2010 Second International Conferences on Advances in Multimedia

Video Scene Detection Based On Recurring Motion Patterns

Manfred del Fabro
Institute of Information Technology (ITEC)
Klagenfurt University
Klagenfurt, Austria
manfred@itec.uni-klu.ac.at

Laszlo Böszörmenyi
Institute of Information Technology (ITEC)
Klagenfurt University
Klagenfurt, Austria
laszlo@itec.uni-klu.ac.at

***Abstract*—We present an algorithm for video scene detection based on the identification of recurring motion patterns within a video stream. The motion information is extracted in the compressed domain of H.264/AVC videos, no full decoding of the video stream is needed. Based on the motion information our algorithm identifies sequences of adjacent frames with similar motion. Throughout all identified motion sequences we are searching for recurring patterns. The pattern recurring most often is used for the segmentation of the video stream into scenes. The evaluation shows promising results.**

***Keywords*-Non-Sequential Multimedia; Video Segmentation; Video Scene Detection**

I. Introduction

We are experiencing a shift in the video usage today. Sequential playback of video streams from the beginning to the end is not the only usage pattern anymore. For professional use of videos often only a fraction of the available material is needed. Think of medical doctors looking for specific pathological scenes in a surgery, of traffic managers looking for traffic jams in a video surveillance system for public motorways or of sports trainers wishing to compare the competitions of different athletes - only to mention a few examples. Usage is changing strongly in private usages as well. For many public events a lot of videos are captured from different people and from many vantage points in different qualities. It is a real challenge to gain relevant information out of this mass of data.

During the recent years many efforts in the field of video abstraction have been made to tackle this problem [8]. Many solutions for the selection of representing key frames or for the creation of video summaries have been proposed. Nearly all of them perform shot boundary detection. Thus, this problem can be regarded as well solved. The limitation of shot based approaches is that shots are normally rather short clips, which only contain few information. In most cases semantically meaningful information is spread across several coherent shots. The challenge is to detect such high-level semantic scenes.

We present an approach based on the motion information of the video stream. For the extraction of the motion information we use an algorithm that works in the compressed domain, therefore, no full decoding of the video stream is needed. We observe not only the motion within one frame, but also how motion is distributed across several frames. If there are adjacent frames that have the same motion direction, these frames form a coherent motion sequence. We try to find recurring patterns of motion sequences in the video stream and decompose it into scenes that correspond to such patterns. In contrast to typical video segmentation approaches we do not define a process with one static result. We rather propose many different, logical segmentations for each video, according to the hierarchical video representation presented in [4], where an indexing of videos on scene-, shot- and frame-level is suggested.

Automatically detected scenes can serve for different purposes. The hierarchical video representation can be used as table of contents for video browsing applications or as basis for video summarization approaches. Another possibility is the implementation of an interactive tool for the annotation of video content. Our algorithm does not extract any semantic information, but the suggested high-level scenes can be easily annotated by users. A self-organizing video delivery system [7] can also take advantage of our scene detection approach. By transmitting only those units that are needed for particular video compositions, the network load could be reduced. By predicting places, where certain units are supposed to become popular, these units can be replicated to proxies that are near to those places. Thus, startup delay can be reduced.

This paper is organized as follows. In Section II we present the related work, in Section III we introduce our algorithm for identifying recurring motion patterns within video streams. Section IV shows the evaluation results and in Section V we conclude this paper with some thoughts for future activities.

II. Related Work

An approach for finding recurring visual sequences in live TV broadcasts is presented in [1]. The authors use an edge feature for comparing video clips of a 24 hour live TV broadcast. The purpose is to find and document recurring commercials throughout the TV program of one day. In contrast to that approach, our algorithm does not try to find

978-0-7695-4068-9/10 $26.00 © 2010 IEEE
DOI 10.1109/MMEDIA.2010.26

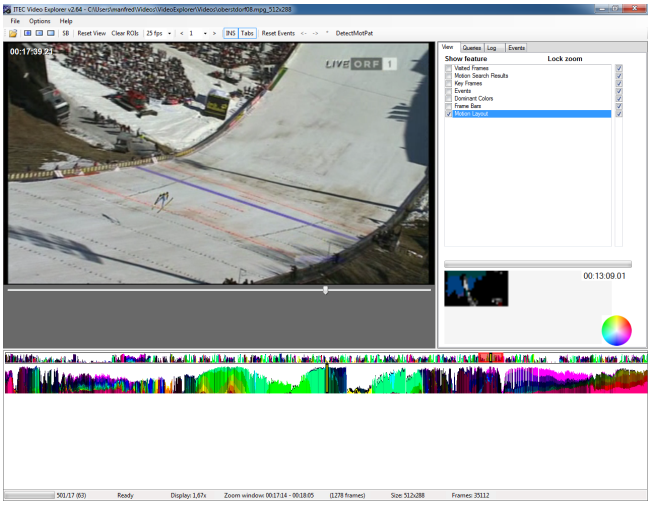

Figure 1. The Video Explorer showing the motion based navigation index for a video of a ski jumping competition.

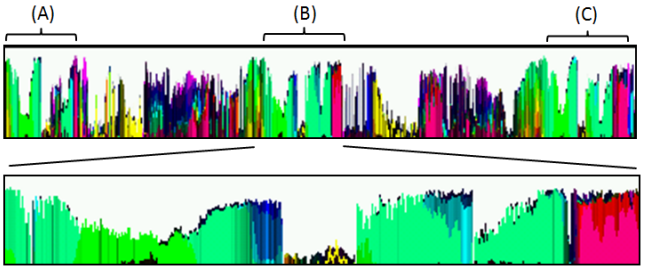

Figure 2. Visualization of the motion sequences that occur in a ski jumping video during the jumps of three athletes (A), (B), and (C). The lower bar shows an amplification of the motion sequences during the jump of athlete (B).

recurring scenes that are identical, but scenes with similar motion.

In [2] a shot detection algorithm is presented that calculates an activity level for each frame and segments the video stream into shots that contain high activity, because it is assumed that these parts are the most interesting ones within a video. As already mentioned, scenes created in such a way are rather short. With our approach we try to identify longer scenes, consisting of shots that semantically belong together.

Another solution for motion based detection of identical events is introduced in [9]. Shots that correspond to one semantic event are grouped based on the camera motion during the capturing process. Videos of an American Football game that have been recorded from different camera positions are analyzed and shots with similar camera motion according to the position of the camera are grouped to one event. Thus it is possible to retrieve different streams of one event. Compared to our work, this approach tries to find scenes of one event based on similar motion, whereas we want to retrieve as many different real life events as possible that share the same motion pattern.

A motion based selection of relevant video segments is done in [3]. Like our approach, this solution also does a segmentation of the video based on the dominant motion, but the motion information is obtained from the pixel domain, not from the compressed domain, like our algorithm does. After the shot detection the recognized shots are classified using a classification algorithm that works on an off-line created training set. With the help of the classified shots videos can be divided into semantically meaningful scenes. The evaluation of the algorithm shows that concept detection only works for a very limited set of concepts. In our approach we do not make any assumptions on the semantic concepts contained in the video stream. We suppose that many domains exist where the most recurring motion sequence pattern within a video is an indicator for interesting scenes and thus we try to segment videos according to such patterns.

The algorithm introduced in this paper is based on previous work on video exploration [6]. With our interactive Video Explorer users can easily identify different motion sequences and search for similar subsequences. The motion information is extracted from compressed H.264/AVC videos and mapped to the HSV color space in order to visualize motion direction and intensity. A screenshot of the Video Explorer can be seen in figure 1. Below the video playback area an interactive navigation index is shown that visualizes the motion information. With the help of this index users can quickly and easily recognize semantics like fast or slow motion, the direction of the motion and camera zooms or camera pans. By selecting a motion sequence users can search for similar sequences throughout the whole video stream. To queries formulated this way a mnemonic can be assigned, which can be saved for later usage. These queries are not bound to a certain video, they can be used with any similar video, as only the query and not the result is saved. The strength of this tool is that an expert user is able to easily formulate a lot of queries and to provide them to non-expert users. The latter get a powerful and easy to use navigation tool for video content.

The limitations of the Video Explorer are that users can only search for small sequences and that recurring sequences have first to be identified by the user. Therefore, the new algorithm presented in this paper can be seen as an extension for it, towards automatic, semantic segmentation.

III. Recurring Motion Sequences

In many domains videos can be found that have been captured with fixed position cameras and that show several similar events throughout the video. In the sports domain e. g. these properties can often be observed. Think of a ski jumping event where the jumps of all competitors are always shown from the same camera positions and angles

with nearly the same zoom and pan sequences. An example for recurring sequences is given in figure 2. It shows an amplified part of the navigation index of figure 1. The upper bar represents 150 seconds of a ski jumping competition, where three athletes are shown. We have marked the jump offs of the three competitors with (A), (B) and (C). The lower bar shows a 15 seconds long excerpt of athlete (B) in detail. The different motion sequences that occur during his jump can clearly be recognized. The first greenish V-like sequence describes the motion during the jump, the yellow sequence shows the landing and the two other greenish and the pink sequences occur while the athlete arrives at the finish and stops. Comparing sequence (B) with the other ones in the upper bar, it can be well recognized that the three scenes consist of similar motion sequences. Regarding the aim of our algorithm, the lower bar shows a pattern of motion sequences that our algorithm searches for in the video stream.

Human observations with the video exploration tool, like the ones presented in figure 2, have shown that for many domains recurring motion sequences often belong to one semantical scene of the video. Thus, only by examination of the low-level visual feature motion, we can find high-level scenes in a video. It must be noted that we do not segment a video in all its scenes, but only in those scenes that match the identified pattern. These scenes typically consist of several shots. We do not intent to gain any semantic information from the video motion, but we suppose that the results contain semantically meaningful content. The identification of the semantics can be done by the user looking at the results. However, in many cases we do not need to identify the semantics at all. For example, in order to improve delivery, it is sufficient, if we are able to identify scenes, being good candidates for becoming popular. The delivery system can handle such candidates with priority - without the need to know their semantics.

The algorithm is divided into four steps. First we extract the motion information from the compressed video stream and build a motion histogram for each frame. Then we try to find coherent sequences with similar motion. After that we form clusters of similar motion sequences to be able to identify recurring patterns.

A. Motion Classification

The motion histogram for each frame is created using the motion vector information contained in H.264/AVC bit streams. As I-frames do not contain motion vectors, the motion information for I-frames is interpolated from the two adjacent frames. The extraction of the motion information is done in the compressed domain, therefore, no full decoding of the video stream is needed. The algorithm is described in detail in [5]. The resulting motion histogram consists of 13 bins as illustrated in Figure 3. Each of the bins expresses the percentage of pixels that move in that particular direction,

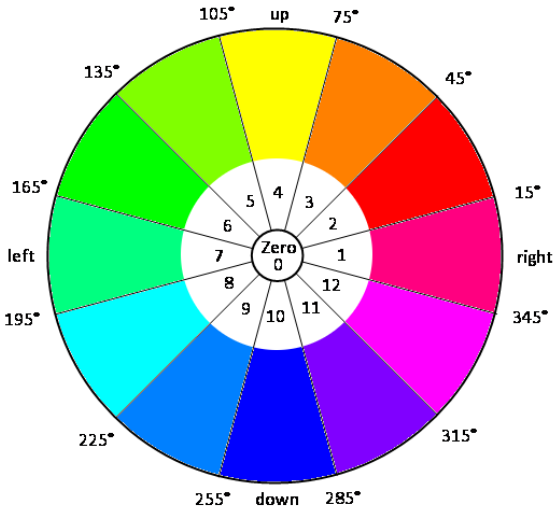

Figure 3. Motion vector classification for a motion histogram with 13 bins. Bin 0 expresses the amount of pixels with no motion.

e.g. bin 1 shows how many pixels move to the right, which means how many pixels belong to macroblocks with a motion vector angle between 345 and 15 degrees. Bin 0 indicates the amount of pixels that do not move at all.

B. Sequence Detection

After the motion histogram has been created for each single frame, we try to identify sequences of frames with a dominant motion into the same direction. Each frame is compared with its successor in the stream. If both show a similar motion direction, they belong to the same sequence and we compare the second frame with the next one. Two adjacent frames are regarded to have a similar motion if the difference in the motion direction between them does not exceed a certain threshold. Empirical investigations have shown that 40 % is a good value for a broad range of videos. If the relative difference between all bins of two frames is below that threshold, these frames are regarded to have a similar motion direction.

With one iteration over all frames we are able to identify all connected motion sequences within the video stream. To reduce the effort for the clustering and sequence detection steps we only consider motion sequences that have at least a certain length (currently 25 frames). Short motion sequences often occur due to noise in the motion information. With this restriction we avoid to include noisy information in the following steps of our algorithm. The motion sequence detection is somehow similar to shot detection, because we identify rather short video sequences and in most cases motion sequence boundaries correspond to the boundaries of a shot. An exception can be zoom and pan sequences,

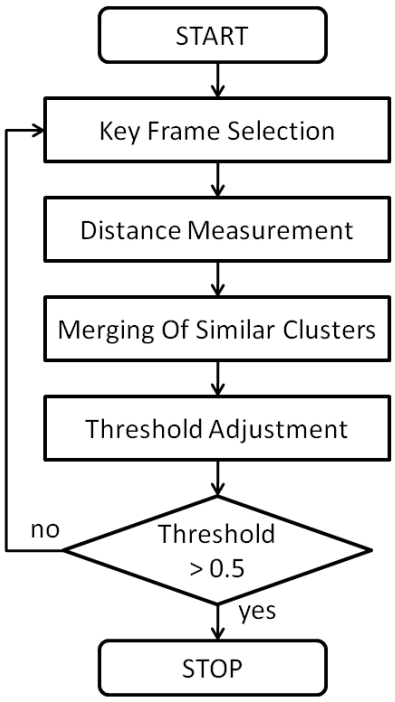

Figure 4. Flow chart of the hierarchical clustering algorithm. If the distance threshold exceeds 0.5, the iteration stops.

where the dominant motion can change within one shot.

C. Clustering

In the next step of our algorithm we perform a hierarchical clustering with the detected motion sequences. A flow diagram that illustrates all clustering steps is shown in figure 4. For each motion sequence we select a key frame that represents this sequence. To keep our algorithm simple and fast we decided to use the center frame of the detected motion, which is a common approach in video retrieval [8].

At the beginning of the hierarchical clustering each detected motion sequence forms an own cluster. Each key frame of a cluster is compared to the key frames of all other clusters. As distance metric we use the absolute difference between all motion histogram bins of the key frames. Clusters that have a minimal absolute distance that is below 5 % are merged and a new clustering round starts.

Beginning with the second iteration, we have to identify key frames for clusters that consist of more than one motion sequence. Again for each sequence we take the center frame as representing frame. Each of these frames is compared to all other representing frames of the cluster and the one that has the minimal distance to all other ones is selected as key frame for the cluster.

Then we calculate the minimal distances between all clusters again to merge all with a distance below the threshold. If in a round no clusters can be merged, because no pair of them has a distance below the threshold, we increase the threshold by another 5 %. This is repeated until the threshold reaches 50 %. Then the clustering stops and we start with the search for recurring patterns of motion sequences. Empirical observations showed that merging clusters with a distance higher than 50 % results in blurred clusters that have negative impact on the results of our algorithm.

D. Identification of Recurring Patterns

The clusters created in the previous step are numbered ascending. Each one of the chronologically ordered shots gets the corresponding cluster number assigned. In this sequence of cluster numbers we try to identify recurring patterns. We use a sliding window with an initial size of 4 and move it over the created cluster sequence to identify the pattern candidates. For each pattern candidate we estimate how many matches we can find in the whole cluster sequence for it. If the end of the cluster sequence is reached, we increase the size of the sliding window by one and start over with the pattern matching at the beginning.

For a better understanding we illustrate the basic principle of the pattern matching in figure 5. Each detected motion sequence is added to one of our clusters. The assignment of motion sequences to clusters is expressed by the cluster numbers above of each sequence. In the example shown, our algorithm detects four occurrences of the pattern 1-2-2-3, which corresponds to exactly one jump scene in the video stream.

These steps are repeated until the size of the sliding window equals to one third of the number of identified motion sequences. This is done because we are of the opinion that at least three occurrences of a pattern of motion sequences should be within a video stream in order to talk from recurring sequences.

At the end the pattern with the most occurrences within the cluster sequence is the one we use for the video segmentation. Each sequence of motion sequences that matches that pattern forms a scene. This leads to a partial segmentation of the original video, because all other scenes that do not correspond to the pattern found are ignored. We do only search for the most recurring event in a video. Each identified scene contains all frames from the beginning of the first sequence to the end of the last sequence in the pattern. Therefore, frames between the included motion sequences that have been left out due to the restrictions of the sequence detection (only sequences with at least 25 frames are recognized) are also added to the scene.

The identified scenes form the top level of a hierarchical video representation. The second level consists of the motion sequences that build the basis for the scenes. The third level contains the single frames of all motion sequences and the fourth level consists only of the key frames that represent the motion sequences. As already mentioned we consider the center frame of each motion sequence to be the key frame.

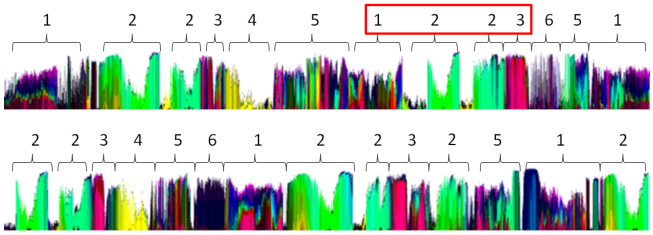

Figure 5. Each motion sequence is assigned to a cluster, referenced by the number of the cluster. In this sequence of numbers we try to find recurring patterns. In the example shown the sequence 1-2-2-3 is the most recurring pattern.

IV. Evaluation

We have tested our scene segmentation solution with two test videos from the sports video domain, in particular with videos of ski jumping competitions. Although our algorithm is unsupervised and only based on the low-level video feature motion that does not contain any semantic information, we get promising results.

A. Test data

Videos of ski jumping competitions are well suited for testing our algorithm, because they contain a lot of short recurring sequences that have been captured from fixed camera positions, good preconditions to test the effectiveness of our algorithm.

The first video (”oberstdorf”) has an overall length of 23 minutes and 24 seconds and it shows the jumps of 16 competitors. At the beginning of the video stream some commercials and interviews are shown. The second video (”garmisch”) has a duration of 1 hour 39 minutes and 36 seconds. It shows 68 jumps that are distributed across the whole stream. From time to time single jumps are interrupted by commercials, interviews or result tables. Four of the jumps are replay sequences that are shown with normal playback speed in the break between the two runs of the competition, thus they are also considered to be found by our algorithm. This ground truth has been manually created for both videos.

B. Results

We ran our algorithm on both test videos and counted how many jump scenes we got in our result set. We express the performance of the algorithm using the common evaluation metrics Recall and Precision.

$$Recall = \frac{|R \cap P|}{|R|}$$

$$Precision = \frac{|R \cap P|}{|P|}$$

Where R is the set of relevant scenes and P is the set of found scenes.

The results for both videos are shown in Table I. For the first video (”oberstdorf”) we get 12 relevant scenes from 16.

Video	oberstdorf	garmisch
Relevant scenes	16	68
Scenes found	12	59
Slow motion scenes	2	21
Size resultset	20	115
Recall	0.75	0.87
Precision	0.6	0.51
Precision (incl. slow motion)	0.7	0.7

Table I
Evaluation results

The result set has a size of 20 scenes, this leads to a recall of 0.75 and a precision of 0.6.

For the second video (”garmisch”) we receive a result set of 115 scenes, 59 of them show jump scenes. This means a recall of 0.87 and a precision of 0.51.

The low precision values are a result of the way we rate the results of our algorithm. We only count the jump scenes that are shown with normal playback speed as relevant results, but the result set also contains slow motion replay scenes that show jumps. In our first calculation of the precision these playback scenes are not regarded to be relevant results, thus we only reach these low values. Therefore, we investigated the motion sequence patterns of the slow motion scenes of our result set with the Video Explorer. We noticed that 2 slow motion scenes of the video ”oberstdorf” and 21 slow motion scenes of the ”garmisch” video show the same motion pattern like the relevant scenes in our result set. If we calculate precision by including the found slow motion scenes, we reach a value of 0.7 for both videos. These results are also shown in Table I.

C. Performance

The most time consuming task of our algorithm is the extraction of the motion vector information from the video stream. Our advantage is that we extract this information in the compressed domain of H.264/AVC videos. Motion classification is a low-complexity task, as it does not require full decoding of the video. The complexity is dominated by the entropy decoding, which consumes 22 to 42 percent of the full decoding workload [5].

For the three other steps of our algorithm (motion sequence detection, clustering and pattern matching) we did detailed measurements on a desktop computer with an Intel Core2 Duo CPU with 2.8 GHz and 4 GB RAM. The ”oberstdorf” video consists of 35112 frames, we detect 215 motion sequences and 20 recurring patterns in 1230 ms. The ”garmisch” video consists of 149415 frames, our algorithm finds 933 motion sequences and 115 recurring patterns in 48678 ms, in fact less than one minute for a video that has a length of nearly 1 hour and 40 minutes. The processing time grows significantly for the longer video. This is due to the fact that more motion sequences are found in the longer video and, therefore, the clustering needs more time.

V. Conclusion and Future Work

We presented an algorithm that identifies recurring patterns of motion sequences within a video stream. In many domains a recurring pattern that can be found throughout the video stream seems to comply to important semantic scenes of this video. This interesting fact can be used to automatically segment video streams into high-level scenes that are typically larger than shots and that contain more meaningful information. The evaluation has shown that the algorithm delivers promising results.

In future, we plan further improvements of our scene detection approach. First of all, we plan to investigate the used thresholds in detail. We want to evaluate how different thresholds influence the performance of our algorithm. In the end it should be possible to define metrics how these thresholds could be obtained and adjusted automatically.

Moreover, the algorithm has to be evaluated with further videos, not only from the sports domain, but also from other ones in order to evaluate the overall performance of our algorithm. In addition we are going to develop further segmentation methods based on other visual features like dominant color streams or concept detection with local key points. We hope to gain insights for which domain which segmentation method works best.

We plan to integrate our results in the Video Explorer to provide additional methods for the retrieval and annotation of relevant video segments. Further, we intend to take advantage of existing annotations, created by users manually. They could be used to improve the results of our scene detection. Last but not least we want to introduce an implicit relevance feedback for video segmentation. Simply by monitoring, which results are accessed by users and which not, we are going to train the algorithm for future segmentation tasks.

Acknowledgment

The results presented in this paper are part of the research efforts for the SOMA (Self-organizing Multimedia Architecture) project, a Klagenfurt University and Lakeside Labs cooperation (URL: http://soma.lakeside-labs.com).

References

[1] I. Döhring and R. Lienhart. Mining tv broadcasts for recurring video sequences. In *CIVR '09: Proceeding of the ACM International Conference on Image and Video Retrieval*, pages 1–8, New York, NY, USA, 2009. ACM.

[2] R. Laganière, R. Bacco, A. Hocevar, P. Lambert, G. Païs, and B. E. Ionescu. Video summarization from spatio-temporal features. In *TVS '08: Proceedings of the 2nd ACM TRECVid Video Summarization Workshop*, pages 144–148, New York, NY, USA, 2008. ACM.

[3] N. Peyrard and P. Bouthemy. Motion-based selection of relevant video segments for video summarization. *Multimedia Tools and Applications*, 26(3):259–276, August 2005.

[4] Y. Rui, S. X. Zhou, and T. S. Huang. Efficient access to video content in a unified framework. *Multimedia Computing and Systems, International Conference on*, 2:735+, 1999.

[5] K. Schoeffmann, M. Lux, M. Taschwer, and L. Böszörmenyi. Visualization of video motion in context of video browsing. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 658–661, August 2009.

[6] K. Schoeffmann, M. Taschwer, and L. Böszörmenyi. Video browsing using motion visualization. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 1835–1836, August 2009.

[7] A. Sobe, L. Böszörmenyi, and M. Taschwer. Video Notation (ViNo): A Formalism for Describing and Evaluating Non-sequential Multimedia Access. *IARIA Journals*, submitted January, 2010.

[8] B. T. Truong and S. Venkatesh. Video abstraction: A systematic review and classification. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(1):3+, 2007.

[9] Y. Yasugi, N. Babaguchi, and T. Kitahashi. Detection of identical events from broadcasted sports video by comparing camera works. In *MULTIMEDIA '01: Proceedings of the 2001 ACM workshops on Multimedia*, pages 66–69, New York, NY, USA, 2001. ACM.