SBD, Search and Rushes: TRECVid 2006 experiments at URJC

Pablo Toharia [1], Oscar D. Robles[1], Ángel Rodríguez[2] and Luis Pastor[1]

[1]Dept. de Arquitectura y Tecnología de Computadores e Inteligencia Artificial,
U. Rey Juan Carlos (URJC). C/ Tulipán, s/n. 28933 Móstoles. Madrid. Spain.
{pablo.toharia,oscardavid.robles,luis.pastor}@urjc.es

[2]Dept. de Tecnología Fotónica.
U. Politécnica de Madrid (UPM). Campus de Montegancedo s/n.
28660 Boadilla del Monte. Madrid. Spain
arodri@dtf.fi.upm.es

ABSTRACT
This paper describes the work performed by the URJC team in TRECVid 2006. Three tasks have been tackled: shot boundary detection, search and rushes (in collaboration with DCU group). We present an analysis of the results achieved in the official TRECVid tests for each one of these tasks.

KEY WORDS
Shot Segmentation, CBIR primitives, Video Retrieval, Rushes processing

1 Introduction

This paper presents the work performed by the URJC team for TRECVid 2006. Continuing with the line of work of past editions, the URJC team contributes with new runs on the shot boundary detection task, presenting a deeper study about the behavior of some low-level detail primitives and a comparison between the results of last year and the ones of current session.

But, apart from that, new tasks have been tackled in this year opening new research challenges under the TRECVid framework. We have tested in the search task some of the previously developed low-level features in order to check if this type of primitives could obtain noticeable results in some specific topics of the available set. This hypothesis arises from the idea that some high-level concepts are very related to low-level features and simple solutions could give good results in those cases.

Furthermore, the third action carried out this year has been the experimental rushes task, developed in collaboration with the Dublin City University team (DCU). Combining the experience of both teams in processing low-level features (URJC) and high-level features (DCU), some tests have been performed in order to process and discard useless shots in non edited video sequences.

The notation for identifying the runs submitted for evaluation to TRECVid 2006 is the same that the one used in the submission, but removing the prefix that identifies the team, in this case, URJC.

All the tools involved in the developed software are free distribution tools, like vs. 2.6.9 Linux operating system, vs. 4.0 of the GCC GNU compiler [1], CVS version of the FFmpeg video stream decoder [2] and vs. 2.6.21 of the LIBXML2 library for processing XML files [3].

The contents of this paper may be broken down into a description of the proposed shape based shot extraction technique (Section 2), continuing with a description of the way we perform high level semantic search using color and shape information (Section 3), followed by the presentation of the rush processing technique implemented 4.

2 Shot boundary detection

2.1 Task Description

The work presented here is based on URJC team experiments fusing color and shape primitives under the TRECVid 2005 shot boundary detection task. A deeper analysis of the tested features is provided behind a comparison among 2005 and 2006 TRECVid results.

The selection of a color primitive for its combination with the shape primitive has been influenced by our previous experience on shot segmentation using standard color histograms, as well as global multiresolution histograms computed over the analysis coefficients of the frame's wavelet transform. Actually, the best results reported during TRECVid04 were obtained quantifying standard histograms to 16 bins [4]. The shape primitive is based on color Zernike invariants, and the description of fusing shape and color primitives can be found in [5].

We have introduced a certainty parameter th in the shape primitive in order to improve the precision measure keeping recall in high values. This confidence is measured based on the difference computed between the current frame and a window of frames and the dynamic threshold which allows to decide whether a cut was found or not [5]. The global certainty th is computed averaging each primitive's confidence. This time three global certainty values have been tried.

Ten runs have been tested, submitted with the following labels:

1. `hisc16_1`: color primitive based on histograms quantified to 16 classes with redistribution of boundary values.
2. `zer3_1`: shape primitive based on 3$^{\text{rd}}$ order Zernike invariants.
3. `zer3_th15_1`: same case as 2 but introducing a certainty value $th = 15$.
4. `zer3_th25_1`: same case as 2 but $th = 25$.
5. `zer3_th35_1`: same case as 2 but $th = 35$.
6. `zer3_and_hisc16_1`: AND combination of color and shape using differences of Zernike moment invariants up to third order polynomials and differences of color histograms, quantified to 16 classes or bins, with redistribution of boundary values.
7. `zer3_or_hisc16_1`: same case as 6 but changing AND operator by OR operator.

Method	TRECVid 2005		TRECVid 2006	
	Recall	Precision	Recall	Precision
hisc16_1	0.903	0.757	0.860	0.696
zer3_1	0.891	0.482	0.854	0.406
zer3_and_hisc16_1	0.811	0.866	0.735	0.802
zer3_or_hisc16_1	0.950	0.453	0.924	0.383
zer3_or_hisc16_th15_1	0.935	0.501	0.908	0.430
zer3_or_hisc16_th25_1	0.920	0.535	0.895	0.465
zer3_or_hisc16_th35_1	0.894	0.567	0.871	0.500
zer3_th15_1	0.865	0.528	0.824	0.449
zer3_th25_1	0.845	0.560	0.803	0.480
zer3_th35_1	0.822	0.594	0.780	0.514

Table 1: Comparison among precision and recall values obtained for cut detection evaluated on TRECVid 2006 and run over TRECVid 2005 dataset.

8. `zer3_or_hisc16_th15_1`: same case as 7 but introducing a certainty value $th = 15$, ranged in the interval [0,100].

9. `zer3_or_hisc16_th25_1`: same case as 8 with $th = 25$.

10. `zer3_or_hisc16_th35_1`: same case as 8 with $th = 35$.

As mentioned, these experiments allow us to deeply study the primitives comparing the results with the ones obtained in TRECVid 2005. Furthermore, the primitive labelled as `hisc16_1` will make possible to compare them with TRECVid 2004's.

2.2 TRECVid Results

Table 1 shows recall and precision values as returned by the TRECVid 2006 team for the shot boundary extraction task. It must be noticed how, as in previous TRECVid editions, our system does not consider very short dissolves as cuts. This is the reason why the table does not show global results as evaluated by TRECVid team.

It can be seen how the histogram-based primitive obtains very interesting results in terms of recall and precision. Since this conclusion can be stated for third year, it confirms the power and usefulness of this primitive. On the other hand, Zernike based primitive obtains similar results in terms of recall but behaves worse in terms of precision. As in TRECVid 2005, its results have been improved with the certainty measure. While recall value only drop down around a 9%, precision increases a 21%.

An examination of the combination of primitives reveals that the OR technique improves the recall but not the precision. On one hand, the number of true positives is higher, since each primitive contributes with true positives not detected by the other one. On the other hand, each primitive is adding to the combination a number of false positives and false negatives not considered by the other one. Again, an improvement of the precision value has been achieved by introducing the certainty value. In this case, recall value only falls a 6% while precision increases a 24%.

By contrast, the AND combination reflects the true positives that both primitives have detected. Recall value is lower since there are some true positives detected by one of the primitives but not by the other one, and also a few of them not detected at all. In this case the precision is greater than the ones achieved by the primitives alone since using this technique the number of false positives and negatives is heavily reduced to only those detected by both primitives. Results obtained in this case show that further work should be done in this type of combination, since its precision is the highest and the balance with the recall value is quite interesting.

Table 1 also shows results for these year's tests run over TRECVid 2005 dataset. In this case, it can be seen how they follow the same tendency explained for TRECVid 2006 results. Recall values are between a 3 and a 6% higher and precision is between a 12 and a 16% lower, something explained by the differences in the dataset.

But two primitives do not follow the general tendency explained. On one hand, precision showed by primitive `hisc16_1` only drops down a 9%. On the other hand, the precision obtained by the AND combination falls only an 8%. These facts remark the power of both primitives based on histograms and the AND combination.

Results obtained with the primitive `hisc16_1` for TRECVid 2004 were a recall of 0.868 and a precision of 0.855. It means a maximum variation of a 4% in the recall while goes to around 18% for the precision. This variation in the precision can be explained by the different edition effects introduced by the inclusion of signal from China.

Execution times range from 2653 to 232819 seconds. It means to wait for more than 64 hours to have the segmentation of one video completed. This is due to the complexity of computing Zernike invariants. We have been working on different approaches that achieve a reduction on the execution times using shared-memory multiprocessors or a cluster of PCs [6].

It can be concluded that this comparison has contributed to show the consistency of our experiments. Our low-level features show independence from the dataset, although it has to be said that different signal containing other types of programs and coming from other cultures should be tested in order to state the real limits of these primitives.

3 Search Task

3.1 Task Description

Six runs have been tested, divided in two alternative set of experiments: fully automatic and interactive searches.

A graphical interface has been developed in order to assist users in their queries and result refinements in the interactive runs. Figure 1 shows some captions of the user interface and an example of the steps to follow so as to refine a query. The interface allows a user to select a reference set of images to guide the query. Once the reference set has been chosen, a search criteria is selected among the available implemented primitives so the query can then be launched. The application returns an ordered list of images sorted by their average similarity with respect to the images belonging to the reference set. If some of the retrieved images are considered useful for refining the query, the user can introduce them in the reference set. Another option available at this moment is to put aside these images in another location of the interface to be considered for a later search or refinement step. Before doing a new search, the user may remove some of the images of the reference set if the results provided are not considered as successful. When all input parameters are ready for a new iteration, the user will use the search button, obtaining new results. The user can do as many iterations as he wants, considering the images returned by the retrieval system at each moment and inserting or removing a subset of them from the reference set. An update of the reference set implies a new computation of the low-level features used as search parameters in the following iteration.

The interface implements the following primitives:

- `hin`: Multiresolution histograms computed over the analysis coefficients of the frame's wavelet transform [4].
- `had`: Multiresolution histograms computed over the analysis and diagonal detail coefficients of the frame's wavelet transform [4].
- `en2`: Multiresolution energies computed over the analysis coefficients of the frame's wavelet transform [7].
- `zer`: Multiresolution shape primitive based on Zernike invariants. It is based on the Zernike primitive presented at TRECVid 2005 [5], although the invariants are now computed over the analysis and detail coefficients of the frame's wavelet transform.

These primitives may be also fused with the shape primitive in order to make queries combining shape and color information. The user can choose a set of query images different from the one provided by the retrieval system, specifying an URL to supply more suitable examples if the retrieval system is not able to do it.

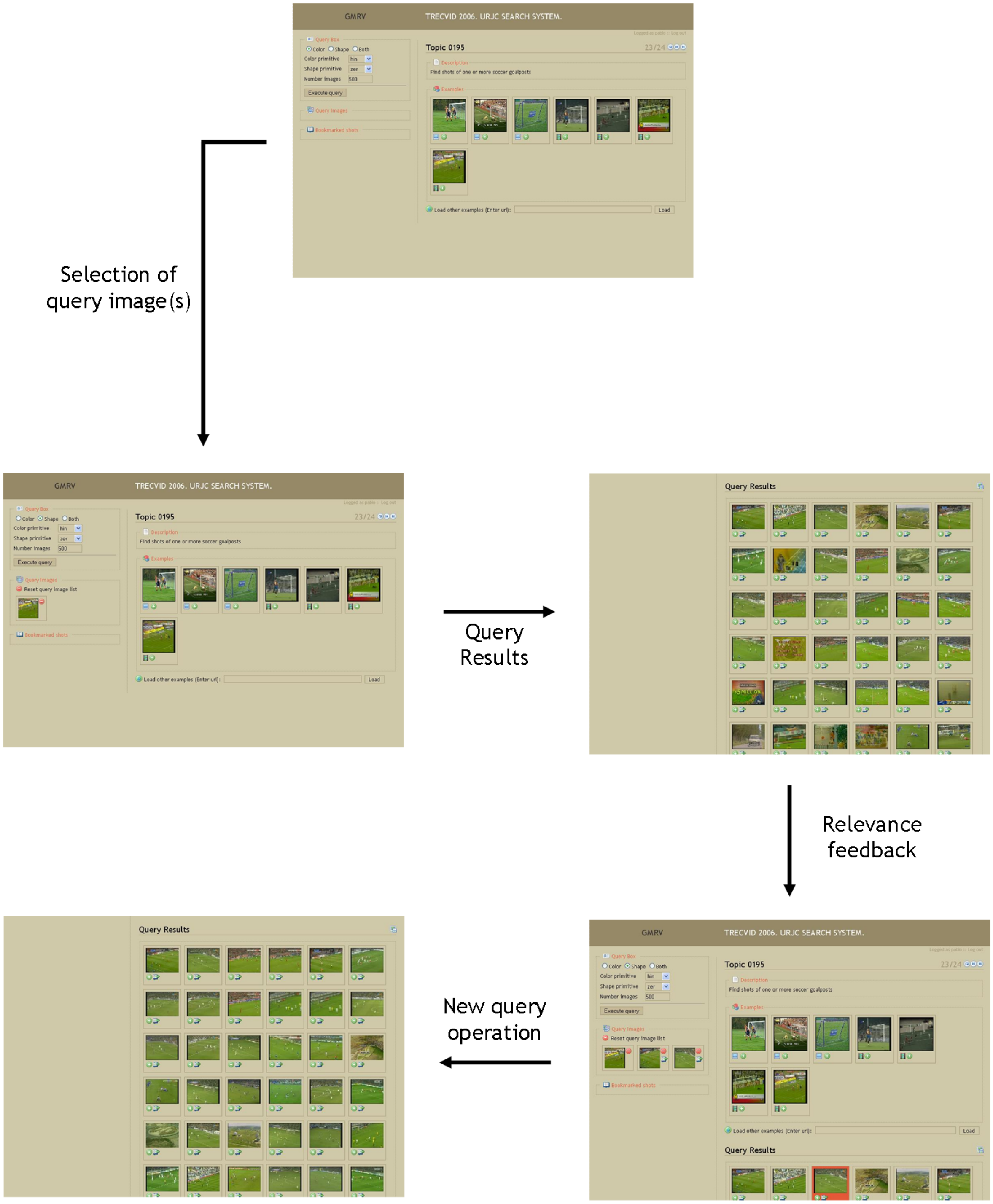
In the case of fully automatic searches, tests have been done for each topic performing the query over the available data independently and averaging the results to show the definitive one.

3.2 TRECVid Results

As mentioned above, two different tests have been performed:

- Fully automatic searches (labelled with prefix FA).
- Interactive searches.

The users selected for interactive searches were completely unaware of any topic regarding CBIR systems. In general, users did not find enough time to plan appropriate refinement strategies. After tracking their sessions, it can be said that a few users were not able to follow appropriate strategies at all, due to probably the novelty of the retrieval process for them. One interesting case to mention is the users conservative behavior. In this way, it seemed to be hard for users to remove any of the images used to begin the very



Topic 0195

Find shots of one or more soccer goalposts

Color primitive: hin

Shape primitive: zer

Number Images: 500

Figure 1: Example of user interaction with the search interface.

first iteration from the reference set, even when it was clear that the results they were providing were quite poor.

Following the notation of the submissions, the tested runs can be identified as:

- FA_zer: shape primitive based on 3$^{\text{rd}}$ order Zernike moment invariants.

- FA_hrn: color primitive based on multiresolution histograms.

- FA_zmr: shape primitive based on multiresolution 3$^{\text{rd}}$ Zernike moment invariants.

- FA_hrnzer: fusion of shape and color primitives (zer and hrn) equally weighting each primitive.

- inter1: Interactive search fusing shape and color primitives (zer and hin) equally weighting each primitive.

- inter2: Interactive search selecting primitives had and zer and equally weighting each primitive.

The interactive runs differ in the primitives available to perform queries: zer and hin for inter1, and zer and had for inter2. The topics have been assigned to different users and a total of 14 people have been involved in the experiments.

Table 2 shows the results evaluated by the TRECVid team for each one of these runs. First of all, it must be noticed that we have used low-level primitives without any high-level information. It explains the low precision values obtained, with the exception of topic 195. This topic consists on finding shots with one or more soccer goalposts. The restricted domain in which soccer goalposts can be found has helped our low-level primitives to obtain a higher precision.

Each user has performed three searches at most. As mentioned before, the lack of experience in this problem has reduced the possibilities to obtain better results.

Topic 188 has been a clear example of users conservative behavior. In our system only a few of the very first images available to begin the first iteration of the query were able to produce relevant results. After completing the first iteration without finding relevant results, users tend to add images to the reference set not removing the ones previously introduced. This action reduces the number of new relevant images that potentially can appear in the following iteration

Apart from that, in spite of their lack of experience, some users did not used all the available time, so the time to familiarize with the interface and the posed search problem has been insufficient to obtain better results in comparison with those achieved by the fully automatic runs.

This is a fact to consider in the future: how to design interactive tools to guide inexperienced users in the searches, suggesting changes when some attempts have been made and unsatisfactory or repetitive results are returned by the retrieval system. In addition, users impatience is another question to take into account when designing these tools. Perhaps it should be analyzed from a psychological point of view, since sometimes users may prefer a lower precision if a higher one means to spend more time in front of the system.

Topic number	Method					
	FA_zer	FA_hrn	FA_zmr	FA_hrnzer	inter1	inter2
173	0.000	0.000	0.000	0.001	0.001	0.007
174	0.010	0.004	0.020	0.007	0.013	0.006
175	0.001	0.001	0.001	0.002	0.008	0.006
176	0.000	0.001	0.000	0.000	0.001	0.000
177	0.002	0.015	0.004	0.013	0.016	0.011
178	0.000	0.002	0.001	0.002	0.000	0.001
179	0.001	0.000	0.001	0.000	0.008	0.008
180	0.000	0.001	0.002	0.002	0.001	0.009
181	0.000	0.000	0.000	0.001	0.000	0.000
182	0.002	0.000	0.009	0.002	0.004	0.007
183	0.001	0.000	0.003	0.002	0.004	0.015
184	0.000	0.002	0.001	0.001	0.007	0.003
185	0.000	0.000	0.000	0.000	0.000	0.001
186	0.001	0.000	0.002	0.001	0.003	0.001
187	0.003	0.000	0.006	0.000	0.021	0.013
188	0.001	0.001	0.001	0.000	0.008	0.007
189	0.000	0.000	0.001	0.001	0.004	0.016
190	0.000	0.000	0.000	0.001	0.003	0.000
191	0.001	0.006	0.001	0.005	0.006	0.001
192	0.004	0.000	0.001	0.000	0.038	0.001
193	0.000	0.000	0.000	0.000	0.000	0.002
194	0.000	0.000	0.001	0.000	0.025	0.000
195	0.069	0.000	0.166	0.001	0.159	0.151
196	0.006	0.000	0.016	0.001	0.024	0.006

Table 2: Average precision for search task evaluated by TRECVid 2006.

4 Rushes

4.1 Task Description

Rushes were initially analyzed in order to structure raw footage into “events”. These will be similar to shots but are not based on detection of shot transitions and keyframe selection. Keyframes are now detected as frames which are significantly different from the previous (and subsequent) keyframes. For these event frames the 39 TRECVid 2006 features are then identified and a retrieval tool has been built to allow filtering and keyframe browsing based on these 39 features. Three stages can be identified in this process:

1. A kind of shot boundary detection.
2. Filtering of keyframes using previous known useless video patterns.
3. Shot classification using SVM classifiers.

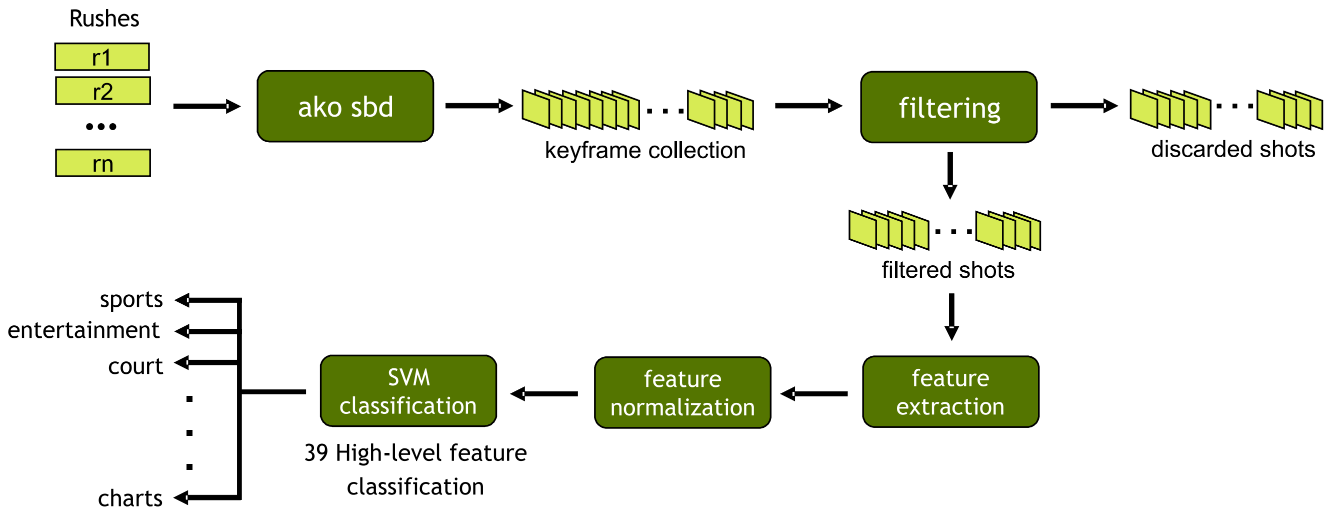


Figure 2: Overall scheme of the rushes extraction process.

Fig. 2 depicts the whole rushes extraction process. The following paragraphs describe each one of these components.

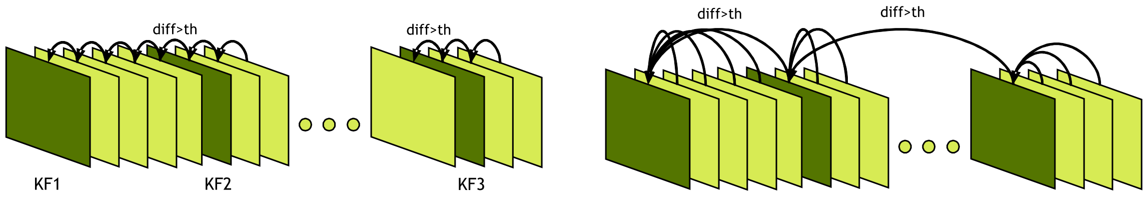
4.1.1 A Kind of Shot Boundary Detection

In almost every system dealing with video data a temporal segmentation stage is performed so as to be able to extract the content by characterizing its independent but meaningful parts. This process normally aims to divide the video streams into their smallest semantic units, i.e the shots. In regular video content, as can be the case of news data used in TRECVid or any other regular TV signal, the action occurring is variable. In most cases it can be noticed there are quite a few shot transitions, which can be both hard cuts and gradual transitions.

When dealing with video rushes the process to structure the video is not so obvious. Video rushes normally contain footage which is highly redundant but, what is more important to this point, they have a small number of camera operations and no post-production effects. These characteristics make shots to be very long and difficult to detect, since there are sometimes many different events within a shot, or the camera operation will not stop at all. Taking all this into account it can be assumed that if a regular shot detection approach is used to divide the streams, two things will happen: the first one is that detected shots will be very long. The second one, also as a consequence of the first one, is that normally quite a few real events will be missed. The conclusion is that conventional video shot contents will be be poorly characterized when taken from rushes video.

All of this leads us to propose a specific approach that suits this particular type of video. Instead of using the traditional way of performing shot boundary detection (3(a)), based on computing differences between sets of consecutive frames, a new approach has been developed (3(b)) . Differences are now computed between each frame and the previous isolated keyframe rather than on the previous frame, focussing then on events happening and not in local changes leading to shot transitions.

This strategy will make shot the detection threshold more sensitive to small changes in the scene as well as detecting regular shot transitions. This approach to event detection can help to extract more precise



(a) Traditional approach.

(b) Event detection based approach for rushes.

Figure 3: Two alternative approaches for keyframe extraction.

information about the content of rushes as well as being used to detect things happening during a shot, i.e. what we are going to call events from now on. For each event a representative keyframe will be extracted. In this work we have chosen the frame that triggers the event detection as the keyframe.

4.1.2 Filter of Known Useless Video Patterns

After computing the shot boundary detection process the result is a set of "shots" and their corresponding keyframes. It is very common when working with rushes that the contents of quite a few shots are completely useless. Examples of these are calibration shots showing a template or shots containing different artefacts produced by an analog camera start or stop operations. Taking this fact into account it has been considered that filtering this data out can make the following stages easier and more effective.

A simple way to achieve this filtering is to take a matching-based approach. As this system is meant to be user-interactive a semi-automatic filtering algorithm has been designed. At a first stage low-level features from all the keyframes are extracted. Then, if the user wants to apply the cleaning process he can select some seed keyframes in order to filter out similar shots, picking from the keyframe set some seeds (normally it is enough with 3 or 4) that he considers not useful. Then a comparison operation is performed between each seed and the whole keyframe data set. Each comparison result is then sorted and the most similar key-frames to the seeded ones are candidates for being discarded.

The number of filtered out shots can be modified depending on a threshold that the user can select. This rejection operation can be done within an interactive time so the user can adjust the threshold visually.

Features used for comparison are low-level features. We specifically selected one of our previously developed features based on Multiresolution Energy [7].

Figure 4 depicts the described filtering process.

Even though this is a very simple approach it works quite well for the rushes data and it can also be used in a very intuitive way and, above all, it is an interactive process.

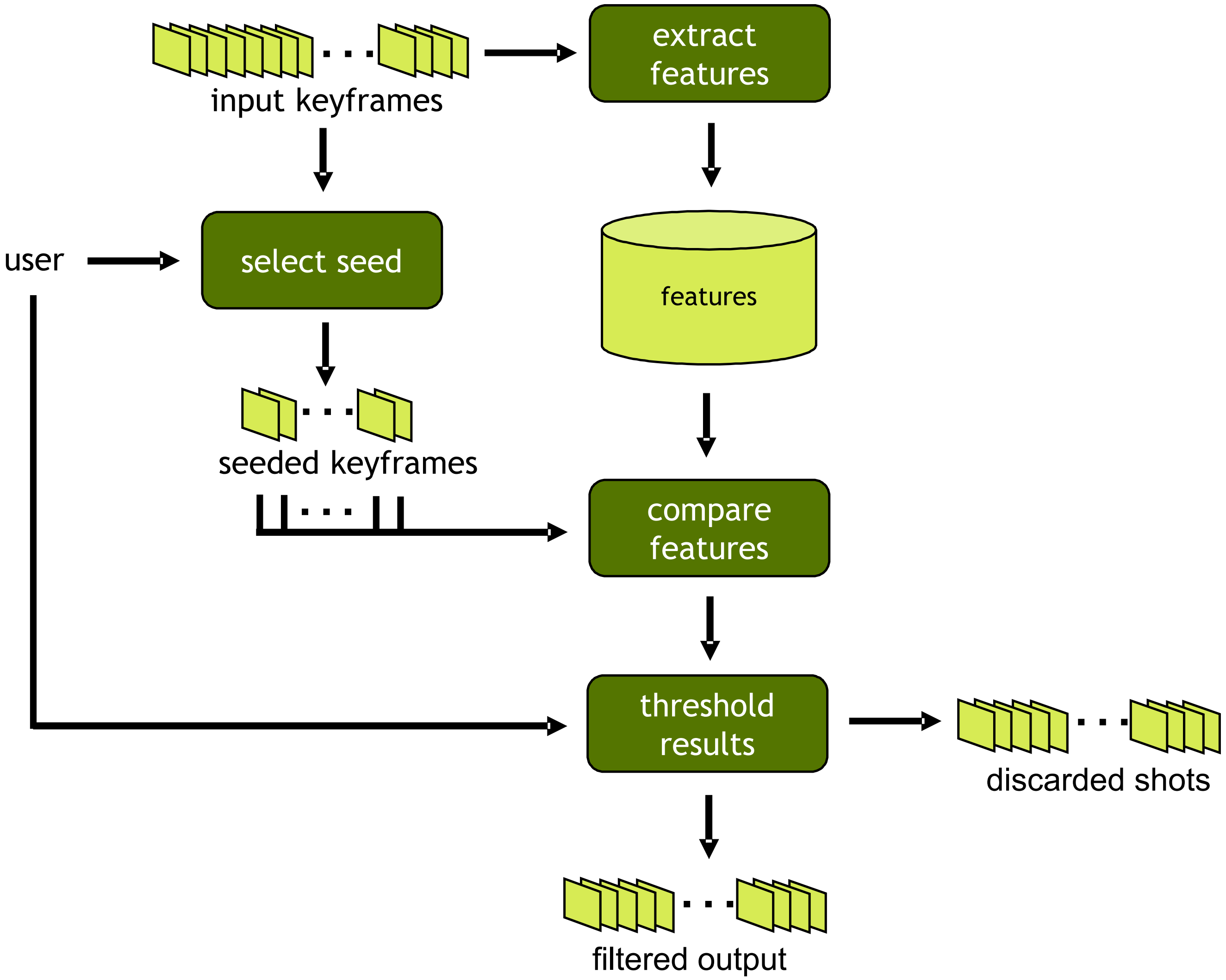


Figure 4: Scheme of the filtering process.

4.1.3 Shot classification using SVM

The next stage of our rushes work has focused on extracting high-level information from previously detected events. Thanks to the collaboration with the DCU team, their SVM-based systems for the TRECVid 2006 high-level feature detection task was used throughout our rushes shot collection. In this manner we have not only low-level information but some high-level information as well. This high-level information comes from the 39 features listed in the high-level feature detection task.

As mentioned above, the SVM classifier has been trained for high-level feature detection task, i.e. the data used was mainly TV news content. In fact, the 39 SVMs were trained using the development set from TRECVid 2006 TV news collection. This is a fact to be taken into account since rushes data are far from being similar to news content. This is also the reason why results can not be as good as expected. If this approach is found to be interesting, perhaps in the future it can be worth to re-training the SVMs using data with contents more similar to the rushes.

In order to train the classifier, the DCU team extracted 7 different low-level features using the Ace-

Media Toolbox from the AceMedia Project [8]. Those features included color, shape and texture based descriptors. In order to train the system all the features were linked together in a unique bitstream for each keyframe extracted from the shots in the development data set.

At this stage it should be taken into account that SVM systems need training values comprised between -1 and 1. It also should be noticed that the range of each descriptor value is variable and dependant on the nature of the feature itself. Both facts mean that a normalization phase must be applied. A standard normalization scheme has been applied, using each feature's absolute maximum value. This process ensures values are in the correct range.

In order to construct the training data set each of the previously detected shots were classified manually as a positive or negative example on each of the 39 features. Then, this manual classification was used as the input training data for the SVM. At this point it has to be stated that in fact 39 different SVM models have been trained, i.e. one for each concept.

Once the SVM has been trained it can be used to label each shot with the concepts detected. This means that the 7 features used for training have to be extracted from the detected events as well, and once concatenated can be used as the input for the 39 classifiers. This will output a certainty value for each shot and each concept, which means that using a threshold over this data a concept labeling is obtained for each shot.

This high-level information can be very useful for both browsing and retrieval stages. It could be combined with the low-level features as well to improve the performance.

The SVM implementation used at this stage was SVM^{light} [9, 10].

Acknowledgments

This work has been partially funded by the Spanish Ministry of Education and Science (grant TIC-2003-08933-C02), Government of the Community of Madrid (grant S-0505/DPI/0235) and EC (contract FP6-027026, K-Space). We are grateful to the AceMedia Project FP6-001765 for use of the AceToolbox.

References

[1] GNU. The GNU operating system. Web, 2006. www.gnu.org.

[2] Sourceforge. FFMPEG multimedia system. Web, 2006. http://ffmpeg.sourceforge.net/index.php.

[3] Gnome Project. Gnome XML C parser and toolkit. Web, 2006. www.xmlsoft.org.

[4] Oscar D. Robles, Pablo Toharia, Angel Rodríguez, and Luis Pastor. Towards a content-based video retrieval system using wavelet-based signatures. In M. H. Hamza, editor, *7th IASTED International Conference on Computer Graphics and Imaging - CGIM 2004*, pages 344–349, Kauai, Hawaii, USA, August 2004. IASTED, ACTA Press. ISBN: 0-88986-418-7, ISSN:1482-7905.

[5] Pablo Toharia, Oscar D. Robles, Ángel Rodríguez, and Luis Pastor. Combining shape and color for automatic video cut detection. In *Proc. of the TRECVID 2005 Workshop*, pages 336–345, Gaithersburg, Md, December 2005.

[6] Pablo Toharia, Oscar David Robles, José Luis Bosque, and Angel Rodríguez. Video shot extraction on parallel architectures. In M. Guo et al., editors, *Proc. on International Symposium on Parallel and Distributed Processing and Applications (ISPA 2006)*, volume 4330 of *Lecture Notes on Computer Science*, pages 869–883, Sorrento, Italy, December 2006. Springer Verlag.

[7] Angel Rodríguez, Oscar D. Robles, and Luis Pastor. New features for Content-Based Image Retrieval using wavelets. In Fernando Muge, Rogério Caldas Pinto, and Moisés Piedade, editors, *V Ibero-american Simposium on Pattern Recognition, SIARP 2000*, pages 517–528, Lisbon, Portugal, September 2000. ISBN 972-97711-1-1.

[8] Noel E. O'Connor, Edward Cooke, Herve Le Borgne, Michael Blighe, and Tomasz Adamek. The acetoolbox: Low-level audiovisual feature extraction for retrieval and classification. In *2nd European Workshop for the Integration of Knowledge, Semantics and Digital Media Technology, EWIMT 2005*, pages 55–60, London, UK, November 2005. ISBN: 0-86341-595-4, http://www.acemedia.org/aceMedia/files/document/wp7/2005/ewimt05-dcu.pd%f.

[9] Thorsten Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 41–56. MIT-Press, 1999.

[10] Thorsten Joachims. SVM^{light} support vector machine. Web, 2004. Retrieved october 24, 2006, from source, Developed at University of Dortmund, Informatik, AI-Unit Collaborative Research Center on 'Complexity Reduction in Multivariate Data' (SFB475), http://svmlight.joachims.org/.