6 SYNTACTICAL AND SEMANTICAL DESCRIPTION OF VIDEO SEQUENCES

N. Luth, A. Miene, P. Alshuth

Center for Computing Technology
Image Processing Department
University of Bremen

{nluth, andrea, petera}@tzi.de

Abstract: Archiving, processing, and intelligent retrieval of digital video sequences can be efficiently solved by a high-level representation of video sequences. We describe a complete system for encoding the contents of video sequences based on their syntactical and semantical description. Our techniques are built on two phases: analysis and synthesis. The analysis phase involves the automatic generation of a syntactical structure using image and image-sequence analysis, image understanding as well as text recognition. Moreover, the syntactical description of images will be completed by a semantical understanding of video sequences. The full representation of video sequences will be generated during the synthesis phase by combining results from the analysis phase. Our developed prototype system is planned to be integrated into the co-operative work of a TV-team of the TV-station "*Radio Bremen*". Additionally, we describe some special functionality: scene analysis by clustering of video sequences with the aim of identifying a typical video genre. The latter is an useful approach for automatic trailer generation and for intelligent video editing as add-on for a video-cut-system.

6.1 INTRODUCTION

The last years have been significant for the rapid growth of multimedia technology. The process was especially accelerated by the fast development of hardware, and the increased use of Internet-communication. In this context vast amount of video data needs to be archived, browsed, and retrieved using a structure that allows to carry out an efficient access and retrieval of video sequences. The content-based analysis and retrieval of video data is important, but not yet suitable integrated in current video database systems. A great

amount of research work has been done on the analysis, indexing, and retrieval of still images in image databases. The image retrieval paradigm is a keyword annotation, mostly done manually. There are three main difficulties with this approach:

1. a great deal of manual effort to annotate interactively every video sequence,
2. different interpretation of image content because of human subjectivity, and
3. inconsistency of the keyword assignments among different archivists.

An automatic, content-based annotation is an additional way to overcome these difficulties. Surveys and discussions of approaches and systems for image retrieval have been published by [4] [23, 13]. The ImageMinerTM¹ was developed by the Image Processing Group at the University of Bremen [10] and realizes content-based retrieval of still images through an appropriate combination of methods for image analysis and understanding. The system has been applied to two domains: landscape images and technical drawings. The system provides an automatic generation of object-based image description.

Our early work on indexing MPEG videos extended the same approach used for still images by treating video sequences as collections of still images – extracting representation frames and their indexing using tested image database techniques [17]. With ImageMinerTM, the representation frames are used to process color, texture, and contour detection. To support MPEG videos in an image retrieval system, a special mechanism is needed to reduce the high number of frames of a video. A first step is the automatic shot detection. In a second step the amount of frames is reduced by choosing a representational frame for each shot. This frame should represent the entire content of a shot.

In this paper we concentrate on a complex content-based representation of the video sequences, including the syntactical and semantical description of images and recognized text. A special algorithm for a fast shot detection to index a video by image processing based on single images was developed. The parameters of the frame sequence to still image conversion are useful to describe camera parameters for a shot. This is done by a mosaicing technique. The structure of a given video is useful to efficiently and automatically index long video sequences. A comparison between shots gives an overview about cut frequency, cut pattern, and scene units. After a shot detection, the shots are grouped into clusters based on their visual similarity.

We further address the problem of automatic image analysis and appropriate use of application uncertain domain knowledge using stochastic grammars. The image analysis approach taken in this paper combines color, texture, contour, and information of shots for a robust description of video and image structures.

¹ImageMiner is a trademark of IBM Cooperation.

By using domain knowledge, our approach for object recognition is able to identify primitive objects and complex scenes on basis of various regions extracted during the image analysis. The resulting content descriptions will be stored in a textual form for automatic retrieval which could be carried out by any text retrieval algorithm.

6.2 APPLICATIONS

The developed system supports and facilitates the archiving process, especially the content-based video annotation and retrieval and the non-sequential access to video sequences. Part of the system is intended to be used in the daily work of the archivists and editorial staff at "Radio Bremen" TV-station for archiving and retrieval of a local TV-programm "*buten un binnen*". The archiving procedure includes the automatical digitalization and annotation of the programm as described in Sec. 6.4. Most of the video analysis algorithms are performed in real time. After that the archivist is able to add additional information to the automatical description if needed. The retrieval performance is very high because the search is done over the textual annotations without any necessity of time expensive image processing within the retrieval process.

The automatic identification of video genre in combination with expert knowledge makes it possible to extend the functionality of the system to automatic trailer generation.

Additionally, it is possible to extend a commercial cut system, like AVID [3], by the functionality for intelligent support during the configuration of a new video film or document similar to a genre chosen from an available case-base.

6.3 RELATED WORKS

6.3.1 Retrieval of still images

One of the first image retrieval projects was QBIC [24]. With a visual query interface an user can draw a sketch to find images with similar sketches, or find images with specific positioned color, texture, or formulate object motion for the video domain. Some systems are developed to perform retrieval systems on both sides: formulating an user query by finding useful techniques, like hierarchical representation of icons, objects, or keywords, and representing the results in a graphical view to collect similar objects together and to navigate for additional information. ART MUSEUM [12] is used to find images from a data base which contains artistic paintings and photos only. The algorithm of sketch retrieval and similarity retrieval is based on graphical features. An user can formulate a query by using sketches – from a template or drawn free-hand. The PHOTOBOK system is a set of interactive tools for browsing and searching single images and sequences [26]. A query is based on image content rather than using text annotations.

Another example for an interactive content-based retrieval is based on user's relevance feedback (URF) [27]. The URF is generated by capturing dynamically updated weights during user's query and perception subjectivity.

6.3.2 Video Retrieval

Last years image retrieval systems have been extended to video retrieval systems by addition of shot detection and representing shots by still images (key-frame or mosaic).

The VIRAGE VIDEO ENGINE uses several video-specific data, like motion, audio, closed caption, etc., to build an information structure and content information about a video [9]. The user can formulate a query to find commercials, scenes with special camera motion, e.g. a talking head, or just a scene with some given words. In most of the systems only limited information, like color- and texture-vectors or cut detection, is calculated automatically. None of them is able to build objects.

The extraction of spatially reduced image sequences from MPEG-2 compressed video sequences is investigated in [31]. Comparing to MPEG-1, MPEG-2 has an extended functionality and additional syntax; many different options can be mixed into a video sequence, making the task of analysis very complicated. The method is based on a new class of DCT domain operations. The reduced images extracted from MPEG-2 video are useful for video processing and fast low-cost browsing.

The task of similarity detection between video sequences is described in [18]. Their approach is based on the matching of frames, shots, scenes and videos, as a multiresolution method. At each level the features of the level of higher temporal resolution are considered. A similarity measure of video sequence matching is defined.

The Informedia-system [6, 7] aims to handle effectively about 1000-hour digital video library supporting content-based indexing and retrieval of video segments. Image processing, powerful speech recognition tool Sphinx-II, and natural language processing techniques were applied to the video data for content extraction and temporal segments. Moreover, a novel techniques for the automatic generation of video skims, consisting from significant audio and video components, was developed. This allows to produce automatically best abstracts of video sequences [5].

6.4 OVERVIEW OF THE APPROACH

We consider video annotation as a complex content representation, consisting of syntactical and semantical description of video sequences. The whole system architecture is shown in Fig. 6.1. The syntactical description is a result of video dividing into primitives, which constitute a video sequence. For example a video consists of shots and image frames, every image consists of color or gray level, texture, contour, and maybe textual information. The syntactical description is generated from low-level information. The semantical description

is a result of dividing syntactical description into units, which are similar to the human perception of the contents. For example, an image consists of green forest, blue sky and so on.

Figure 6.1: System architecture

6.4.1 Extraction of Low-Level Information

6.4.1.1 Shot Analysis. Our first step to extract syntactical information is a shot analysis based on MPEG-1 video streams. That means dividing the complete video stream into single shots to be analyzed separately. The developed method for shot detection is based on the difference of the chrominance and luminance values. The color values U and V of the chrominance are treated separately. If the difference reaches a certain threshold, a shot boundary is detected [8]. The difference is calculated for every two succeeding frames. This guarantees precise shot boundaries, which are essential for our approach because all further analysis is done on single shots. In the same step we analyze the camera motion within the shot. For this task we use the motion estimation part of the MPEG format [8], with the aim of automatically detecting camera pans or tilts in a shot.

One way to visualize the results of a shot detection is a video icon as shown in Fig. 6.2.

Figure 6.2: Shot detection

6.4.1.2 Still image generation from single shots. In a second step every detected shot is reduced to one representative still image by key-frame extraction or mosaicing of images. The choice of the method depends on the results of the camera motion analysis. If the shot contains one dominant camera motion direction, i.e. a pan or a tilt, mosaicing is useful to join the whole information included in the pan or tilt in one single mosaic image. The mosaicing process is described in section 6.4.1.2.

If no dominant camera motion direction is detected one representative key-frame will be extracted. Some ideas how to obtain representative key-frames are introduced in section 6.4.1.2.

Both key-frames and mosaicing of images will be considered as a still image. Therefore the analysis of frame sequences in shots is reduced to a still image analysis described in section 6.4.1.3.

The mosaicing procedure. For a detailed shot representation it would be perfect to view a shot as a still image that contains all single frames based on the camera movements. The first step is to calculate the camera movements between each frame. Using this information it is possible to combine the frames within one mosaicing image.

The motion parameters are included in a continuous equation for the optical flow. Camera movements like tilt, pan, and zooms are detected with a projective model using only pixel information. The model contains eight parameters that can be used for the transformation equation, $[x, y]^T$ describes the pixel coordinate at time t , the vector $[x', y']^T$ at time t' .

$$X' = [x', y']^T = \frac{A[x, y]^T + \vec{b}}{\vec{c}^T[x, y]^T + 1} = \frac{Ax + \vec{b}}{\vec{c}^T x + 1} \quad (6.1)$$

The 2×2 matrix describes the rotation of the image, $\vec{b}$ and $\vec{c}$ are the parameters to describe the transformation. This transformation can be used for

deriving the model flow components. Minimizing the sum of the squared differences between flow velocity and model velocity, and expanding the results into a Taylor series using only the first three terms, this leads to a formula corresponding to the bilinear model [21].

This approach is well suitable if there exists only a global motion. Particular motion of objects inside a shot destroys the global motion property. A special segmentation of the frames and their movements can extract a moving object and a background image. With the dominant mosaicing technique [29], two images are generated: one that contains the motion objects in the foreground and one those in the background.

Figure 6.3: Mosaic image

Extraction of representative key-frames. An easy way to extract key-frames is to take every n th frame of the sequence or to take the first or the middle frame of every shot detected. But such methods do not necessary come up with the frames which are most important for the understanding of the content of the scene. As mentioned before our further content analysis is based on the still image analysis done on the key-frames extracted from each shot.

To obtain best results with the still image analysis we need key-frames which represent the key information included in the shot. Provided that the camera holds on objects or on parts of the scene which contain key information we use a heuristic to obtain significant key-frames: for each shot take the frame where the camera motion as well as the difference in chrominance and luminance to the neighboring frames is minimal.

In some cases it is not necessary to extract one frame for each shot. In an interview, for example, one shot shows the interviewer and the next one shows the interview partner and so on. In such a case one can use the results of the video clustering algorithm described in section 6.4.3 to detect such scenes and extract one key-frame for each cluster instead of one for each shot.

If the shot contains text inserts, a frame with the insert visible would be a good key-frame because the textual information often supplies an additional description of the content.

Further work has to be done to expand the set of rules mentioned above for extraction of representative key-frames. For example, as proposed in [30] the audio signal may also give some useful hints where to find important scenes within a video sequence.

6.4.1.3 Still image analysis. The still image analysis consists of three modules, each extracting one of the low level features color, texture, and contour. Three modules work independent of each other.

Color segmentation. First the image is transformed from *RGB* to *HLS* color space. *HLS* defines the color space by three parameters: hue, lightness and saturation. The main idea of our approach is to group the pixels of the image into regions with equal color, using certain thresholds for the difference between the color values *H*, *L*, and *S* of neighboring pixels. Then we calculate the circumscribing rectangle for each of the color segmented regions. The result of the color analysis consists of the color, the coordinates of the circumscribing rectangle, and its center of gravity for each color region that was found.

Texture classification. Our approach for texture segmentation is based on a region- and edge-oriented method to detect regions of homogeneous texture [16]. A rectangle is cropped out of every texture region found within the image. These texture segments are analysed by a texture analysis system [22, 11], which calculates the value of characteristics for seven visual texture properties (see Fig. 6.4). The visual texture properties are called shape of primitives, line-likeness, coarseness, regularity, directionality, contrast and softness, and allow a domain-independent description of textures [2].

Figure 6.4: Texture analysis.

We performed a significance analysis to find the most suitable statistical feature for each visual property to compute its value of characteristics from a digital grayscale image [11]. The setting of the parameters of the statistical features were evaluated in this significance analysis, too. An overview of all seven properties and the corresponding statistical features is given in Tab. 6.1. For example, to find out whether the shape of primitives of a given texture is homogeneous, multi-areas or blob-like, we use the statistical feature roughness F_{rgh} , as proposed in [33]. To estimate the roughness of a texture a statistical feature matrix is used, which calculates the difference of the gray levels for pairs of pixels having a certain distance. The maximum distance is specified by the parameters L_c and L_r . To calculate the line-likeness, coarseness, regularity, and directionality of a texture we use the statistical features F_{lin} , F_{crs} , F_{reg} , and F_{dir} defined by [32]. The main idea for the calculation of the line-likeness of a texture is based on an edge detection and a counting of edges appearing as lines. The parameter t is a threshold for the gray level difference used for edge

detection. The directionality is derived from using a histogram measuring the direction of edges in the image. If the histogram has a single peak the texture has a main direction. The value of directionality is given by the height of the peak. The regularity of a texture is calculated by measuring the variation of a certain feature over the whole texture region. The region is divided into s^2 subregions. Then the feature is calculated for each subregion and the difference of the results is used as a measure for regularity. The feature calculated for the subregions to estimate the regularity is the gray level variance. It measures the contrast of a texture. The algorithm for the calculation of coarseness is described in detail in [32, 11].

The value of the statistical feature symbolizes that of the visual properties, except for softness, where a high value for complexity (f_{com}) implies a non-soft texture whereas a low value implies a soft one. The parameter d specifies the width of the local neighborhood the neighborhood gray-tone difference matrix is calculated for. This matrix is used to estimate the statistical feature complexity [1].

Table 6.1: Significant statistical features

Visual Property	Statistical Feature	Parameter
shape of primitives		
<i>homogeneity</i>	F_{rgh} – Wu & Chen	$L_c = L_r = 2$
<i>bloblikeness</i> $\leftrightarrow$ <i>multiareas</i>	F_{rgh} – Wu & Chen	$L_c = L_r = 8$
linelikeness	F_{lin} – Tamura et al.	$t = 32$
coarseness	F_{crs} – Tamura et al.	$d = 2$
regularity	F_{reg} – Tamura et al.	$s = 4$
directionality	F_{dir} – Tamura et al.	–
contrast	graylevel variance	–
softness	f_{com} – Amadasun & King	$d = 12$

All statistical features are calculated on the original gray scale images of the textures, with the exception of directionality, where a linear histogram scaling is performed to pre-process the image before calculating the statistical feature.

The result of our texture analysis is an automatically generated texture description based on visual properties of textures. The main advantage of this approach lies in its usability for several texture domains.

Contour approximation. The detected contours are boundaries of extracted regions as results of the texture and color segmentation process. The contours are represented by a pixel sequence which will be transformed in approximated polylines:

$$\{(x_i, y_i)\}_{i=0}^N \rightarrow \{p_j\}_{j=1}^K, K \ll N. \quad (6.2)$$

Our approximation method of pixel sequence to polylines is based on the detection of Digital Straight Segments (DSS).

The main problem is to find an optimal DSS with

- minimal distance between every DSS and pixel sequence, and
- minimal number of DSS for a given pixel path.

The process is designed in two steps:

1. detection of DSS considering local pixel constellation $\{p_j\}$, and
2. approximation of polylines $\{p_j\}$ considering curvature's criterions.

6.4.1.4 Text string extraction. Additional information about the contents of a video stream is contained in text inserts superimposed on some frames.

To extract this information the following steps are necessary:

- to localize the frames with superimposed textual inserts,
- to separate text candidates from background, and
- to recognize the detected text candidates.

Problems are caused by the low resolution of the characters and the often very complex background. Lately, a number of methods have been proposed for text string extraction and character recognition for videos. One method introduced by [28] uses an interpolation filter, multi-frame integration and a combination of four filters to solve the problems addressed above. Another method proposed by [25] is based on adaptive thresholding to segment the image first in regions. Character candidate regions are then detected by observing gray level differences between adjacent regions. Another way to segment character candidate regions is to treat text as a distinctive texture. Subsequently strokes are extracted from the segmented text regions. Using heuristics on the appearance of text strings like for example height similarity and character as well as line spacing, the strokes are processed to form tight rectangular bounding boxes around the corresponding text strings. In a last step the background is cleared up and the text is binarized [34]. Both approaches use grayscale images as input. An approach which uses color images was proposed by [19]. It is based on a split and merge algorithm which first splits the image into segments of homogeneous color and then merges adjacent segments if their color difference is less than a threshold.

Our approach combines some of the ideas mentioned before. On the assumption that an insert has to be displayed for a certain minimum time to be read by the viewer, we first examine every n th frame for text inserts. If a certain frame seems to contain an insert, we examine a surrounding sequence of frames, in order to support or neglect this hypothesis.

After localizing the sequence of frames with imposed inserts, we have to separate the text strings from the background. We analyze a sequence of succeeding frames showing the same insert and combine the results to enhance the recognition.

To find the text candidate regions in a frame we make use of our color analysis algorithms described in section 6.4.1.3. The algorithm detects the characters as homogeneous color regions. In a next step we concatenate these separate regions to text strings and the strings to text blocks using the following heuristics about the appearance of text:

- all characters within one image are of the same color, and the contrast to the colors appearing in the background behind the text is high,
- the spacing between the characters of one word is always the same,
- the spacing between all words of a text string is the same,
- characters are aligned on horizontal lines, and
- the line spacing between two lines of text is constant.

After that, the image is binarized by setting all pixels belonging to text candidates to black and the background pixels to white.

The final step, i. e. the recognition of the characters in the binarized image, is done by a commercial OCR software.

6.4.2 Semantical description with graph grammars

To analyze the semantics of a video or an image we use the syntactical information extracted with the previously described methods for object recognition. In a first step we have to combine all the quantitative, syntactical informations we got until now from the different low level analysis methods to recognize atomic entities. These atomic entities are on a qualitative, semantical level, i. e. primitive objects, e.g. grass or forest on a landscape image.

The results build hypotheses related to the primitive objects. In a second step we combine the hypotheses about the primitive objects according to the compositional semantics of more complex objects by means of which the hypotheses becomes a thesis [15]. For both steps we use neighborhood-controlled node-labeled and node-attributed feature graph grammars (1-NRCFGG) as specified in [14]. The edges of the feature graph represent the topological neighborhood *nr* relations between the different image segments resulting from the image analysis, for example certain color or texture regions detected within the image. An example of a feature graph is shown in Fig. 6.5. These neighborhood relations are for example *meets*, *overlaps* or *contains*. The fundamental assumption for our object recognition is, that these relations restrict the recognition of objects, i.e. a kind of *nr*-inheritance is guaranteed. The process of object recognition can be treated as a process of graph transformation. This step is performed by a graph parser - the so called *GraPaKL* [14]. Two graph grammars are used: one to bridge the gap between lower-level information and primitive objects, and a second to combine the primitive objects in order to recognize complex objects.

Figure 6.5: Feature graph

To detect a forest scene, for example, one has to combine the primitive objects grass and forest. The primitive object forest consists of a texture, a color, and a contour segment which must hold certain conditions. The derivation from a complex object on the highest level of abstraction via primitive objects down to the low-level features is called a path through the feature graph. An example is shown in Fig. 6.6.

Figure 6.6: A path from high to low level of abstraction

6.4.2.1 1-NRCF Graph Grammars and GraPaKL. In this section we introduce 1-NRCF Graph Grammars and GraPaKL as described in detail in [10].

A finite undirected node attributed and node labeled graph, short *feature graph*, is a 6-tuple $(V, E, \mathcal{A} \cup \mathcal{R}, \psi, \pm, \varphi)$, where:

- V is a finite, not empty set of attributed and labeled nodes,
- $E \subseteq V \times V$ is a finite set of undirected edges,
- $\mathcal{A}, \mathcal{R}$ are finite sets of attribute-value pairs, where $\mathcal{A} \cap \mathcal{R} = \emptyset$,
- $\psi : V \rightarrow 2^{\mathcal{A} \cup \mathcal{R}}$ is an attribute function,
- Σ is a finite not empty alphabet of node labels, and

- $\varphi : V \rightarrow \Sigma$ is a label function.

As mentioned before, the interpretation of the edges is restricted to the topological neighborhood (nr).

A neighborhood-controlled node labeled and node attributed feature graph grammar (1-NRCFGG) is a 10-tuple $(V_{NT}, V_T, P, S, \mathcal{A} \cup \mathcal{R}, \psi, \mathcal{AGL}, \xi, \pm, \varphi)$, where:

- V_{NT} is a finite, not empty set of nonterminals, where $\forall v \in V_{NT} : \psi(v) \cap \mathcal{R} \neq \emptyset$,
- V_T is a finite, not empty set of terminals, where $\forall v \in V_T : \psi(v) \cap \mathcal{R} = \emptyset$,
- P is a finite set of productions,
- $S \in V_{NT}$ is the start node,
- $\mathcal{A} \cup \mathcal{R}, \psi, \pm, \varphi$ as previously defined,
- $\mathcal{AGL}$ is a set of conditions, so called dependency relations, for example equations or predicates etc., and
- $\xi : P \rightarrow 2^{\mathcal{AGL}}$ is a condition function.

The productions left-hand side (lhs) is described by one single node and the productions right-hand side (rhs) is given by any graph. Additional conditions, called *dependency relations* are used for proving, passing, or generating information by calculating attributes. The global embedding specification is given by the nr -inheritance.

The chart parser GraPaKL is described in detail in [14]. The main algorithm consists of the rules *initialize*, *choose* and *combine*. The input for the GraPaKL is a terminal feature graph, which contains the low-level features as terminal nodes and a 1-NRC graph grammar representing the productions. Starting from this terminal feature graph complex objects are detected.

6.4.2.2 Extension of the 1-NRCF Graph Grammar to a stochastic grammar. In object recognition there are often several alternatives to combine the low-level information in order to achieve primitive objects and later on the primitive objects in order to achieve complex objects. These different alternatives may have different probabilities. The idea is to extend the 1-NRCFG grammar in a way that for each production a certain probability is specified [20]. The probability for each parse is calculated by multiplying the probabilities of the productions that were applied to achieve parse. The probability calculated for a parse is the probability that the corresponding (primitive or complex) object has been recognized.

For each production rule of the form

$$A_j \rightarrow \beta_{j,i} \quad 1 \leq i \leq u \quad (6.3)$$

associated with it exists a probability $p_{j,i}$, and u is the number of productions with premise A_j .

A stochastic grammar is proper if

$$\sum_{i=1}^u p_{j,i} = 1 \quad \text{for all } A_j \in N. \quad (6.4)$$

The grammatical word function generated by G is defined as

$$\begin{aligned} f(v) &= \sum_{n=1}^{N(v)} \prod_{k=1}^{K(v,n)} p_{n,k}(v) & \text{if } v \in L(G) \\ &= 0 & \text{if } v \notin L(G) \end{aligned} \quad (6.5)$$

where:

- $L(G)$ Language defined by G ,
- $N(v)$ is the number of distinct leftmost derivations of v ,
- $K(v, n)$ is the number of steps in the n th derivation, and
- $p_{n,k}(v)$ is the probability of the production used at the k th step of the n th derivation.

Following this approach, by defining a mapping function δ , we extend the previously defined 1-NRCFGG, which serves the conditions above given. The likeliness of the alternative productions has to be observed by empirical analysis of the domain.

The automatically estimation of $p_{j,i}$ will be calculated by the formula:

$$p_{j,i} = \frac{2 \cdot (u - i + 1)}{u \cdot (u + 1)}, 1 \leq i \leq u \quad (6.6)$$

$$\sum_{i=1}^u p_{j,i} = \frac{2}{u(u+1)} \sum_{i=1}^u (u - i + 1) \quad (6.7)$$

The image illustrated in Fig. 6.7 represents a forest scene combined with a sea scene. The seawater reflects the sky with clouds, located in lower part of the whole scene, that is not typical for a sky scene. This picture would be never found using only 1-NRCF Graph Grammar, because the sky location is defined in upper part of the picture. By extending the corresponding rule by additional rules considering these spatial location and smoothed texture it is possible to capture these scene as forest-sky scene.

6.4.3 Cluster analysis of videos

After a shot detection the shots are grouped into clusters based on their visual similarity. A time-constraint clustering procedure is used to compare only those shots that are located inside a time range. Shots from different sections

Figure 6.7: forest-sky scene.

of the video (e.g., begin/end) are not compared. With this cluster information that contains a list of shots and their clusters, it is possible to calculate scene bounds. Labeling of all clusters gives a description of cut pattern. It is now easy to distinguish a dialogue from an action scene.

Consecutive shots in a scene are in a close semantical relation. A dialogue can be characterized by one or two opening shots followed by at least two alternating shots until the end, which adds another shot. An action scene usually contains many short shots. The same used within the cut detection algorithm can be used for the grouping procedure of shots. A cluster contains all shots with similar visual properties. An additional parameter can be used as a time constraint. Only those clusters are compared with the current one that fit into a fixed time interval [35].

A histogram of a key-frame is taken from each shot and compared with the histogram of the previous key-frames of the clusters. In the initial phase the first shot is assigned to the first cluster, the second shot to the second cluster. All succeeding shots will be compared to all previous shots of the clusters except the first previous one, because two consecutive shots are never taken from the same position or contain the same visual content. Therefore they cannot belong to the same cluster. If there is no cluster found which matches the histogram a new cluster will be generated containing the current shot. To minimize computing time only the last shot of a cluster will carry the current histogram. An example for the clustering is shown in Fig. 6.8. Each row represents the shots belonging to the same cluster.

After all shots are grouped into clusters all clusters should be checked for a scene cut. All clusters between two scene cuts are called a scene unit. Clusters are represented in a hierarchical scene transition graph. Nodes represent the clusters that contain the shots with visual similarity under a time constraint. Edges represent the connection corresponding to the story flow. Only one key-frame from a shot of each cluster will be shown in the graph.

There are three different kinds of edges:

- a directed edge that connects two scene units,
- a directed edge inside a scene unit which connects two clusters in one direction, and

Figure 6.8: Clustering of shots

- a bidirectional edge inside a scene unit which connects two clusters in both directions.

This kind of representation makes it easy to sum up the entire video at a glance. Individual scene units can be easily recognized through the clustering. Action scenes and scenes with only a few shots are represented by special patterns.

A scene contains a number of shoots that are cut into shots. These shots are found inside a start and end sequence in which the viewer of a video focuses towards a new situation. The algorithm of the clustering procedure cannot sort these separate shots into one scene unit because of the visual dissimilarity. The scene transition graph representation of a video still abstracts from the dynamic flow of the single cuts. Therefore, it is desirable to start a classification of the shot order which would make it easy to distinguish an action scene from a dialogue between a sequence of starting and a sequence of ending shots [36].

6.5 CONCLUSION

In this paper we investigate the generation of a syntactical and semantical description of video sequences as full process from low-level to high-level representation. The important advances of our system lie in the extension of uncertain domain-knowledge representation using stochastic grammars.

The use of uncertainty provides us with the possibility to estimate how sure is the object recognition and scene interpretation. Additionally, the bidirectional inference process makes it possible to continue a parser in case of deadlocks.

The syntactical analysis involves the image analysis and text recognition.

From our current experiences with audio processing and speech recognition we plan to integrate an audio analysis in our system.

The syntactical description as a time-consuming process can be created only once, since the results are deterministic and non-subjective. Therefore they are reusable for the generation of an updated version of semantical description. If novel objects or synonyms are inserted into a domain representation, the semantical description, in its turn, can be created repeatedly according to the possibly extended domain knowledge. The manual adaptation of domain knowledge is everytime allowable, especially for information updating.

Acknowledgements

This work is being supported in part by the German Research Association (DFG, contract no. HE 989/4-1)

References

- [1] M. Amadasun and R. King. Textural Features Corresponding to Textural Properties. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1264–1274, 1989.
- [2] G. Asendorf and T. Hermes. On Textures: An Approach For A New Abstract Description Language. In *Proceedings of IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology*, pages 98–106, San Jose, CA, USA, 29 January – 1 February 1996.
- [3] Avid Technology GmbH. *Cutter Guide – AVID*. Hallbergmoos, 1997.
- [4] A. Cawkell. Imaging systems and picture collection management: a review. In *Information Services & Use 12*, pages 214–220, 1992.
- [5] M. G. Christel, M. A. Smith, C. R. Taylor, and D. B. Winkler. Envolving Video Skims into Useful Multimedia Abstractions. In *Proceedings of the ACM CHI'98 Conference on Human Factors in Computing Systems*, Los Angeles, USA, April 1998.
- [6] M. G. Christel, S. Stevens, T. Kanade, M. Mauldin, R. Reddy, and H. Wactlar. Techniques for the Creation and Exploration of Digital Video Libraries. In B. Furht, editor, *Multimedia Tools and Applications*, Vol. 2. Kluwer Academic Publishers, Boston, MA, USA, 1996.
- [7] M. G. Christel, D. B. Winkler, and C. R. Taylor. Improving Access to a Digital Video Library. In *Human-Computer Interaction: INTERACT'97, the 6. IFIP Conference on Human-Computer Interaction*, Sydney, Australia, July 14–18 1997.
- [8] A. Dammeyer, W. Jürgensen, C. Krüwel, E. Poliak, S. Ruttkowski, T. Schäfer, M. Sirava, and T. Hermes. Videoanalyse mit DiVA. In *Proceedings of KI-98 Workshop Inhaltsbezogene Suche von Bildern und Videosequenzen in digitalen multimedialen Archiven*, Bremen, Germany, 16 - 17 September 1998.

- [9] A. Hampapur. Virage Video Engine. In *Proceedings of IS&T/SPIE Symposium on Electronical Imaging Science & Technology*, San Jose, CA, USA, February 1997.
- [10] T. Hermes, C. Klauck, J. Kreyß, and J. Zhang. Content-based Image Retrieval. In *Proc. of CASCON '95 (CD-ROM)*, Toronto, Ontario, Canada, 7-9 November 1995.
- [11] T. Hermes, A. Miene, and O. Moehrke. On Textures: Analysis and Description. *Image and Vision Computing (submitted to)*, 1998.
- [12] K. Hirata and T. Kato. Query by visual example. In *Proceedings of Third Intern. Conf. on Extending Database Technology*, pages 56–71, Vienna, Austria, March 1992.
- [13] B. R. Jain. NSF Workshop on visual Information Management Systems, Workshop report. Technical report, Computer Science and Engineering Division, The University Michigan, Ann Arbor, Mich., 1992.
- [14] C. Klauck. *Eine Graphgrammatik zur Repräsentation und Erkennung von Features in CAD/CAM*. PhD thesis, University of Kaiserslautern, vol. No. 66 of DISKI, infix-Verlag, St. Augustin, 1994.
- [15] C. Klauck. Graph Grammar Based Object Recognition for Image Retrieval. In *Proc. of the 2th Asian Conference on Computer Vision (ACCV'95)*, Singapore, Republic of Singapore, December 1995.
- [16] P. Kreyenhop. Textursegmentierung durch Kombination von bereichs- und kantenorientierten Verfahren. Master's thesis, University of Bremen, 1998.
- [17] J. Kreyß, T. Hermes, P. Alshuth, and M. Röper. Video Retrieval by Still Image Analysis with ImageMiner. In *IS&T/SPIE Symposium on Electronical Imaging Science & Technology (Storage and Retrieval for Images and Video Databases II)*, volume 3022, pages 36–44, San Jose, CA, Feb. 1997.
- [18] R. Lienhart, W. Effelsberg, and R. Jain. VisualGREP: A systematic method to compare and retrieve video sequences. In *Proceedings of IS&T/SPIE Symposium on Electronical Imaging Science & Technology (Storage and Retrieval for Images and Video Databases)*, volume 3312, pages 271–282, San Jose, CA, USA, January 1998.
- [19] R. Lienhart and F. Stuber. Automatic Text Recognition in Digital Videos. In *Proceedings of SPIE Image and Video Processing IV*, volume 2666, pages 180–188, September 1996.
- [20] S.Y. Lu and K.S. Fu. Stochastic Tree Grammar Inference for Texture Synthesis and Discrimination. *Computer Graphics and Image Processing*, 9(3):234–245, 1979.
- [21] S. Mann and R.W. Picard. Video Orbits of the Projective Group: A New Perspective on Image Mosaicing. Technical Report 338, MIT Media Laboratory Perceptual Computing Section, 1996.
- [22] A. Miene and O. Moehrke. Analyse und Beschreibung von Texturen. Master's thesis, University of Bremen, 1997.

- [23] G. Nagy. Image Databases. *Image and Vision*, 3(3):111–117, 1985.
- [24] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The QBIC Project: Querying Images By Content Using Color, Texture, and Shape. In *IS&T/SPIE Symposium on Electronical Imaging Science & Technology*, San Jose, CA, USA, Feb. 1993.
- [25] J. Ohya, A. Shio, and S. Akamatsu. Recognizing Characters in Scene Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):214–220, 1994.
- [26] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. In *Proceedings of IS&T/SPIE Symposium on Electronical Imaging Science & Technology (Storage and Retrieval for Images and Video Databases II)*, San Jose, CA, USA, Feb. 1994.
- [27] Y. Rui, T. S. Huang, and S. Mehrotra. Relevance Feedback Techniques in Interactive Content-Based Image Retrieval. In *Proceedings of IS&T/SPIE Symposium on Electronical Imaging Science & Technology (Storage and Retrieval for Images and Video Databases)*, volume 3312, pages 25–36, San Jose, CA, USA, January 1998.
- [28] T. Sato, T. Kanade, E. Hughes, and M. Smith. Video OCR for Digital News Archives. In *IEEE Workshop on Content-Based Access of Image and Video Databases (CAIVD'98)*, Bombay, India, January 1998.
- [29] H.S. Sawhney, S. Ayer, and M. Gorkani. Dominant and Multiple Motion Estimation for Video Representation. In *IEEE International Conference on Image Processing, ICIP'95*, pages 322–325, Washington, D.C., Oct. 1995.
- [30] M. Smith and T. Kanade. Video Skimming and Characterization through the Combination of Image and Language Understanding. In *IEEE International Workshop on Content-Based Access of Image and Video Databases*, Bombay, India, 1998.
- [31] J. Song and B.-L. Yeo. Spatially Reduced Image Extraction from MPEG-2 Video: Fast Algorithms and Applications. In *Proceedings of IS&T/SPIE Symposium on Electronical Imaging Science & Technology (Storage and Retrieval for Images and Video Databases)*, volume 3312, pages 93–107, San Jose, CA, USA, January 1998.
- [32] H. Tamura, S. Mori, and T. Yamawaki. Textural Features Corresponding to Visual Perception. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-8:460–473, 1978.
- [33] C.-M. Wu and Y.-C. Chen. Statistical Feature Matrix for Texture Analysis. *CVGIP: Graphical Models and Image Processing*, 54(5):407–419, 1992.
- [34] V. Wu, R. Manmatha, and E.M. Riseman. Finding Text In Images. In *20th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, Philadelphia, USA, 1997.

- [35] M.M. Yeung and B.L. Yeo. Time-constrained Clustering for Segmentation of Video into Story Units. In *International Conference on Pattern Recognition*, August 1996.
- [36] M.M. Yeung and B.L. Yeo. Video Content Characterization and Compaction for Digital Library Applications. In *IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases*, volume 3022, pages 45–58, San Jose, CA, Feb. 1997.