VIDEX: An Integrated Generic Video Indexing Approach

Roland Tusch, Harald Kosch, and Laszlo Böszörmenyi
Department of Information Technology
University Klagenfurt
9020 Klagenfurt, Austria
{rtusch, harald, laszlo}@itec.uni-klu.ac.at

ABSTRACT
This paper presents an integrated generic technique for low- and high-level video indexing. The proposed approach tries to integrate the advantages of existing low- and high-level video indexing approaches by reducing their shortcomings. Furthermore, the model introduces concepts for a detailed structuring of video streams, and for correlations of low- and high-level video objects. The proposed model is called generic, as it only defines a framework of classes for an integrated video indexing system. It has been verified by implementing a prototype of a distributed multimedia information system supporting content-based video retrieval.

Categories and Subject Descriptors
H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*abstracting methods, indexing methods*

1. INTRODUCTION
In recent years a number of video indexing models have been proposed, mostly covering either low-level (physical) ([14],[4]) or high-level (semantic) visual features ([6]) of video streams. Dealing only with physical or semantic indexes has certain disadvantages. On the one hand, low-level visual features can only provide very preliminary video classification and do not allow high-level content-based video queries. On the other hand, high-level semantic-based indexes, not being based on corresponding low-level video features, may lead to semantic inconsistencies and to exhaustive manual indexing work. Inconsistencies may appear through biased subjective content interpretations of different people during the video indexing process. Some approaches even tried to index videos by both physical and semantic features ([7],[8]). They provide integrated video indexing schemes which do not lack the shortcomings of the separate ones. But there are still some weaknesses concerning the segmentation of video streams and the high-level annotation process.

The main contribution of this paper is the definition of an integrated generic video indexing model based on low-level visual features and typed, high-level semantic objects. The proposed model allows the integration of a number of existing tools during the video indexing process. First, automatic and manual methods for segmenting video streams can be used to give videos a hierarchical structure. Second, mechanisms for extracting and tracking low-level visual objects need to be used to physical index videos. The extracted objects may be spatially, temporarily or spatio-temporarily related to each other. And third, tools for grouping low-level objects together to typed high-level, semantic objects are supported. These semantic objects can further be related to physical or logical video segments.

2. THE VIDEX MODEL
Our integrated generic video indexing model, presented in figure 1, integrates low- and high-level content information by picking up the advantages of existing models and reducing their shortcomings. Concerning content descriptions it is similar to the MPEG-7 standard. The differences are first, that our indexing model denotes a generic object-oriented model for low- and high-level video indexing, defining an indexing framework. The data model can be easily extended for a specific application domain, as we will demonstrate it on the example of the soccer domain in section 3. Second, we explicitly specify different granularities of video segments for video structuring. Here we take advantage of concepts of the *VideoSTAR* model [6]. Furthermore, we also provide a video hyper-link concept similar to the one presented in [7]. And finally, we integrate concepts of the MPEG-4 [9] standard in our model to be able to specify content-annotations to elementary video streams. The proposed model is a successor of the generic video indexing model presented in [12], which primarily focuses on high-level, semantic-based indexes.

2.1 The Video Segmentation Part
One main drawback of related approaches is that in most of them scenes are the only possibility to structure video streams. However, scenes do not form the basic structure of videos as they denote video units, which already incorporate the underlying semantics of physical video streams. As proposed in [6], [8] and [11], a multi-level video structuring approach should rather be followed. The higher the level, the more semantic information is covered.

Our model defines a video stream to be physically structured by so-called *PhysicalVideo Units*. A physical video unit

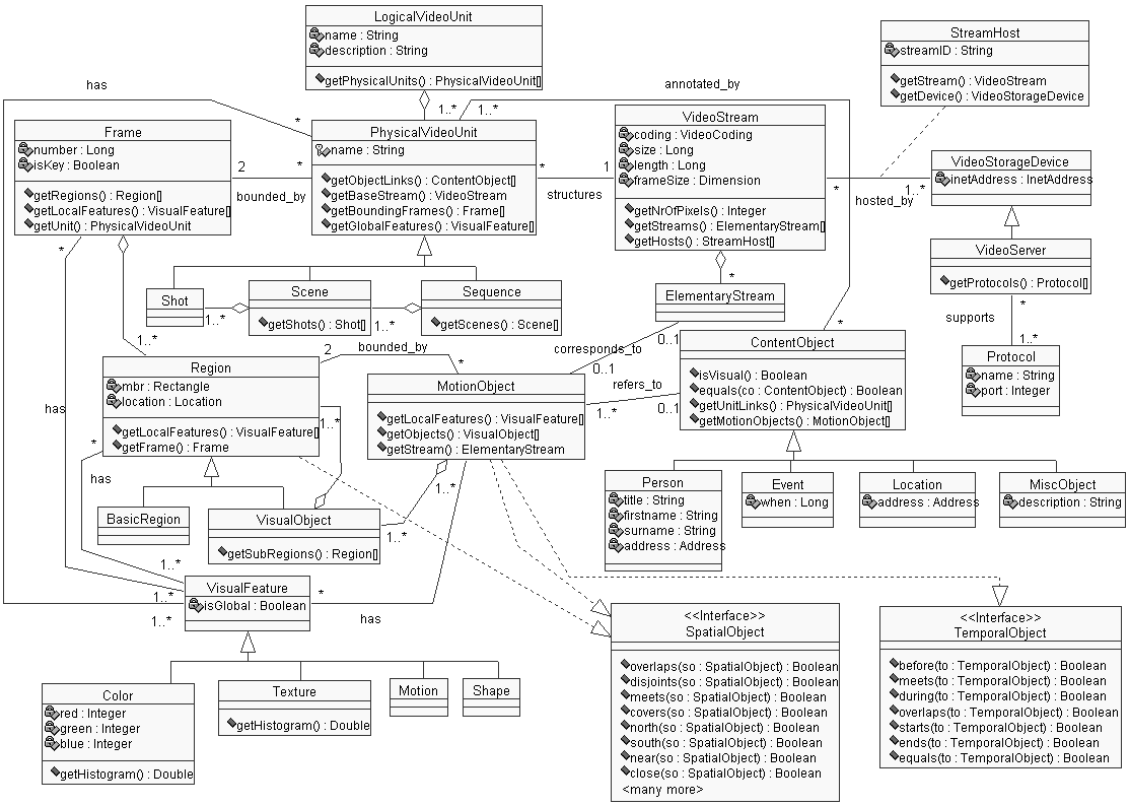

Figure 1: The generic VideX data model in UML notation

specifically can be a *Shot*, a *Scene* or a *Sequence*. Shots, which are the smallest physical video units, consist of one or more consecutively generated and recorded frames, representing continuous actions in time and space [3]. A continuous usually is marked by physical boundaries like camera breaks and editing points. These boundaries are the most important shot transition indicators used by automatic shot boundary detection algorithms. Shot boundaries are usually represented by their first and last frames, denoting two of their key frames.

A key frame, represented as class *Frame* with key-property in the *VideX* model, is a video frame representing a still image including spatial features of its contained video objects. These features can be used for satisfying spatial conditions in video queries. In our model, frames are not considered as the smallest physical video unit, as they do not have a temporal extent. Frames only allow the definition of spatial relations between sub-objects not regarding temporal or spatio-temporal relations between them.

Shots are not sufficient for a meaningful video browsing and retrieval for mainly two reasons [11]. First, there are too many shots in a long video to be presented to the user. And second, shots do not capture the underlying semantic structure of a video since users might want to browse or retrieve videos at a semantic level, and not by physical shots or key frames. Therefore, all shots, which are related semantically in time and space, are grouped together to form the *scene* level.

Scenes represent the first abstraction level which take into account the underlying semantics of video streams. They are marked by semantic boundaries, which makes the scene boundary detection a much more difficult task in comparison with shot boundary detection. However, in [11] an algorithm for automatic construction of scene structures is proposed. Scenes deliver very usable video units for retrieval in content-based video queries, due to their ability in structuring large video streams by semantic content.

In some cases it might be necessary to relate several scenes to *sequences*. As a scene only represents semantically related shots, the granularity might even be too small for browsing very long video streams effectively. Consider e.g. a video capturing a whole soccer match with a duration of at least 90 minutes and a lot of events occurring in it. Within this domain one shot could be e.g. the happening of a foul from player A on player B, another shot could be the execution of the following free-kick which results in scoring a goal. Both shots can be semantically grouped together to form a goal scene. Now, if someone would also be interested in some previous actions and events in the match, which lead to this foul, all the previous actions forming several scenes should be returned together with the goal scene. The whole returned unit denotes a sequence consisting of scenes, which together give a meaning [6]. The definition of sequences needs human assistance due to the very difficult task of keeping sequences semantically consistent.

For video authoring, our model provides the concept of log-

ical video units (class *LogicalVideoUnit*). They allow the assembling of different physical video units to form user-created video documents. Thereby, the physical video units may belong to more than one physical video stream.

To be able to construct a reliable and consistent video structure, it is necessary that the low-level video abstractions are made accurately. For this reason as well as for performing the video segmentation process in a more efficient way, the shots and maybe the scenes should be automatically extracted from the physical video streams. Therefore appropriate shot and scene boundary detection algorithms have to be used. The effectiveness of the algorithms concerning accuracy and robustness depends on the kind of video being segmented [2]. In the case of videos with abrupt camera transitions as e.g. in medical videos, where the noise level caused by camera and object motion is low, any of the known shot detection algorithms can be used. However, in videos with many gradual transitions and object motions the histogram-based approach using twin-comparison is preferable [13]. Histogram-based shot detection techniques give a good tradeoff between robustness and speed.

As extension to the video segments described above, our model provides means to integrate elementary video streams of the MPEG-4 standard described in [9]. An elementary stream in terms of the MPEG-4 standard is a coded data stream that can belong to a media object within an audio-visual MPEG-4 scene. Thus, one media object may consist of one or more coded elementary streams, which contain the coded physical object or some parts of it. In our model, elementary streams can be related to tracked objects over time, which further can correspond to high-level, semantic-based objects.

2.2 The Low-level Video Indexing Part

A number of existing video indexing techniques stop at the frame-level rather than exploring its sub-frame granularities. This shortcoming disables content-based video queries including spatial and spatio-temporal constraints on low-level visual objects.

In our model, the class *Frame* denotes the entry point for low-level video access. Within a physical video unit, a frame is uniquely identified by its number and may represent a key-frame. It consists of one or more regions (abstract class *Region*), which specifically can be *BasicRegions* or *VisualObjects*. A region is a spatial object, as it implements the interface *SpatialObject*. This interface defines a set of methods for implementing qualitative spatial relations including topological, directional and distance relations. Topological relationships between spatial objects are defined by neighbor or incidence relations between them. Two spatial object may i.e. intersect (*overlap*), touch externally (*meet*) or internally (*covers*), or be disjoint. A complete set of topological relationships between two spatial objects, called the *4-intersection-model*, is presented in [5]. Directional relations describe the relative positions between two given spatial objects. In [10], a complete set of 169 directional relationships is presented. It includes e.g. *north*, *south*, *above-left* and *above-right*. And finally, the distance relations describe the space range between spatial objects, including i.e. *far*, *near*, *close*. In the *VideX* model, regions are currently geometrically represented by *minimum bounding rectangles* (MBRs), as this kind of object approximation needs only a few points for their representation. Furthermore, MBR approximations are used to efficiently retrieve candidates that could satisfy a spatial query.

A *BasicRegion* in our model is a region which can be considered as being atomar, i.e. it does not consist of any other sub-regions. If a region represents an aggregation of sub-regions, then it is called a *VisualObject*. Such a visual object could e.g. be a person, where the visual object *person* consists of two visual sub-objects *head* and *body*. The body further could consist of the regions *hands*, *trunk*, and *feet*. As a visual object is a special type of a region, it also implements the interface *SpatialObject*.

The concept of visual objects is required for tracking compound objects over time, which results in *MotionObjects*. A motion object denotes a sequence of visual objects and is bounded by a starting and ending visual object. As a visual object may consist of visual sub-objects, a motion object implicitly may consist of sub-motion objects. Using the previous example, a person-object could be tracked over time. As the person consists of the sub-objects *head* and *body*, implicitly two sub-motion objects would be tracked. As motion objects are tracked objects over time, they also have a temporal extent. Therefore, the class *MotionObject* implements the interface *TemporalObject*. This interface declares a set of methods for implementing temporal interval relations between temporal objects. The interface implicitly includes the 13 temporal interval relations proposed in [1], although it only defines seven. The basic temporal interval relations are: *before*, *meets*, *during*, *overlaps*, *starts*, *ends* and *equals*. All temporal relations except *equals* have inverse ones, which are implicitly covered by the return values of the methods.

2.3 The High-Level Video Indexing Part

The high-level indexing part of the model denotes that set of classes, which specifies typed, semantic-based content annotations to physical video units. These annotations are necessary since users usually do not want to search for videos by specifying constraints on their low-level visual features. Furthermore, content annotations allow a semantic-based browsing of video streams, not having to deal with unrelated sequences of key frames or shots.

In our *VideX* model, the basic class for content annotations is the *ContentObject* class. It represents an abstract class providing elementary functionalities for an arbitrary typed content object. It may refer to a low-level motion object for dealing with spatial, temporal or spatio-temporal characteristics of its physical pendant. As in most video streams, in which humans usually are interested in, objects like persons, locations and events are contained, the model even keeps track of these objects in its generic form. Furthermore, we are providing a class *MiscObject*, which can be any miscellaneous object that cannot be categorized as person, location or event. These four elementary concrete classes define the entry points for further specialization in specific application domains. In our prototype implementation, which is discussed in section 3, we illustrate the specialization in the case of soccer videos.

The model also includes the concept of video hyper-links, as described in [7], by relating content objects to physical or logical video units. There are mainly two different types of hyper-links. First, all content objects related to the same physical unit are hyper-linked together. These hyper-links are called *object-links* and delivered by method *getObjectLinks()* of class *PhysicalVideoUnit*. And second, all physical video units referenced by the same content object are hyper-linked together. These links are called *unit-links* and returned by method *getUnitLinks()* of class *ContentObject*. There are implicitly two further types of hyper-links, namely object-links of logical video units and logical unit-links of content objects. Video hyper-links allow a non-linear browsing of video story units. A non-linear browsing is much more effective and flexible than sequential approaches based on key-frames are.

3. THE SOCCER-BASED PROTOTYPE IMPLEMENTATION

We implemented a prototype of a distributed multimedia information system based on soccer videos. The system consists mainly of three interacting parts. First, a parallel video server providing selective access to physical video streams and their segments. Second, an index database capturing actually the segment and high-level indexing-part of the *VideX* model contains structural and semantic meta-information about the stored soccer streams. The video indexing model for the soccer domain extends the high-level indexing part of the generic *VideX* model with a rich set of soccer-specific classes. There are special events like e.g. *goals*, *free kicks* and *fouls* and special persons like *players* and *referees*. And third, a Java-based client provides users with graphical user-interfaces for video annotation, query specification and parallel browsing the video segments of the query results. Users may run queries against the system like:

"Find all video sequences, where a given player A scored a goal by head after a cross from the right over at least 30 meters, executed by player B."

The Java-based client requests the retrieved video segments from the video server and presents the received segment streams to the user in a parallel fashion. This is achieved by using Sun's *Java Media Framework*.

4. CONCLUSION

The main contribution of this paper to the research area of multimedia systems was the presentation of an integrated generic approach for indexing video streams. The proposed indexing model, called *VideX*, integrates both low- and high-level visual objects. Thereby, it tries to pick up the advantages of existing models and to reduce the number of their possible shortcomings. Additionally, it introduces extensions for structuring video streams and correlations of low- and high-level video objects. Video units are annotated by typed semantic video objects, as they facilitate an exact query formulation by using standard query languages. The *VideX* model is called generic, as it defines only the basic classes for an integrated video indexing system. It can easily be extended for specific application domains.

Future work concerns the integration of low-level features in the indexing part, as they are actually not contained by the database. To reduce semantic ambiguity during the annotation process, we will provide access to all low-level visual and motion objects known by the database. Then users will be able to associate directly high-level objects with low-level ones.

5. REFERENCES

[1] J.F. Allen, *Maintaining knowledge about temporal intervals*, Communications of the ACM **26** (1983), no. 11, 832–843.

[2] J.S. Boreczsky and L.A. Rowe, *A comparison of video shot boundary detection techniques*, Journal of Electronic Imaging **5** (1996), no. 2, 122–128.

[3] G. Davenport, T.G.A. Smith, and N. Pincever, *Cinematic primitives for multimedia*, IEEE Computer Graphics and Animation, Special Issue on Multimedia, 1991, pp. 67–74.

[4] Y. Deng and B.S. Manjunath, *NeTra-V: Toward an object based video representation*, IEEE Transactions on Circuits and Systems for Video Technology **8** (1998), no. 5, 616–627.

[5] M. Egenhofer and R. Franzosa, *Point-set topological spatial relations*, International Journal of Geographic Information Systems **5** (1991), no. 2, 161–174.

[6] R. Hjelsvold and R. Midtstraum, *Modeling and querying video data*, Proceedings of the 20th VLDB Conference, 1994, pp. 686–694.

[7] H. Jiang and A.K. Elmagarmid, *Spatial and temporal content-based access to hypervideo databases*, The VLDB Journal **7** (1998), no. 4, 226–238.

[8] International Standardization Organization, *MPEG-7 context and objectives*, ISO/IEC JTC1/SC29/WG11 N2460, October 1998.

[9] _____, *MPEG-4 overview (melbourne version)*, ISO/IEC JTC1/SC29/WG11 N2995, October 1999.

[10] D. Papadias and Y. Theodoridis, *Spatial relations, minimum bounding rectangles, and spatial data structures*, International Journal of Geographic Information Systems **11** (1997), no. 2, 111–138.

[11] Y. Rui, T.S. Huang, and S. Mehrotra, *Exploring video structure beyond the shots*, IEEE International Conference on Multimedia Computing and Systems, 1998, pp. 237–240.

[12] Roland Tusch, *Content-based indexing, exact search and smooth presentation of abstracted video streams*, Master's thesis, Klagenfurt University, Department of Information Technology, 1999.

[13] H.J. Zhang, A. Kankanhalli, and S.W. Smoliar, *Automatic partitioning of full-motion video*, ACM Multimedia Systems **1** (1993), no. 1, 10–28.

[14] D. Zhong and S.-F. Chang, *Video object model and segmentation for content-based video indexing*, IEEE International Symposium on Circuits and Systems (ISCAS'97), 1997, pp. 1492–1495.