

- [8] M. Gorkani and R. W. Picard. Texture Orientation for Sorting Photos "at a Glance". *Proc. International Conference on Pattern Recognition*, Jerusalem, Vol. I, pp. 459-464, 1995.
- [9] A.G. Hauptmann, and M.A. Smith. Text, Speech, and Vision for Video Segmentation: The Informedia Project. *AAAI-95 Fall Symposium on Computational Models for Integrating Language and Vision*, Nov. 1995.
- [10] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, R. Zabih. Image Indexing Using Color Correlograms. *IEEE Computer Vision and Pattern Recognition Conference*, San Juan, Puerto Rico, pp. 762-768, June 1997.
- [11] M. Hunke. Locating and tracking of human faces with neural networks. Master's thesis, University of Karlsruhe, 1994. <http://ernie.sfsu.edu/~hunke/>
- [12] B. Jähne. Digital Image Processing. Concepts, Algorithms, and Scientific Applications. Springer-Verlag, Berlin, Heidelberg, New York, 1995.
- [13] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face Recognition: A Convolutional Neural Network Approach. *IEEE Transactions on Neural Networks, Special Issue on Neural Network and Pattern Recognition*, accepted for publication.
- [14] W. Li, S. Gauch, J. Gauch, and K.M. Pua. VISION: A digital video library. *ACM Multimedia Magazine*, pp. 19-27., 1996.
- [15] R. Lienhart. Methods of Content Analysis, Indexing and Comparison of Digital Video Sequences. Ph.D. thesis, Shaker Verlag, July 1998. (in German)
- [16] R. Lienhart, W. Effelsberg, and R. Jain. Towards a Visual Grep: A systematic analysis of various methods to compare video sequences. *Proc. SPIE 3312, Storage and Retrieval for Image and Video Databases VI*, Ishwar K. Sethi, Ramesh C. Jain, Editors, pp. 271-282. 1998.
- [17] C.Y. Low, Q. Tian, and H. Zhang. An Automatic News Video Parsing, Indexing and Browsing System. *Proc. ACM Multimedia 96*, Boston, MA, pp. 425-426, Nov. 1996.
- [18] K. Minami, A. Akutsu, H. Hamada, and Y. Tonomura. Enhanced Video Handling based on Audio Analysis. *Proc. IEEE International Conference on Multimedia and Systems*, Ottawa, Canada. pp. 219-226, June 1997.
- [19] A. Merlino, D. Morey, and M. Maybury. Broadcast News Navigation Using Story Segmentation. *Proc. ACM Multimedia 97*, Seattle, USA, pp. 381-391, Nov. 1997.
- [20] G. Pass, R. Zabih, and J. Miller. Comparing Images Using Color Coherence Vectors. *Proc. ACM Multimedia 96*, Boston, MA, USA, pp. 65-73, Nov. 1996.
- [21] A. Pentland, B. Moghaddam, and T. Starner. View-Based and Modular Eigenspaces for Face Recognition. *IEEE Conf. on Computer Vision and Pattern Recognition*, Seattle, WA, July 1994.
- [22] S. Pfeiffer. The importance of perceptive adaptation of sound features for audio content processing. Accepted for *SPIE conferences, Electronic Imaging 1999, Storage and Retrieval for Image and Video Databases VII*, EI24.
- [23] H. A. Rowley, S. Baluja, and T. Kanade. Human face recognition in visual scenes. *Technical Report Carnegie Mellon University*, CMU-CS-95- 158R, School of Computer Science, Nov. 1995.
- [24] C. Saraceno and R. Leonardi. Audio as a Support to Scene Change Detection and Characterization of Video Sequences. *Proc. 22nd IEEE International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, pp. 2597-2600, May 1997.
- [25] Y. Taniguchi, A. Akutsu, and Y. Tonomura. PanoramaExcerpts: Extracting and Packing Panoramas for Video Browsing. *Proc. ACM Multimedia 97*, Seattle, USA, pp. 427-436, Nov. 1997.
- [26] M.M. Yeung, B.-L. Yeo, W. Wolf, and B. Lu. Video Browsing using Clustering and Scene Transitions on Compressed Sequences. *Proc. SPIE 2417*, pp. 399-413. 1995.
- [27] M.M. Yeung and B.-L. Yeo. Time-constrained Clustering for Segmentation of Video into Story Units. *Proc. International Conference on Pattern Recognition*, Vol. C, pp. 375-380. 1996.
- [28] M. Yeung, B.-L. Yeo. Video Content Characterization and Compaction for Digital Library Applications. In *SPIE 3022, Storage and Retrieval of Image and Video Databases 1997*, pp. 45-58, Jan. 1997.
- [29] R. Zabih, J. Miller, and K. Mai. A Feature-Based Algorithm for Detecting and Classifying Scene Breaks. *Proc. ACM Multimedia 95*, San Francisco, CA, pp. 189-200, Nov. 1995.
- [30] H.J. Zhang, S. Tau, S. Smoliar, and G. Yihong. Automatic Parsing and Indexing of News Video. *Multimedia Systems*, 2, (6)1995, pp. 256-266.
- [31] D. Zhong, H.J. Zhang, and S.-F. Chang. Clustering Methods for Video Browsing and Annotation. *Storage and Retrieval for Still Image and Video Databases IV, IS&T/SPIE's Electronic Imaging Science & Technology 96*, 2670, San Jose, CA, Feb. 1996.

7.3.3 Quality of Dialog Determination

Unlike the other features, the dialog detection requires no additional parameter since the feature detector already made that binary decision. The hit rate for shot/reverse shots dialogs was 35% and 60%, respectively. The false hit rates were about 5% and 118%.

Movie	# shots	# dialogs	#hits / hit rate	# missed hits / miss rate	# false hits / false rate
Groundhog Day	713	173	60 / 35%	113 / 65%	9 / 5%
Forrest Gump	918	111	66 / 60%	45 / 40%	131 / 118%

Table 4: Performance of the dialog detection

7.3.4 Quality of Scene Determination

We also combined the scenes of the dialog detection, setting detections and audio sequence to construct even better setting scenes. The algorithm proceeds by combining the determined clusters into clusters of maximum size. Whenever two clusters overlap, they form one bigger cluster. Then, all scenes were split up if two shots were combined by a fade since fades always separate scenes. Finally, the “gaps” in between clusters were merged into scenes of their own.

This scene merge procedure resulted in 35 scenes for “Groundhog Day” and in 83 scenes for “Forrest Gump”. The percentages of shot boundaries which our algorithms have correctly found to be mid-scene ones is quite impressive: 96% and 92% (see Table 5). This means, that we have deleted most of the mid-scene boundaries which a human would also delete. However, we have also deleted 18% respectively 11% more.

Movie	# shots	# scenes	#hits / hit rate	# missed hits / miss rate	# false hits / false rate
Groundhog Day	713	592	569 / 96%	23 / 4%	110 / 18%
Forrest Gump	918	813	750 / 92%	63 / 8%	89 / 11%

Table 5: Performance of the scene determination

8 Conclusion and Outlook

We have presented four features which allow shots to be clustered into audio sequences, settings and dialogs. Each scene type provides important information about the structure of a video. We measured the performance of our shot clustering approach against the ground truth manually created by the authors. The hit rate ranged from 35% to 90% at false hit rates between 5% to 118% for the two feature films “Groundhog Day” and “Forrest Gump”. To our knowledge, this is the first time that the performance of a shot clustering algorithm was evaluated against a ground truth.

In general, the performance depends mainly on the feature and much less on the type of clustering algorithm employed. The better a feature or a feature set captures the semantics of certain kinds of scenes, the more correct are the constructed scenes. Thus, in the future we will try to improve the features which capture audio setting, the setting in general and the dialogs. Moreover, we are working on using the distance tables to construct a hierarchical video representation, which would lead to an intuitive video table of contents (VToc) by finding acts, scenes, and shots. Browsing, abstracting and video annotation applications could benefit from such an automatically generated VToc .

References

- [1] H. Aoki, S. Shimotsuji, and O. Hori. A shot classification method of selecting effective key-frames for video browsing. *Proc. ACM Multimedia 96*, Boston, MA, pp. 1-10, Nov. 1996.
- [2] F. Arman, R. Depommier, A. Hsu, and M.-Y. Chiu. Content-based Browsing of Video Sequences. *Proc. ACM Multimedia 94*, San Francisco, CA, pp. 97-103, Oct. 1994.
- [3] A.A. Armer. *Directing Television and Film*. Wadsworth Publishing Co., 2nd ed., 1990.
- [4] F. Beaver. *Dictionary of Film Terms*. Twayne Publishing, New York, 1994.
- [5] S. Belognie, C. Carson, H. Greenspan, and J. Malik. Recognition of Images in Large Databases Using a Learning Framework. *UC Berkeley CS Technical Report 97-939*, 1997.
- [6] R.M. Bolle, B.-L. Yeo, and M.M. Yeung. Video Query: Research directions. *IBM Journal of Research and Development*, Vol. 42, No. 2, pp. 233-252, March 98.
- [7] D. Bordwell, K. Thompson. *Film Art: An Introduction*. McGraw-Hill, Inc., 4th ed., 1993.

sequences, which calculates to 94 semantic units based on audio sequences if intermediate shots are integrated into one semantic unit. This implies a reduction of the number of shots by 87%. For “Forrest Gump”, the hit rate was 66% for a suboptimal parameter setting at a false hit rate of 7%. We get 76 unconnected audio sequences implying a reduction in semantic units by 83%.

A qualitative assessment via the performance visualization tool and the scene overview shows that audio sequences which are based on continuing music do not necessarily imply a scene boundary. Often, music in movies is used in transitions from one scene to the next in order to facilitate an emotional change or to intensify an existing mood [3]. The audio sequence operator then deletes a shot limit which may be fundamentally important as a semantic break. However this does not happen often, because such music is often accompanied by a soft edit such as a dissolve or a fade causing the audio sequence boundary to fall within the following shot.

Difficult to determine are audio sequences based on speech. Speech is very dynamic in its general structure as it is often interrupted by short breaks and its spectral composition changes very quickly. Semantic examination of the audio stream by determination of music, speech and noise parts, similar to [18], can overcome this difficulty.

7.3.2 Quality of Video Setting Determination

Again, we first performed some tests on “Groundhog Day” in order to obtain an optimal choice of the distance threshold for the clustering algorithm on the distance table (see Figure 4 (a) and (b)). This process led us to a threshold of 0.025 for orientation and 0.10 for color. At this threshold, the hit rate for settings was 82%, and 76% for orientation and color, respectively, at false hit rates of 15% and 10%. We found 28 (unconnected) like settings via orientation and 52 via color. For “Forrest Gump”, the same thresholds were used, leading to hit rates of 90% and 58% at false hit rates of 10% respectively 42%. We found 75 like settings via orientation and 82 via color (see Table 2 and Table 3).

Qualitatively speaking, it seems that the settings are either determined with a high precision by the algorithm - and this is the general case - or are completely screwed up.

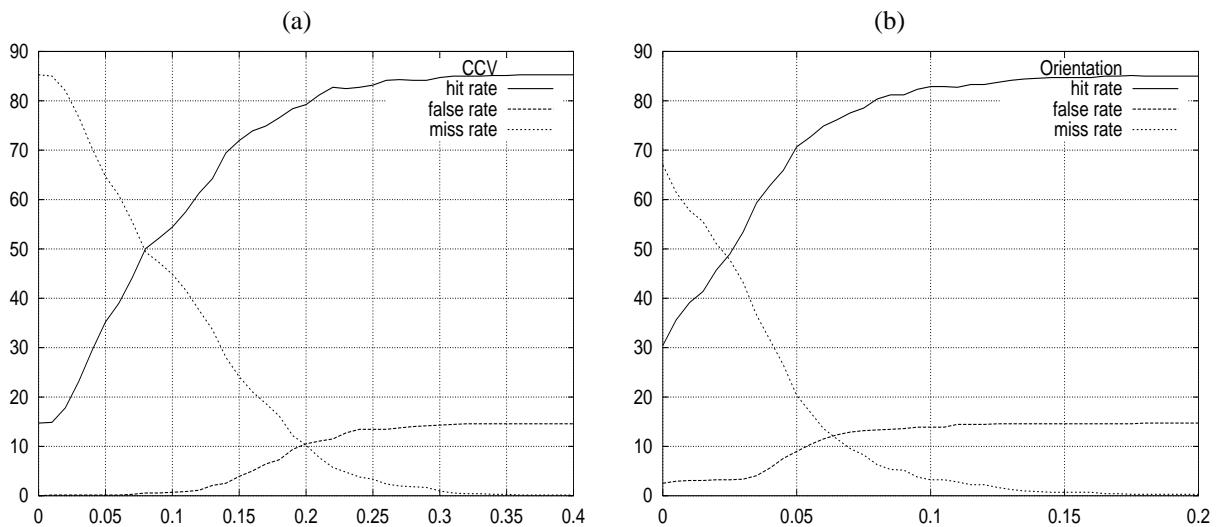


Figure 4: Performance of the setting determination by (a) CCV or (b) orientation in dependence of the absolute threshold

Movie	# shots	# settings	#hits / hit rate	# missed hits / miss rate	# false hits / false rate
Groundhog Day	713	580	475 / 82%	105 / 18%	90 / 15%
Forrest Gump	918	680	613 / 90%	67 / 10%	195 / 28%

Table 2: Performance of the setting determination with orientation

Movie	# shots	# settings	#hits / hit rate	# missed hits / miss rate	# false hits / false rate
Groundhog Day	713	580	438 / 76%	142 / 24%	56 / 10%
Forrest Gump	918	680	391 / 58%	289 / 42%	69 / 10%

Table 3: Performance of the setting determination with color



Figure 2: Interface of the performance visualization tool

formance as described above.

Movie	# shots	#sequences	#hits / hit rate	# missed hits / miss rate	# false hits / false hit rate
Groundhog Day	713	586	474 / 81%	112 / 19%	99 / 17%
Forrest Gump	918	809	530 / 66%	279 / 34%	60 / 7%

Table 1: Performance of the audio setting determination

For “Groundhog Day”, the hit rate was 81% at a false hit rate of 17%. We found 47 (unconnected) audio

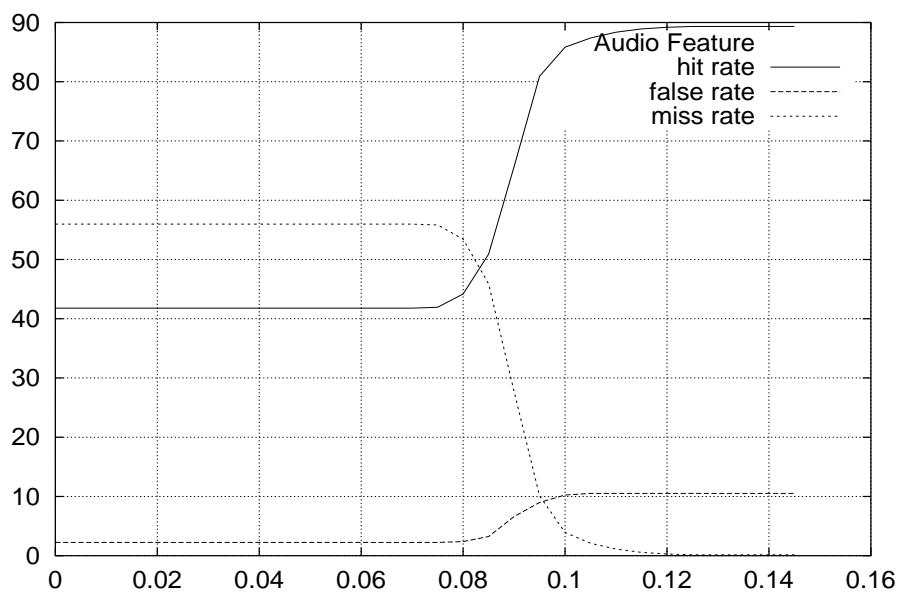


Figure 3: Performance of the audio setting determination in dependence on the absolute threshold

7 Experimental Results

7.1 Setup

The proposed individual shot clustering algorithms and their combination have been implemented in C++. Experiments were performed on a video database consisting of two full-length feature films: “Groundhog Day” and “Forrest Gump”. The former was digitized in motion JPEG from German TV at a resolution of 360x270 and a compression rate of 1:15, while the latter was extracted as an MPEG-1 video from a CD-I. For the audio tracks, a sampling rate of 8000 Hz, mono, coded in 8 bit μ -law was sufficient because most of the audio content is present in the frequency bands below 4000 Hz. For each feature film we calculated the features and performed the shot clustering, as described above.

7.2 Methodology

The task of shot clustering or scene determination can be formulated either as the task of finding the scenes or of eliminating shot boundaries. Both formulations - in the result - are equivalent to each other. Here, in the experimental results section it is more convenient to use the shot boundary elimination view.

In order to judge the results of our clustering algorithms, we determined “by hand” for each feature the ground truth telling which shots belong together and which do not. We stored a “1” for a shot boundary in the reference track, if the two associated shots belong to the same scene and a “0” otherwise. The reference track was constructed jointly by the authors after intensive discussion at some critical points.

The performance of the different features for shot clustering is measured by three basic numbers. For their definition we use the term “scene boundary” as a place holder for dialog, audio sequence and setting boundaries:

- **hit rate h :** The hit rate h specifies the percentage of correctly detected scene boundaries in relation to their actual number.
- **miss rate m :** The miss rate m determines the percentage of missed scene boundaries in relation to their actual number, i.e. $1.0 - h$
- **false hit rate f :** The false rate f gives the percentage of falsely detected scene boundaries in relation to their actual number.

The hit and false hit rates of the three shot clustering algorithms are influenced by their respective parameters. In general, if the change of a parameter increases the hit rate, the false hit rate also increases. Thus, it is difficult to define optimal parameters. There is always a trade-off between hit rate and false hit rate. In Section 7.3, we will show how the performance changes with the parameters.

A visualization of the performance of the different shot clustering algorithms gives a more intuitive overview of their quality than do quantitative data. We therefore constructed a tool which compares the detected clusters with the manually produced reference clusters. Figure 2 shows the results for the beginning of “Groundhog Day” for the audio setting. Each rectangle specifies a shot with its specific length. Each row shows the shots which have been clustered manually into a scene by humans with respect to the chosen semantic feature. No gap between two shots signifies clustering by our algorithm. If two automatically clustered shots overflow a manual cluster end, an arrow is painted.

Analogous to the evaluation of performance of image or text databases tens of thousands of shots and scenes are needed to evaluate the performance of any video comparison algorithms reliably. Unfortunately, building up such a large video database and determining the ground truth presently exceeds the possibilities of our multimedia lab. Thus we had to restrict our experiments to a much smaller database.

7.3 Results

7.3.1 Quality of Audio Sequence Determination

At first, we performed some tests on the distance table in order to determine the optimum distance threshold for the clustering algorithm. This led us to a threshold of 0.087 for “Groundhog Day” (see Figure 3). This value was also set for the analysis of “Forrest Gump”. With this threshold, we compared the resulting audio sequences with the manually determined ones. The resulting hit, miss and false hit rates are shown in Table 1. The first column in the table specifies the number of (automatically detected) shot boundaries. The second column gives the number of boundaries which were clustered manually in the reference database. The three following columns give the per-

However, such approaches have to deal with the problems caused by important orientations which are close to the boundaries of regions and which result in accidental assignments to one or another region.

We took another approach which captures the local orientation of an image independently of translations and small or middle scalings. Local orientation of an image is captured by an orientation correlogram. It is defined - like the color correlogram in [10] - as a table indexed by an orientation pair $\langle i, j \rangle$. The k th entry $\gamma_{i, j}^k$ of an orientation pair $\langle i, j \rangle$ specifies the probability that within a distance of k of an orientation i the orientation j can be found in the image. Thus, the orientation correlogram describes how the spatial correlation of orientation pairs changes with distance. As a feature it is more robust than the detailed raw orientation image while avoiding the poor discriminating power of highly aggregated features such as histograms of local orientation.

In defining orientation correlograms, the orientation ϕ is discretized in n classes K_i :

$$K_i = \frac{(i-1)}{n} \cdot \pi \leq \phi < \frac{i}{n} \cdot \pi, i \in N = \{1, \dots, n\}.$$

Using d different distances the space requirements come out as $O(n^2 d)$. In the experiments, we chose $n = 8$ and $d = |D = \{1, 3, 5, 7\}|$. The distance between two orientation correlograms is measured based on the probability of the components by

$$|I_1 - I_2|_{Orientation} = \sum_{i, j \in N, k \in D} \frac{|\gamma_{i, j}^k(I_1) - \gamma_{i, j}^k(I_2)|}{1 + \gamma_{i, j}^k(I_1) + \gamma_{i, j}^k(I_2)}$$

5.3.2 Calculation of Distance Tables

The distance between two shots with respect to their color or orientation content is measured based on the disaggregated set representation of the shots, using the minimum distance between the most common feature values (see [16] for more details), i.e. each shot S_i is described by the set of feature values $\{f_1^i, \dots, f_n^i\}$ derived from each of its frames and compared with $S_j = \{f_1^j, \dots, f_m^j\}$ by $dist(S_i, S_j) = \min\{d_{feature}(f_i, f_j) | f_i \in S_i, f_j \in S_j\}$.

6 Scene Determination

The question now is how to use the distance matrix to determine scenes. Two issues arise. Firstly, although scenes are defined by a common feature this does not imply that the feature has to be present in each shot of the scene. In many cases a set of feature values describes a scene which cannot be presented in each shot. One example of this is the setting. The three-dimensional space is usually introduced by shots from different perspectives. Thus, one must look not only at neighboring shots but also several shots ahead.

Secondly, it is easy for humans to judge whether two shots are similar with respect to some feature, however, our algorithm requires a decision function. We considered two possible decision functions:

- (1) Absolute thresholding or
- (2) Adaptive thresholding.

There are two possibilities for adaptive thresholding: choose the threshold based either on the distance matrix of the video by specifying the number of scenes or based on the distance values in a temporal neighborhood. However, it is not clear how the temporal neighborhood should influence the threshold. Higher distance values could either mean that distances between the shots in the scene are greater in that part of the movie, such that adaptation of the threshold would be correct. But it could also mean that the story in this part of the movie is developing very rapidly, so the shots have nothing in common. An example are different short views of different landscapes implying that the actors undertook a long journey. If the like setting determination would adapt the threshold, it would reduce its threshold and thereafter group settings which are dissimilar.

We tested each clustering scheme, and absolute thresholding worked best. For the two movies which we analyzed, we determined optimal absolute thresholds automatically. Refer to the experimental results for details. For the lookahead, we have chosen a value of 3 shots.

Our scene determination algorithm works as follows: A shot cluster comprises all shots between two shots which are no further apart than the lookahead and their distance is below the threshold. Overlapping shot clusters are grouped into scenes.

$$\sum_{j=1}^n \left(\frac{|\alpha_j^1 - \alpha_j^2|}{\alpha_j^1 + \alpha_j^2 + 1} + \frac{|\beta_j^1 - \beta_j^2|}{\beta_j^1 + \beta_j^2 + 1} \right)$$

In experimental results this measure outperformed the Euclidean distance when retrieving images similar to a given image from a large database [20]. The distance values range from 0 to about $2n$.

ORIENTATION. Orientation of structures in images is another feature that is suitable to characterize a scene to some extent. For instance, in pictures of city scenes with many buildings, one can expect many vertical edges. In a frontal view many horizontal lines are visible, too. In contrast, this type of orientation is much more unlikely in images of humans or natural scenes [8]. Thus, it may be reasonable to describe a picture by the orientation it contains. Moreover, orientation might be especially suitable to describe characteristics of background settings.

The *prototype of local orientation* is defined as an image structure in which the gray or color values change only in exactly one direction, but remain static in the orthogonal direction. Orientation, however, does not distinguish between direction x° and $(x + 180)^\circ$. Consequently, it varies only between 0 and 180° , unlike direction which ranges from 0 to 360° [12].

The various algorithms to determine local orientation commonly operate on gray-scale images. Before computation it is useful to increase the global contrast in order to prevent structures from emerging inadequately in dark or bright images. The minimum and maximum gray-scale value, which occur with at least a certain significance (i.e. their frequency exceeds a threshold value), are determined by a gray-scale histogram and used to scale the pixels' gray-scale values to the full gray-scale range. The determination of orientation is carried out on such histogram-normalized images.

We derived orientation via the inertia tensor [5][12]. It allows neighborhoods of constant gray values to be distinguished from neighborhoods with isotropic structures or local orientation. The following presentation is based on the approach in [5]. A detailed derivation can be found in [12].

Designating the gray-scale value of a pixel at the position (x, y) by $I(x, y)$, the gradients along the x and y directions by $\nabla I(x, y)$ and the Gaussian filter by $G(x, y)$. The second momentum window matrix at position (x, y) is computed by

$$J(x, y) = \begin{bmatrix} J_{xx} & J_{xy} \\ J_{yx} & J_{yy} \end{bmatrix} = G(x, y) * (\nabla I(x, y) \cdot \nabla I^T(x, y))$$

with

$$J_{pq} = B(D_p \cdot D_q), p, q \in \{x, y\}.$$

Let λ_1 and λ_2 , $\lambda_1 \geq \lambda_2$ denote the eigen values of $J(x, y)$. Then, the orientation ϕ of the eigen vector associated with λ_1 can be determined by

$$\tan 2\phi = \frac{2J_{xy}}{J_{yy} - J_{xx}}$$

It measures the angle of orientation of the image structure around the location (x, y) . The relation of the eigen values to each other can be used as a certainty measure of the estimated orientation. Three cases can be distinguished:

1. $\lambda_1 \gg \lambda_2$: There is an orientation in direction ϕ .
2. $\lambda_1 \approx \lambda_2 \gg 0$: The gray-scale values change similarly in all directions, and thus, the structure is isotropic.
3. $\lambda_1, \lambda_2 \sim 0$: The local environment has a constant gray-scale value.

Only pixels with dominant local orientation are considered further.

In addition to the question how local orientation can be determined, it is also important to find a suitable aggregated feature that captures the characteristics of local orientation in the entire image. A standard approach would be to summarize local orientation by an orientation histogram. Although histograms are robust against slight changes in camera view and local object motion, they have little discriminating power and are therefore not suitable for large video databases. A typical proposal to overcome this drawback is to divide an image into several rectangular regions or into a foreground and a background region for each of which a histogram is calculated.

with similar faces in order to find groups of frames showing the same actors. Such a group of related frames is called a *face-based class*. In a first step, faces within shots are related to each other according to the similarity of their position and size in neighboring frames, assuming that these features change only slightly from frame to frame. This is especially true for dialog scenes. In addition, we dispose of accidental mis-classifications by the face detector by discarding all face-based classes with fewer than three occurrences of a face, and by allowing up to two drop-outs in the face-tracking process. This simple grouping algorithm works very well within shots and is computationally cheap. It does not demand complex face recognition algorithms such as described in [13]. In a second step, face-based classes with similar faces within the same shot are merged by the Eigenface face recognition algorithm [21] in order to obtain the largest possible face-based classes.

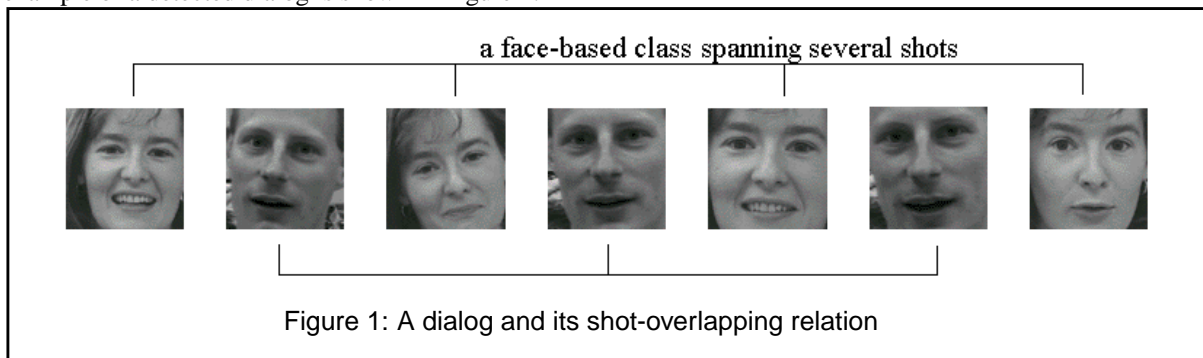
The same face recognition algorithm is used to identify and merge face-based classes of the same actor across shots throughout the video, resulting in so-called face-based sets. They describe where, when and in what size the actors appear in the video. However, the Eigenface face recognition algorithm cannot guarantee that all face groups of the same actors merge together. An actor's face varies too much throughout a video. Our grouping algorithm typically splits the main actors into a few distinguished face-based sets.

5.2.2 Dialog Detection

It is easy to detect typical shot/reverse-shot dialogs and multi-person dialogs with the frontal face detector. A sequence of contiguous shots of a minimum length of three shots is denoted as a *dialog* if

- (1) at least one face-based class is present in each shot no further apart from its neighbor than 1 second and
- (2) the Eigenface-based shot-overlapping relations between face-based classes interlink shots by crossings.

The length of the dialog is cut down to the first and last face-based set that has a shot-overlapping relation. An example of a detected dialog is shown in Figure 1.



5.3 Setting Determination

A setting is defined as a locale where the action takes place. Often, it can be detected by the repetitive appearance of the same background in a constant illumination. This makes it possible to use color and structure information to determine a setting.

5.3.1 Features: CCV and Orientation

CCV. Shots with very similar color content usually belong to a common setting because they share a common background. The color content changes more dramatically from setting to setting than within a single setting. Color content is usually measured by some sort of refined color histogram technique such as the color coherence vector (CCV)[20]. The CCV makes use of spatial coherence and discriminates much better than the basic color histogram. Instead of counting only the number of pixels of a certain color, the CCV additionally distinguishes between coherent and incoherent pixels within each color class j depending on the size of the color region to which they belong. If the region (i.e. the connected 8-neighbor component of that color) is larger than a threshold t_{ccv} , a pixel is regarded as coherent, otherwise as incoherent. Thus, there are two values associated with each color j :

- α_j , the number of coherent pixels of color j and
- β_j , the number of incoherent pixels of color j .

Then, the color coherence vector CCV_i is defined as the vector $\langle (\alpha_1^i, \beta_1^i), \dots, (\alpha_n^i, \beta_n^i) \rangle$ normalized by the number of pixels. Two CCVs CCV_1 and CCV_2 are compared by

t . The window is advanced by about 3/4 of the total window size to calculate the next audio feature vector. A *forecasting vector* is calculated by exponential smoothing forecasting, i.e by

$$F_{t+1} = \alpha x_t + (1 - \alpha)F_t,$$

where $0 < \alpha < 1$ is called the smoothing constant (here set to 0.2) and F_0 is the audio feature vector of the very first window x_0 . The forecast is the weighted sum of the last observation and the previous forecast. The forecasting vector therefore contains values similar to previous vector values. It represents the distribution of the intensities of the frequencies of all previous windows and therefore constitutes a template for the regarded audio shot. The speed at which the forecasting vector adapts to new feature values is controlled by the smoothing constant.

The decision about an *audio cut* is based on the difference between a new feature vector and the forecasting vector. The difference is calculated via the Euclidean distance. We have two difference thresholds: a high threshold which directly determines a cut (because there was a significant difference) and a lower threshold which determines similarity (i.e. the difference is only small). If too many consecutive feature vectors are classified as similar, we also deduce a cut. After a cut, calculation of the forecasting vector starts again with $F_t = x_t$.

Audio cuts determine audio shots, whose spectral content is represented by their last forecasting vector. Such a vector is also calculated on background segments. Normalized versions of these vectors are used in the calculation of the distance table. Thus, the background and audio cut algorithms result in a table of audio shots which are described by the vector $AS_k = (FS_k, t_k, \langle x_k \rangle)$, where FS_k specifies the frame sequence covered by the audio shot AS_k , $t_k \in \{Background, Foreground\}$, and $\langle x_k \rangle$ is a vector of real values between [0,1] which signifies the audio content of the audio shot. The normalization to [0,1] is performed as a calibration in order to get values that are independent of loudness influences, and in order to produce distance values between [0,1] when comparing two audio shots.

5.1.2 Calculation of Distance Table

A table of distances between (video) shots is calculated from the audio shots AS_k by comparing all spectral content vectors of each shot based on the normalized Euclidean distance metric. The closest vectors define the distance. It is interesting to note that if an audio shot overlaps two consecutive shots, their distance will be 0 because it has been calculated on the same spectral content vector.

5.2 Dialog Determination

A special scene type is a dialog. The form in which dialogs are presented can differ. Some dialogs show the dialog partners in turn, one at a time, in a frontal view. This pattern is called the shot/reverse shot pattern. In other dialogs, the dialog partners are all visible in the scene, typically from the side. Other dialog setups are possible. We concentrate on the very widely used shot/reverse shot pattern. To detect it, our shot grouping system must understand where the actors appear in the video. Therefore, we have implemented a face detection algorithm and a method of recognizing the face of the same actor across shot boundaries.

5.2.1 Feature: Frontal Face Detector

One of the most reliable *face detectors* in digital images was developed by Rowley, Baluja, and Kanade [23]. Their system detects about 90% of all upright and frontal faces while hardly ever identifying non-faces as faces. We have recreated their neural-network-based frontal face classification system for arbitrary images (e.g. photos, newspapers, and single video frames) as a basis for our frontal face detector in video sequences. To widen the range of detectable faces, our detector also searches for slightly tilted/rotated faces (± 15 degrees). This is necessary because the faces of the actors in motion pictures are always moving, in contrast to the faces in typical still images such as portraits and photographs of sports teams where they are usually depicted upright. However, this more general face search increases by a factor of three the number of patterns which have to be tested in each image. To speed up processing, the candidate frontal face locations are drastically reduced in an extremely fast pre-filtering step: Only locations whose pixel colors approximate human skin colors [11] and which show some structure (such as nose, mouth and eyes) in their local neighborhood are passed to the face detector. This pre-filter reduces the number of candidate face locations by 80%. Moreover, only every third frame of the video sequence is investigated. Each face detected is described by the vector $(t_j^i, x_{pos}, y_{pos}, s, \gamma)$. It specifies the frame t_j^i , in which a face of size s (in pixels) was detected, as well as the x- and y-coordinates (x_{pos}, y_{pos}) of its center and its angle of inclination γ .

So far, each detected face is isolated and unrelated to other faces in the video. The next task is to classify frames

ably. Thus, frames being part of a dissolve or fade are removed from the shots.

5 Feature Calculation

5.1 “Audio Sequence” Determination

Great semantic changes in videos occurs usually occur in conjunction with profound changes in the audio track. Therefore we define an “audio sequence” to be a sequence of shots in which very similar audio signal sequences recur. Examples include music continuing over several shots, ongoing dialog or ongoing noise like the cheering of a crowd.

We define two types of audio sounds, each of which is handled differently: background and foreground. Background sounds in a video give a general feeling of the atmosphere of a scene. They do not carry information necessary to an understanding of the film story - otherwise they would be in the foreground such that the audience could easily hear and understand them. However, background sounds are just as dynamic as foreground sounds. Treated in the same manner as foreground sounds they cause many audio sequence endpoints that are not intended as they are of no interest. Therefore, the first step in determining audio sequences is to determine time periods containing exclusively background sounds. We call such periods “background segments”.

Background segments are disregarded during the further processing. Only foreground segments are analyzed further for their spectral content. Great changes are being registered as “audio cuts”. During their calculation, a characteristic vector is calculated for each time period between two audio cuts. These time periods are called “audio shots”. The vectors are then used to calculate the table of distances between (video) shots. Sequential (video) shots whose audio shots differ greatly belong to different audio sequences and must therefore be assigned a large distance value. Profound changes in the audio track may also occur within shots, an example would be explosions. Such changes are not a problem and do not disturb the measure for two reasons. First, other audio shots are usually also part of the same (video) shot and enter the distance measure giving a small distance. Second, several sequential (video) shots are compared for their similarity so that the odd one out does not disturb the big picture.

5.1.1 Features

AUDIO BACKGROUND. Whenever sounds in a video are meant to be in the background, their loudness is reduced substantially. Therefore, background segments can be determined via a loudness threshold. Our implementation first calculates a perception-based loudness measure from the amplitude values of the audio track of a video[22]. Then, we determine the maximum loudness value l_{max} . It is used to calibrate the digitization level of the sound file. A percentage of this maximum loudness value gives the loudness threshold value $lt = \beta \cdot l_{max}$. This percentage is critical to the analysis, but an automatic determination is difficult. We have used values between 0.05 and 0.5 for β . The percentage depends on the average loudness level of the movie. If there are great amounts of quietly spoken speech, then we have to choose a very low level in order not to declare this speech as background.

The audio file is analyzed in windows of size T_w which are advanced in steps of size T_s . Windows which do not contain loudness values above the threshold lt are considered background windows. The size of T_w is dependent on the desired minimum resolution of background segments: selection of a small window might mean detecting small breaks between words as background segments, selecting a big window, the skipping of small but important background segments between scenes. We found good choices for T_w between 50 and 250 ms with T_s close to the same value.

An alternative method of determining background segments would be to calculate signal power on the analysis windows. This measure works similar to the loudness measure except that we calculate signal power on the windows:

$$P = \frac{1}{n} \sum_{i=1}^n x_i^2$$

and perform thresholding on the calculated power values. We experimented with it but it did not work as well as the loudness thresholding. Signal power does not seem to be a good feature to describe background segments.

AUDIO CUTS. *Audio cuts* are time instances which delimit time periods with similar sound. Hereto, a sliding Hamming window of size 100 ms is used to calculate the Fourier transform. Then the complex Fourier coefficients are converted into real values by calculation of decibels. The resulting real-valued feature vector for each window is called an *audio feature vector* x_i . x_i represents the distribution of the intensities of the frequencies within window

2 Related Work

The term “shot clustering” denotes two fundamentally different kinds of shot grouping: The first kind aims at finding groups of similar shots from within the entire video without any thought of temporal order (see e.g. [2][14][31]). Useful similarity measures were considered in detail in [16]. Our approach in this paper belongs to the second type of shot clustering: grouping contiguous shots according to their semantic correlation.

Much work has been published on this type of shot clustering via video analysis features. Yeo and Yeung have proposed a scheme to recover story units from video using time-constrained clustering [26][27][28]. The basic idea is to assign the same label to shots that are similar and then analyze the patterns of the resultant strings. Three different kinds of temporal events are extracted from those label patterns: dialog, action, and story unit. Each event can be regarded as a scene in the video. Yeo and Yeung use color histograms as the measure of similarity. They never compare their computed scenes against handmade scenes.

Aoki, Shimotsuji, and Hori used labelled keyframes to represent the content of shots and to cluster them [1]. In general, keyframe-based systems suffer from their ignorance of the dynamic aspects of shots such as the duration of events/actions and the time they take place in the shot (for example, the appearance of a new actor at the end of a shot).

There are only few approaches that determine scenes based on “audio” analysis. Confusingly, they do not use the digitized audio as their basis, but the written transcript or the closed captions [14][19]. In many cases a transcript or closed captions are either not available or do not exist, as for home videos. In order to make our approach as general as possible we do not use this information. Therefore, our work concentrates on the more ambitious task of analyzing the digitized audio directly. An idea very similar to our approach for audio sequence determination has been presented by Saraceno and Leonardi [24]. They segment and classify the audio track of videos into silence, speech, music and noise segments, using the segments to support the detection of shot boundaries. On top of the information calculated hereby, they proposed to merge shots into scenes, but - to our knowledge - never implemented or tested their proposal.

The InformediaTM project combines audio and image analysis in order to determine scenes [9]. They determine so-called audio paragraphs based on the analysis of the audio-aligned transcript and match these paragraphs to the nearest shot boundaries to determine video paragraphs. Video paragraphs are defined as a collection of shots adhering to the same context. Our approach also employs both: audio and video information, although in a different manner. It surpasses the InformediaTM approach in that it also uses setting and dialog information for scene determination.

3 System Overview

Our system proceeds in several steps as follows: In a first step, the shots are recovered from the video. They are the atomic units of our clustering system. Then the values for each semantic feature are calculated. Currently we determine audio features, color features, orientation features and faces appearing in the shots, as they are important for the types of scenes we want to determine. The system can be extended at any time by additional features.

Next, we determine the distances between shots with respect to each feature. We do not integrate the different features into one distance measure for three reasons. Firstly, it is not possible to determine a fair integrated distance measure for the different features. Some features may be more important than others, but at this stage, it is not possible to say which are and which are not. Secondly, an integrated distance measure would destroy the semantic message of each individual feature. It is easy to separately judge the correctness of clusters based on each single feature, but nearly impossible if they are combined. Thirdly, the semantics of different features may conflict with each other. For instance, a continuous audio track is often used to move smoothly from one setting to another. Thus, the shot clustering results of the respective features are contradictory.

Finally, based on the calculated shot distance tables, we are able to merge shots for each feature separately but by means of a single algorithm.

4 Determination of Shots

Shots are defined as “one uninterrupted image with a single static or mobile framing” [7]. Many algorithms have been proposed for shot boundary detection (see [29] and the references therein), however, only few try to determine the type and extent of edits precisely. Algorithms that are able to detect, classify and determine the extent of hard cuts, fades, and dissolves have been proposed in [14]. We use them here since it is necessary to eliminate all editing effects which concatenate shots. Edits may bias the similarity measure between contiguous shots unfavor-

Scene Determination based on Video and Audio Features

Rainer Lienhart^{1‡}, Silvia Pfeiffer[†], and Wolfgang Effelsberg[†]

[‡]Intel Research Labs, Santa Clara, CA 95052, USA

Rainer.Lienhart@intel.com

[†]University of Mannheim, Praktische Informatik IV, 68131 Mannheim, Germany

{pfeiffer, effelsberg}@pi4.informatik.uni-mannheim.de

ABSTRACT

Determining automatically what constitutes a scene in a video is a challenging task, particularly since there is no precise definition of the term “scene”. It is left to the individual to set attributes shared by consecutive shot which group them into scenes. Certain basic attributes such as dialogs, like settings and continuing sounds are consistent indicators. We have therefore developed a scheme for identifying scenes which clusters shots according to detected dialogs, like settings and similar audio. Results from experiments show automatic identification of these types of scenes to be reliable.

1 Introduction

Depending on their length, shots are either the letters or words of video productions. Hence, they contain little semantic information: information in the image track can often be reduced to a simple keyframe or panorama [25], while in the audio track it is usually either incomplete (like a part of a dialog) or very limited (like a cry). In order to partition a video into semantically richer entities, shots must be grouped together based on content. This procedure is often denoted as shot clustering².

In general, shot clustering can be performed more easily when restricted to a certain type of film genre. For example, clustering shots in newscasts often boils down to finding anchor shots [17][19][30]. News stories (or scenes) are then determined by grouping all shots in between two anchor shots including the first anchor shot.

However, we did not restrict our shot clustering work to a specific film genre. Preferring as universally applicable an approach as possible, our work focuses on shot clustering in a genre-independent fashion. The ultimate goal is to automatically determine shot clusters which a human would judge as “scenes”. A scene is “usually composed of a small number of interrelated shots that are unified by location or dramatic incident”[4]. In the literature these are also called “video paragraphs” [9], “story segments”[25] or “story units”[6]. Unfortunately, the clustering of shots into “scenes” depends on subjective judgements of semantic correlation analogous to the “clustering” of (text) sentences into paragraphs. At school simple rules for forming paragraphs might be “No longer than about 7 lines” or “Describe just one idea”. Despite these general rules a text given to different people will yield different paragraphs. The same is true for video scenes. Experiments, however, showed that some basic units are clustered by all viewers: contiguous shots of the same setting, dialogs and shot sequences combined by audio such as continuing music background.

In this paper, we present our algorithms for automatic determination of scenes. We begin with a discussion of related work in Section 2. Section 3 gives an overview of our own system. The automatic determination of shots is addressed in Section 4. This Section is kept short because it concerns a rather well-examined subject. The clustering of shots depends on the attribute which stands in the foreground. Each attribute is calculated via one or more content features which are deemed important for representing the attribute: audio content is described in Section 5.1, face and dialog determination in Section 5.2 and like setting information based on color and orientation hints in Section 5.3. All of these features yield a normalized table of distances between the shots of a video, which is exploited for clustering in Section 6. We present experimental results for the algorithms in Section 7 prior to concluding the paper with a discussion about further research in this area.

1.This research was mainly performed while the author was at University of Mannheim, Praktische Informatik IV, 68131 Mannheim, Germany

2.This term is actually a misnomer since clustering of contiguous shots is only one aspect of shot clustering

REIHE INFORMATIK

20/98

Scene Determination based on Video and Audio Features

Silvia Pfeiffer, Rainer Lienhart and Wolfgang Effelsberg

Universität Mannheim

Praktische Informatik IV

L15,16

D-68131 Mannheim