect to events like speech, silence, camera motion and face appearances. This is motivated by the fact that users normally are only interested in parts of a rushes shot and that there are many sequences within a shot that contain redundant material. For example, consider an interview shot with a long silent period in the beginning due to camera adjustment. Thus, we believe that such sub-shots are the fundamental processing unit for the rushes material. In the following, a brief description is given for the several components of the proposed rushes system with respect to: feature extraction, sub-shot segmentation, rushes summarization and visualization, and finally with respect to retrieval of sub-shots.

3.1 Feature Extraction

At first, videos are segmented into shots using the cut detection approach described in section 2.

3.1.1 Camera Motion Features

Motion vectors embedded in MPEG videos are employed to compute camera motion at the granularity of P-frames, according to the approach presented in [5]. The following camera motion types are distinguished: pan, tilt and zoom.

3.1.2 Face Features

Frontal faces are detected in each video frame using the face detector provided by Intel's OpenCV library [www.intel.com/technology/computing/opencv]. The detector normally reports several detections at slightly different sizes and positions for one face. This number of detection hits is considered as a feature as well. To reduce the number of falsely detected face occurrences, detected faces are tracked across several frames. As a result of the face feature extraction process, we obtain face sequences across several frames, the position and size of each detected face, and the number of hits (which is related somehow to the probability that a frontal face is shown).

3.1.3 Keyframe Features

Each sub-shot is represented by a keyframe, for this purpose the frame from the middle of a sub-shot is chosen. From each of those keyframes, color and texture features are extracted. Each keyframe is divided into a number of local regions for feature extraction. Two color features and a texture feature [13] as suggested for the MPEG-7 standard [15] are extracted: the scalable color descriptor (SCD), the color structure descriptor (CSD) and edge histograms are extracted for each region. In addition, Gabor wavelet features are extracted for several orientations and frequencies (similar to the Gabor features used in [8]).

3.1.4 Audio Features

The following low-level features are extracted to analyze audio data: Energy, zero crossing rate (ZCR), mel-frequency cepstral coefficients (MFCCs), band periodicity, brightness and bandwidth, noise frame ratio and spectrum flux. These features are then evaluated in order to recognize the following semantic audio concepts (similar to the approach suggested in [12]): silence, speech, music, action (shooting, sirens, explosions, screams etc) and background (machinery, nature, background, environmental sounds). Furthermore, for each video a speaker indexing [19] is performed which outputs a speaker id for speech segments. The speaker indexing process is based on a clustering of all speech segments exceeding 2.5s in length. Speech segments are modeled by Gaussian Mixture Models (GMM), dissimilarity is measured via the Earth Mover's Distance (EMD), and the clustering partition selection criterion is the within-cluster dispersion [9].

3.2 Sub-Shot Segmentation

Sub-shot segmentation is based on the idea that the user is only interested in those parts of the shots which contain non-redundant, interesting content. First, video cut detection is performed as described in the previous section. Then, face sequences, camera motion and audio features are employed to segment these shots further into sub-shots. In particular, long sequences without any motion respectively long silent sequences are possible indicators for redundancy. The face, motion and audio features are not used at low-level but rely on a classification process of camera motion estimation approach suggested in [5], face sequence generation proposed in [8], respectively audio segmentation according to [12]. Camera motion types

of pan, tilt and zoom are distinguished and used for segmentation as well as the appearance of face sequences, while the features silence and speech are utilized for sub-shot segmentation with respect to audio information. For the purpose of sub-segmentation, the binary classification results for these features are utilized at a temporal granularity of one second. A new sub-shot begins (respectively the preceding sub-shot ends) when one of the feature classifications changes from 0 to 1 respectively from 1 to 0. For example, if a zoom event starts that might be of interest for the user, the related feature value should change from 0 to 1 and a new sub-shot is created.

3.3 Video Summarization and Visualization

An important issue in exploring the rushes material is the removal of redundant material. Since it is believed that it is hard to anticipate the user's interests in exploring rushes and that they normally change over time as well, an unsupervised clustering approach along with an appropriate visualization is suggested. The user can control which features are used for clustering and visualization. Sub-shots can be clustered according to several feature types:

1. Camera motion: pan, tilt, zoom;
2. Audio features: silence, music, speech, action, background;
3. Face information: number of faces and shot size;
4. Color and texture features;
5. Interview: information based on speaker analysis.

Based on the features selected by the user, a distance matrix expressing the similarity of sub-shots is generated. The visualization of the relationships between sub-shots based on their similarity requires a mapping of the high-dimensional feature space to a two- or three-dimensional space. Capturing the structure of these relationships using only linear projections is difficult. However, several approaches exist, aimed at reproducing nonlinear, high-dimensional data structures. These approaches either map objects into the lower-dimensional space analytically, such as classical multidimensional scaling (MDS) [14], or they try to optimize low-dimensional mappings so that the new distances in the low-dimensional space reflect the original distances in the higher-dimensional space, such as non-metric methods. The non-metric methods differ by the distance weighting scheme and the optimization algorithm used. The Sammon mapping [17] represents one such technique that performs non-metric MDS. In our prototype, we have used both classical MDS and the Sammon mapping for achieving low-dimensional projections of sub-shot relationships.

Before visualization can take place, the sub-shots are clustered using k-means where the number k of clusters can be defined by the user and should typically range between 10 and 100. Then, for each pair of clusters, the distance of their centers is computed and all these distances are used to create a similarity matrix. This matrix is then used for the purpose of visualization using either the Sammon mapping or classical MDS. The visualization allows the user to zoom into or out of parts of the visualization.

Via this clustering, similar sub-shots are grouped together and the user gets quickly an overview over the rushes material. The user can browse through these clusters, e.g. zoom into one cluster, and thus search efficiently for interesting sub-shots.

3.4 Retrieval of Rushes Sub-Shots

In the preprocessing stage, the videos have been analyzed and several low-level and high-level features have been extracted or computed automatically. The following features are computed as described above, and the user can start an automated search for sub-shots exhibiting one or a combination of these features:

1. Interview or not: Whether an interview is conducted in a sub-shot;
2. Shot size: Representing the camera distance;
3. No. of faces present in a sub-shot;
4. Camera motion: pan;
5. Camera motion: tilt;
6. Camera motion: zoom;
7. Query-by example: User can select an arbitrary frame region and search for it in the sub-shot database;
8. Silence;
9. Speech:
10. Music;
11. Action;
12. Background sound;
13. Speaker identity.

Relevant shots are retrieved and returned to the user ranked by the probability that they exhibit the requested feature.

3.5 Experimental Results

In the experiments, the TRECVID rushes video test data were used. First, the clustering and visualization method was tested for several k of the k-means clustering process. Furthermore, the visualization

techniques of classical MDS and Sammon mapping were compared with respect to their performance in organizing the rushes content. Finally, we conducted retrieval experiments for features presented in the previous subsection.

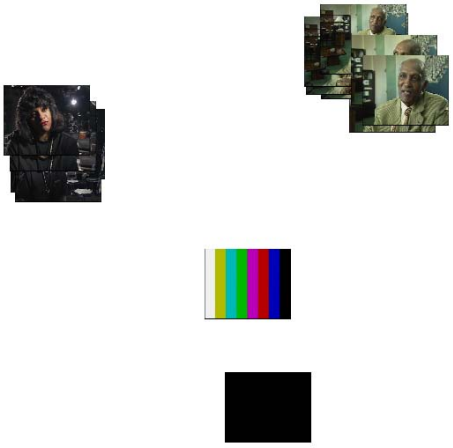

Figure 1: Screenshot of our visualization of the video franco85.mpg using the Sammon mapping and k=30.

Figure 2: Visualization of the same video using classical MDS and only 3 clusters.

Figures 1 and 2 exemplarily display two facts learned from the experiments regarding clustering and visualization:
First, it is shown how the user can efficiently and intuitively reduce the inherent redundancy by first specifying a large number of clusters (as in figure 1), which gives a general overview of a video's content. The likely over-segmentation can be handled in a second step by interactively decreasing the number of clusters to a visually more suitable number (as shown in figure 2). As can be seen from figure 3, our sub-shot clusters offer a high degree of purity necessary for reliable summarization.
Second, the examples show that both the classical MDS and the Sammon mapping technique are able to produce visually appealing, well-organized results. The actual choice of method for this specific task thus depends mainly on personal preferences.
We evaluated the performance of our retrieval component by using our graphical user interface depicted in figure 4. It returns the top-50 retrieval list, for which we computed the precision as performance index. The results for the best working individual features are given in table 4. It is interesting to look at the false positives therein: For music, 43% of the false positives include chirping of birds or blowing of a whistle, which is anyhow strongly related to the concept of music. For camera pan, tilt and zoom, 93%, 71% and 35%, respectively, of the falsely retrieved sub-shots show the movement of a big foreground object along the desired camera-movement axis, which is hard to distinguish even for humans.

Figure 3: A zoom into the lower right cluster of figure 2.

Feature	Top-50 precision
Audio: Music	0.580
Audio: Silence	1.000
Audio: Speech	0.940
Camera motion: Pan	0.720
Camera motion: Tilt	0.720
Camera motion: Zoom	0.420
No. of faces = 1	1.000

Table 4: Top-50 precision of the features in column 1 using the complete rushes test material.

These results are quite promising, though the retrieval performance for camera motion was lower than it was expected. The reason is that currently the rotation angle respectively the zoom factor was used to rank the sub-shots. However, these values are not directly related to the detector's confidence that there was camera motion.

Figure 4: Screenshot of the graphical user interface for the retrieval component. The retrieval list on the right side shows the result for the feature shot size = 30%.

4. Conclusions

In this paper, we have presented our experiments for two tasks of TRECVID 2006: shot boundary detection and the rushes exploration task. Our shot boundary detection approach consisted mainly of the last year's approach. This year's focus was to investigate the impact of different sliding window size ranges for cut detection and the possibility to build an ensemble of classifiers by extending the basic approach with unsupervised classifiers which rely only on different parameter settings. However, the experiments showed that the impact of different sliding window size ranges is negligible, and changing sliding window size ranges had nearly no impact on detection results. The tested unsupervised ensembles achieved only a slightly better performance. This is possibly due to the fact that the individual classifiers of an ensemble normally should exhibit a certain degree of independence in order to improve the overall performance. Experimental results indicated that this was not the case for the ensembles employed this year. Overall, the performance was as follows. For cut detection, a recall of about 90% and a precision of 80% were achieved, while for gradual transition detection a recall of about 60% and a precision of 67% were achieved.

The proposed system for rushes exploration was integrated into our video content analysis platform "Videana". It consists of a summarization and a retrieval component and works as follows. First, video cut detection is applied to the rushes material. Then, rushes shots are segmented further to obtain sub-shots with respect to events like speech, silence, camera motion and face appearances. Summarization is achieved by clustering those sub-shots using high-level audiovisual features (camera motion, face detection, shot size, speech, silence, music, action, background). The resulting clusters are visualized and the user can navigate interactively through these sub-shots to search the videos. Two visualization techniques were compared: Sammon mapping and classical MDS. The experiments demonstrated that both methods work equally well for our application. Furthermore, users can employ an automated search using the retrieval component with respect to the following features: number of faces, shot size, pan, tilt, zoom, silence, speech, music, action, background noise, and interview. In addition, a user can select arbitrary frame regions to search for similar regions in the rushes material (query-by-example). In the experiments, the top-50 precision was measured for several features. These experiments showed very good retrieval results for audio and face features, while our work with camera-motion features showed room for improvement, especially in the case of ranking in the retrieval list. Overall, the proposed system demonstrated its potential for efficient exploration of

rushes videos. In the future, it is planned to extend the retrieval component with the possibility to search for several additional high-level features. Finally, the retrieval performance should be further improved for selected features.

5. Acknowledgements

This work is financially supported by the Deutsche Forschungsgemeinschaft (SFB/FK 615, Teilprojekt MT).

6. References

1. Chang, C.-C. and Lin, C.-J.: LIBSVM: A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm

2. Dante, A. and Brookes, M.: Precise Real-Time Outlier Removal from Motion Vector Fields for 3D Reconstruction. In Proc. of IEEE International Conference on Image Processing, Vol. 1, Barcelona, Spain, 2003, pp. 393-396.

3. Ewerth, R. and Freisleben, B.: Improving Cut Detection Algorithms for MPEG Videos by GOP-Oriented Frame Difference Normalization. In Proc. of the 17th International Conference on Pattern Recognition, Vol. 2. Cambridge, United Kingdom, 2004, pp. 807-810.

4. Ewerth, R. and Freisleben, B.: Video Cut Detection without Thresholds., Proc. of 11th Int'l Workshop on Systems, Signals and Image Processing, Poznan, Poland, 2004, pp. 227-230.

5. Ewerth, R., Schwalb, M., Tessmann, P., and Freisleben, B.: Estimation of Arbitrary Camera Motion in MPEG Videos. In Proc. of the 17th International Conference on Pattern Recognition, Vol. 1. Cambridge, United Kingdom, 2004, pp. 512-515.

6. Ewerth, R., Beringer, C., Kopp, T., Niebergall, M., Stadelmann, T., and Freisleben, B. University of Marburg at TRECVID 2005: Shot Boundary Detection and Camera Motion Estimation Results. In Online Proceedings of TRECVID Conference Series 2005: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html

7. Ewerth, R. and Freisleben, B.: Self-Supervised Learning for Robust Video Indexing. In Proceedings of the IEEE International Conference on Multimedia & Expo, 2006, Toronto, Canada, 2006, (to appear).

8. Ewerth, R., Mühling, M., and Freisleben, B. Self-Supervised Learning of Face Appearances in TV Casts and Movies. In Proceedings of the 6th IEEE International Symposium on Multimedia 2006, San Diego, CA, USA, 2006, (to appear).

9. Jin, H., Kubala, F., and R. Schwartz, R.: Automatic Speaker Clustering. In Proc. of the DARPA Speech Recognition Workshop, 1997, Chantilly, Virginia, USA, pp. 108–111.

10. Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., and Duin, R. P. W.: Limits on the Majority Vote Accuracy in Classifier Fusion. In Pattern Analysis and Applications, Vol. 6, No. 1, Springer-Verlag, London, 2003, pp. 22-31.

11. Li, D., Sethi, I.: MPEG Developing Classes. http://www.cs.wayne.edu/~dil/research/mdc/docs

12. Lu, L., Zhang, H.-J., and Li, S. Z.: Content-based audio Classification and Segmentation by using Support Vector Machines. In Multimedia Systems 8, 2003, pp. 482-492.

13. Manjunath, B. S., Ohm, J.-R., Vasudevan, V. V., and Yamada, A.: Color and Texture Descriptors. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 6, 2001, pp. 703-715.

14. Mardia, K.: Multivariate Analysis. Academic Press, 1979.

15. MPEG-7: ISO/IEC 15938-2 Information Technology - Multimedia Content Description Interface: ISO/IEC 15938, 2002.

16. Petersohn, C.: Fraunhofer HHI at TRECVID 2004: Shot Boundary Detection System. In TREC Video Retrieval Evaluation Online Proceeding s, TRECVID, 2004.URL: www-nlpir.nist.gov/projects/tvpubs/tvpapers04

17. Sammon, J.: A Nonlinear Mapping for Data Structure analysis. In IEEE Transactions on Computers, 18(5), 1969, pp. 401-409.

18. Shen, K., Delp, E. J.: A Fast Algorithm for Video Parsing Using MPEG Compressed Sequences. In Proc. of IEEE International Conference on Image Processing, 1995, Washington, DC, 1995, pp. 252-255.

19. Stadelmann, T. and Freisleben, B.: Fast and Robust Speaker Clustering Using the Earth Mover's Distance and MIXMAX Models. In Proceedings of the 31st International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 1, Toulouse, France, 2006, pp. 989-992.

20. Tahaghoghi, S. M. M., Thom, J. A., Williams, H. E., and Volkmer, T.: Video Cut Detection using Frame Windows. In Proceedings of the Twenty-Eighth

Australasian Computer Science Conference 2005, Vol. 38, Newcastle, NSW, Australia, 2005, pp.193-199.

21. Truong, B.-T. and Venkatesh, S.: New Enhancements to Cut, Fade and Dissolve Detection. In Proc. of ACM International Conference on Multimedia, Los Angeles, USA, 2000, pp. 219-227.

22. Viola, P. and Jones, M. J.: Robust Real-Time Face Detection. In International Journal of Computer Vision 57(2), Kluwer Academic Publishers, Netherlands, 2004, pp. 137-154.

23. Yeo, B. and Liu, B.: Rapid Scene Analysis on Compressed Video. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 5, No. 6, 1995, pp. 533-544.