A Framework for Multimedia Content Abstraction and its Application to Rushes Exploration

Werner Bailer and Georg Thallinger
JOANNEUM RESEARCH, Institute of Information Systems & Information Management
Steyrergasse 17
8010 Graz, Austria
{firstname.lastname}@joanneum.at

ABSTRACT

Multimedia content abstraction methods such as summarisation, skimming and browsing are of growing importance for the exploration of multimedia collections, complementing search and retrieval approaches. The methods proposed in literature differ in a number of aspects (e.g. form, purpose, presentation), but nonetheless many similarities in the creation of multimedia content abstractions can be identified. Thus we propose a generic multimedia content abstraction process and a software framework that implements it. We demonstrate the practical usability of the framework by using it to build an application for browsing a collection of rushes.

Categories and Subject Descriptors

H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.2.4 [**Database Management**]: Systems—*Multimedia databases*

General Terms

Summarisation, content abstraction

Keywords

Content abstraction process, rushes, browsing, MPEG-7, post-production

1. INTRODUCTION

With the increasing amount of multimedia data being produced, there is growing demand for more efficient ways of supporting exploration and navigation of multimedia data. Viewing complete multimedia items in order to locate relevant segments is prohibitive even for relatively small content sets due the required user time for viewing and the amount of data that needs to be transferred. Classical search and retrieval approaches require sufficient metadata to index the content and the feasibility to formulate a query in terms of the available metadata. Manual annotation of the content yield semantically meaningful metadata that can be effectively searched by human users, but at high annotation costs. Automatic metadata extraction approaches are in many cases not able to fully capture the semantics of the content, which makes the metadata difficult to query.

Multimedia content abstraction methods are complementary to search and retrieval approaches, as they allow for exploration of an unknown content set, without the requirement to specify a query in advance. This is relevant in cases where only few metadata are available for the content set, and where the user does not know what to expect in the content set, so that she is not able to formulate a query. In order to enable the user to deal with large content sets, it has to be presented in a form which facilitates its comprehension and allows to quickly judge the relevance of segments of the content set. Media content abstraction methods shall (i) support the user in quickly gaining an overview of a known or unknown content set, (ii) organise content by similarity in terms of any feature or group of features, and (iii) select representative content for subsets of the content set that can be used for visualisation.

The term *video abstract* is defined in [15] as "a sequence of still or moving images presenting the content of a video in such a way that the respective target group is rapidly provided with concise information about the content while the essential message of the original is preserved". The authors of [21] use the term *video abstraction* to denote all approaches for the extraction and presentation of representative frames and for the generation of video skims. In this paper, we use the term *multimedia content abstraction* in order to include also other media types beyond video. It refers to all approaches that aim at providing condensed representations of segments of a single media item or a collection of items, that are relevant or salient with respect to a certain context—independent of the purpose, context, form, creation method and presentation style of the abstract. Despite their differences in all those aspects, the existing approaches for creating multimedia content abstractions share a number of similar steps which allow to define a common process model.

Based on the definition of a process model for multimedia content abstraction, we aim at defining a software framework supporting different content abstraction approaches. The framework shall be flexible enough to allow for the variability in terms of the aspects mentioned above. While the process definition can be transformed into a common archi-

CIVR'07, July 9–11, 2007, Amsterdam, The Netherlands

tecture and data flow in the framework, the implementation of some of the framework's components will be specific to the content abstraction approach chosen for a certain application.

In the following section we discuss different aspects of multimedia content abstraction that discriminate the approaches presented in literature and refer to some examples. Based on the similarities between these approaches, we define a generic process for the creation and presentation of multimedia content abstracts. Section 3 presents a software framework for content abstraction applications and Section 4 describes its use for browsing rushes data in the TRECVID 2006[1] rushes task. Section 5 concludes with a discussion of the results and future work.

2. ANALYSIS OF MULTIMEDIA CONTENT ABSTRACTION APPROACHES

Many multimedia content abstraction approaches have been proposed in literature (a comprehensive overview and comparison of video abstraction methods can be found for example in [21]). There are a number of aspects that discriminate these approaches. In the following we discuss some of them, focusing on those that influence the design of our software framework, and define a common process for content abstraction.

2.1 Aspects of Content Abstraction

First of all, content abstraction can be done manually, automatically or semi-automatically (e.g. using user input to define examples of relevant content segments [14]). A basic aspect when creation the abstract is its **purpose**, which can be to objectively summarise the content conveying all of the original message or to deliberately bias the viewer (e.g. when creating a movie trailer, cf. [11]). Somewhat related to the purpose is the **context** of the abstract, which may be undefined and independent of the initial input of the user (e.g. when a user starts browsing), it can be defined by user input, or it can be predefined, for example, when summaries are used for representing search results and the user's query is known [6]. Domain knowledge also contributes to the definition of the context, as it helps defining the relevance of content segments. Most video abstraction approaches for sports broadcasts exploit this knowledge (e.g. goal scenes in soccer games are relevant).

A number of aspects are related to the media type of the content to be extracted. The **dimension** of the content may be a single media item (e.g. one video) or a collection of items (an example for the visualisation of a content set is presented in [16]). In the latter case all items may be of the same or of different types (e.g. a mixed collection of still images and videos). The media type also determines whether the content set has a defined **order**. A video or audio stream has an intrinsic temporal order, while a collection of still images may be ordered or unordered (for example, the still images taken by someone walking around with a camera have a temporal order like a video [13]). An existing temporal order is often kept in the abstract, but can be also discarded (e.g. [11]).

One of the most important aspects is the **content structure**. In [23] the authors discriminate *scripted* (such as movies) from *unscripted* content (such as sports video, surveillance video, home videos). Of course, the boundaries between the two are very fuzzy. Another dimension of structure is *edited* vs. *unedited* ("rushes", "raw video") content. While some content is not intended to be edited (e.g. surveillance video), there exist rushes for both scripted and unscripted content. For edited scripted content the abstraction algorithm can attempt to detect and use the structure of the content (such as dialogs [11]), while for unscripted (and especially also unedited) content other approaches are required (e.g. [5]). Content structure does not only exist on the level of the single media item, but also on the level of the collection in the case of multi-item summaries. In some cases the collection has a "macro-structure", such as a set of rushes produced according to a script.

There is a big variety of approaches for the **presentation** of abstracts. It can be interactive or non-interactive, sequential or hierarchical, and different media types and visualisations can be used. Typical ways of presentation are static visualisations of representative frames (using different visualisations such as story boards or comic book style [22]), hierarchical static or navigatable visualisations of representative frames (e.g. [24]) and video skims [19].

Unified frameworks have been proposed, mostly to integrate summarisation and retrieval (e.g. [17, 23]), but they are limited to some types of media (e.g. only video), to only scripted or only unscripted content, or they only support certain presentation forms (such as skims [20]). The novelty of our approach is to define a generic process that serves as a basis for a framework supporting variability in all the aspects discussed above.

2.2 Multimedia Content Abstraction Process

As we aim to develop a software framework supporting the creation of content abstractions independent of media type, context, presentation (interactive and non-interactive) and visualisation, we have analysed the commonalities of different approaches to multimedia content abstraction. In [21] the authors have defined a generic five step process for video skimming, and have also identified four steps in clustering-based approaches to extracting representative frames. Using this as a starting point, we define a more general process for abstracting multimedia content and discuss the stages of the process in the following.

Design The first stage deals with the conceptualisation of the content abstraction and makes basic decisions about its purpose and form. If the work is done manually, this involves a creative intervention by the user. If it is done automatically, many of these decisions may have been already taken by the developer of the application, and they are hard-wired or depend on the application's state and context.

Selection This step selects relevant segments or groups of segments according to a defined set of criteria. If these criteria have been specified by the user or are known from the application context, the selection step can be performed before clustering (e.g. sports highlights extraction). In other cases, the selection step is performed after clustering, selecting relevant clusters instead of segments. In many automatic summarisation processes the selection criteria are a result of clustering, for example, outliers such as unusual events in

[1]TREC Video Retrieval Evaluation.
http://www-nlpir.nist.gov/projects/trecvid

the content are found relevant to be included in the summary (e.g. when summarizing surveillance video).

Clustering In this stage, similarities within the content set are found and content segments that are related in terms of some feature are grouped. If selection has been performed before, the selected subset of content segments is used as an input, otherwise clustering is performed on the whole content set. Clustering is a key step in content abstraction, as it is crucial for the reduction of redundancy.

Presentation In order to visualise and/or auralise the selected groups of content segments, either new media items are created (e.g. mosaics of a shot [9], plots of a data space, time lines) or representative segments are used (e.g. representative frames for a set of video segments, a short clip). The media items representing groups of content segments are organised according to the layout of the presentation, forming a new multimedia document.

Consumption If the result of the content abstraction is non-interactive (e.g. video skim, movie trailer), the consumption step only consists of viewing the document (and possibly navigation using the player controls). In the interactive case, the user selects a subset of the content segments and maybe also changes further parameters, thus altering the input for selection and clustering. The result of re-running the creation process is an updated presentation that better suits the user's needs and interests.

3. A FRAMEWORK FOR MULTIMEDIA CONTENT ABSTRACTION

Based on the analysis of the multimedia content abstraction problem presented in Section 2 we define a software framework that implements the proposed process, providing as much flexibility as possible in terms of the different aspects of the creation of the abstract.

3.1 Implementing the Abstraction Process

The core steps of the abstraction process, i.e. preprocessing, selection, clustering and presentation, can be mapped to tasks in the software framework in a straight forward fashion. This is more difficult for the first (design) and last (consumption) step, as they are much more dependent on the specific application.

A large part of the design step consists of the basic decisions for a certain application, so that it is out of the scope of a generic framework. Instead, there is the technical need for a **preprocessing** phase in the software implementation of the abstraction process. This step ingests material into the system, performs the required content analysis and annotation and prepares the data structures (e.g. indices) that are required by the following selection and clustering operations. The preprocessing phase is directly influenced by the design step of the process, as the decisions taken there determine the features and annotations needed and thus the content analysis operations that have to be performed.

As mentioned above, the order of the **selection** and **clustering** steps in the workflow may depend on the application. In many cases, these two steps are executed iteratively, for example, to automatically build a hierarchical summary or using user input between the iterations in interactive browsing applications. Clustering is of course very feature dependent and thus the framework can only provide an interface for this task. The basic operation of the selection step is the reduction of the current content set to a subset. If the input for the selection step is determined automatically, this is again a feature dependent part.

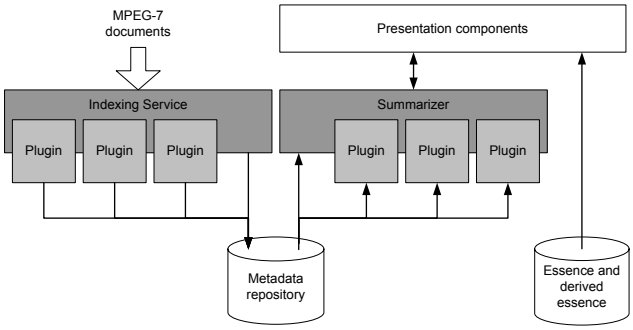

Figure 1: The components of the multimedia content abstraction framework.

The **presentation** step has both a feature specific and an application specific part. The selection of representative media items or the creation of a visualisation of a segment depends on the feature and is also often tied to the clustering and selection approaches. The framework thus only provides interfaces for this step. The rest of the presentation step, such as the layout of a number of representative frames or the editing of a skim, is application dependent, and thus outside the scope of the framework. The same is true for the consumption step.

3.2 Components

This section describes the manifestation of the functionalities of the framework in software components. The framework defines interfaces for all these components. For those which are feature independent, the framework also provides a concrete implementation. The feature specific components are implemented as plug-ins, allowing to easily add new features or change the implementation for a certain feature. Figure 1 visualises the components of the framework.

3.2.1 Data Store

The data store is a basic infrastructure component for managing the media items under control of the content abstraction application. Essence and derived essence created by automatic content analysis (such as for example representative frames) are stored in the file system. The complete metadata descriptions are stored as MPEG-7 documents in the file system as well. In addition, more efficiently searchable index structures are kept for those metadata items that are needed for clustering and selection. When possible they are kept in a relational database. For simplicity, we currently use SQLite[2], but if necessary, a more powerful database system could be integrated instead. If a feature requires specific index structures that cannot be easily represented in a relational database system (e.g. indices for visual similarity search), the appropriate index structures are stored

[2]http://www.sqlite.org

separately in the file system.

3.2.2 Indexing Service

The indexing service is responsible for ingest of content into the abstraction system. It does not perform content analysis itself, as legacy metadata or manual annotations might be available. The input to the indexing service are MPEG-7 descriptions conforming to the MPEG-7 Detailed Audiovisual Profile [1]. In the current implementation, the service watches a directory for new MPEG-7 descriptions. The ingest could also be triggered by other mechanisms, for example a Web Service call. The indexing service processes the metadata descriptions and fills the index data structures. The core implementation of the service just performs feature-independent tasks such as registering new content, while plug-ins are invoked for all other tasks.

Feature specific indexing is performed by a set of indexing plug-ins. The plug-in extracts the feature related information from the MPEG-7 documents and creates the necessary database and/or index structure entries. A plug-in will also create additional tables in the database or index structures if they are not yet there (i.e. if the plug-in for the feature has not been used before). This increases the flexibility of the framework, as new indexing plug-ins can be easily registered with the indexing service and will be used for all further incoming documents. In order to speed up clustering in the summariser, the indexer plug-ins for some features also create and update additional information such as a table of mutual similarities of descriptors in the documents indexed so far.

3.2.3 Summariser

The summariser is the component handling clustering, filtering and selection of representative media items. It accesses the data structures created and filled by the indexing service and has a generic interface towards the presentation layer in order to allow the use of different visualisation and interaction paradigms. The functionality of the core implementation of the summariser is mainly that of a broker, as—like in the indexing service—all feature-specific tasks are delegated to a set of plug-ins. The summariser has a state that is defined by the current data set and its cluster structure. It also keeps a history of the clustering and selection operations carried out so far, as well as their parameters. This allows implementing undo and redo functionality in interactive applications, as well as storing the users' browsing trails in order to improve the clustering and selection algorithms.

Each plug-in provides the following functionality for one feature: the clustering algorithm and optionally algorithms for selecting a subset of the current data and for selecting/creating representative media items for a data set. The framework defines interfaces for all three types of algorithms. For the two latter ones, simple feature-independent default implementations are provided by the framework, but a plug-in can override them. It is possible to provide multiple plug-ins for one feature, for example, to experiment with different clustering algorithms.

4. RUSHES BROWSING APPLICATION

This section describes the use of the proposed framework in an interactive application for rushes browsing. The application scenario is that of content management in the post-production phase of audiovisual media production[3]. In post-production environments, users typically deal with large amounts of audiovisual material, such as newly shot scenes, archive material and computer generated sequences. A large portion of the material is unedited and often very redundant, e.g. containing several takes of the same scene shot by a number of different cameras. Typically only few metadata annotations are available (e.g. which production, which camera, which date). The goal is to support the user in navigating and organizing these material collections, so that unusable material can be discarded, yielding a reduced set of material from one scene or location available for selection in the post-production steps.

From the user's point of view, the basic difference between content browsing and search & retrieval is the limited knowledge of the user about how to formulate a query and about what to expect from the content set. Thus, the content browsing tool must support the user in building a query step by step, by trying to add new restrictions and reducing the content set when applying the chain of restrictions built up so far (cf. the ostensive model of developing information needs [4]).

The basic workflow in the browsing tool is as follows: the user starts from the complete content set. By selecting one of the available features the content will be clustered according to this feature. Depending on the current size of the content set, a fraction of the segments (mostly a few percent or even less) is selected to represent a cluster. The user can then decide to select a subset of clusters that seems to be relevant and discard the others, or repeat clustering on the current content set using another feature. In the first case, the reduced content set is the input to the clustering step in the next iteration. The user can select relevant items at any time and drag them into the result list.

The rest of this section describes the features that have been used in the rushes application (and for which plug-ins for the framework have been implemented) and discusses the application's user interface.

4.1 Features

This section describes the features that are used in our video browsing tool. The feature extraction is performed before ingest and produces one metadata description per video conforming to the MPEG-7 Detailed Audiovisual Profile [1]. The MPEG-7 descriptions are then ingested into the system as described in Section 3.2.2. The plug-in for each feature implements a specific clustering algorithm which is also discussed in the following.

First shot boundary detection and representative frame extraction are performed. The results of this step are used as prerequisites for the extraction of other features discussed below and for visualisation. As shot boundaries are natural limits of the occurrence of most visual features (such as camera movements, objects), they are an important prerequisite for further visual feature extraction algorithms. For each shot, a number of representative frames is selected. The selection of representative frame positions is based on the visual activity in the material, i.e. the more object and/or camera motion, the shorter the time interval between two

[3]The application has been developed within the IP-RACINE project (http://www.ipracine.org) which aims at improving the digital cinema production workflow, amongst others by better tools for handling digital essence and metadata.

representative frame positions.

4.1.1 Camera Motion

The use of camera motion as a browsing feature is twofold: It is often used to guide the user's attention and express relevance of certain parts of the scene, for example, zooming on an object or person is an indicator of relevance, and in field sports, pans indicate the direction of the game. Secondly, it is an important selection criterion during editing, as visual grammar imposes constraints on the camera motion of sequences to be combined. The extraction algorithm (described in detail in [2]) is based on feature tracking, which is a compromise between spatially detailed motion description and runtime performance. The feature trajectories are then clustered by similarity in terms of a motion model and the cluster representing the global motion is selected. Camera motion is described on a sub-shot level. A new camera motion segment is created, when there is a significant change of the camera motion pattern (e.g. a pan stops, a zoom starts in addition to a tilt). For each of these segments, the types of motion present and a roughly quantised amount of motion are described.

We have implemented two clustering algorithms for camera motion. The first creates a fixed number of clusters, one for each type of camera motion (pan left, pan right, tilt up, tilt down, zoom in, zoom out, static). A camera motion segment is assigned to a cluster, if that type of camera motion is present in the segment, e.g. if a segment contains a pan left and a zoom in, it is assigned to both clusters. The second clustering method tries to better model the actual data. Using the amounts of each type of motion, each camera motion segment is described by a vector in a three-dimensional feature space. The feature vectors are then clustered using the Mean Shift algorithm [7]. The algorithm determines the number of clusters and assigns each camera motion segment to one of them. Depending on the data, the clusters contain single camera motions or combinations and a textual label for the cluster is created (e.g. "moderate pan and strong zoom").

4.1.2 Visual Activity

Visual activity is a measure of the dynamics in a scene. Together with camera motion information, it is a measure for the local motion in a scene and can thus be used to discriminate quiet scenes from those with object motion. In this application we just measure the amplitude of visual change. The list of amplitude values is then median filtered to be robust against short term distortions and split into homogeneous segments. Each of these sub-shot segments is described by its average activity value. Clustering is performed using the k-means algorithm.

4.1.3 Audio Volume

Audio volume can for example be used to discriminate shots without any sound, calm shots of inanimate objects, interviews with a constant volume level and loud outdoor shots in city streets. As no content-based audio segmentation is available in the system, we use segments of a fixed length of 30 seconds. A list of audio volume samples is extracted for each of these segments by calculating the average volume of a 0.5 seconds time window. The list is then median filtered to be robust against short term distortions and split into homogeneous segments. Each of these sub-segments is described by its average volume value. Clustering is performed using the k-means algorithm.

4.1.4 Face Occurrence

The occurrence of faces is a salient feature in video content, as it allows inferring the presence of humans in the scene. The size of a face is also a hint for the role of the person, i.e. a large face indicates that this person is in the center of attention. Our extractor is based on the face detection algorithm from OpenCV[4]. In order to make the description more reliable and to eliminate false positives, which mostly occur for a single or a few frames, we only accept face occurrences that are stable over a longer time (we use a time window of about a second to check this). As a result we get a continuous segmentation into sub-shot segments with and without faces. There is no need for a specific clustering algorithm, as there are only two groups of segments (face and non-face).

4.1.5 Global Colour Similarity

Global colour similarity allows to group shots that depict visually similar content, e.g. several takes of the same scene or different shots taken at the same location (if the foreground objects are not too dominant). To describe the colour properties of a shot, the MPEG-7 ColorLayout descriptor [10] is extracted from each representative frame. The ColorLayout descriptor has the advantage of also taking the spatial colour distribution of the image into account. In order to reduce the number of colour descriptors to be processed, similar descriptors extracted from representative frames of the same shot are eliminated. Then the pair-wise similarities between all remaining descriptors of the content set are calculated and stored in a matrix. The similarity matrix is used as input for hierarchical clustering using the single linkage algorithm [8]. The cutoff value for the resulting tree is determined from the desired number of clusters.

4.1.6 Object Similarity

To overcome the limitations of global colour similarity, we additionally use an interest point based approach to detect similar objects in the content. This approach does not only rely on colour information and does not require global visual similarity. As we do not perform segmentation, the term object refers in this context to a set of matching interest points that are detected in an image. Thus, the approach will also respond to partially similar objects or similar parts of the background. Our algorithm uses a combination of the SIFT [12] and MPEG-7 colour descriptors. The interest points are extracted using the difference of Gaussian pyramids originally proposed for SIFT. The MPEG-7 colour descriptors are extracted from a window around the interest points. The descriptor for an image contains the SIFT and MPEG-7 descriptors for the interest points detected in the image. The details of the algorithm are described in [18]. The extraction is performed on the representative frames. Like for the global colour descriptors, similar descriptors within a shot are eliminated and a matrix of pair-wise descriptor similarities across the whole content set is calculated. The similarity matrix is used as input for hierarchical clustering using the single linkage algorithm, using a cutoff value determined from the desired number of clusters.

[4] http://opencvlibrary.sourceforge.net

4.2 User Interface

The user interface is a part that is not covered by the framework; the summariser just defines an interface towards the presentation components. A screen shot of the browsing tool is shown in Figure 2. The central component is a light table view which shows the current content set and cluster structure using a number of representative frames for each of the clusters. The clusters are visualised by coloured areas around the images, with the cluster label written above the first two images of the cluster. The size of the images in the light table view can be changed dynamically so that the user can choose between the level of detail and the number of images visible without scrolling. The tool bar at the top of the screen contains the controls for selecting the feature for clustering and confirming the selection of a subset. The light table view allows selection of multiple clusters by clicking on one of their member images. By double clicking an image in the light table view, a small video player is opened and plays the segment of video that is represented by that image. The size of the player adjusts relatively to the size of the representative frames.

On the left of the application window the history and the result list are displayed. The history window automatically records all clustering and selection actions done by the user. By clicking on one of the entries in the history, the user can set the state of the summariser (i.e. the content set) back to this point. The user can then choose to discard the subsequent steps and use other cluster/selection operations, or to branch the browsing path and explore the content using alternative cluster features. At any time the user can drag relevant representative frames into the result list, thus adding the corresponding segment of the content to the result set, which can then be saved as edit decision list (EDL).

4.3 Discussion

The presented application for rushes browsing has been evaluated as part of our participation in TRECVID 2006. A group of seven users has used the browsing tool to fulfill a number of material requests specified by textual descriptions. Using a ground truth set for each of the material requests, precision and recall have been measured. Precision was in the range of 0.40 to 0.80 for nearly all of the tasks, while recall was between 0.22 and 0.45. In addition, feedback on the usability of the browsing tools was collected from the users. The results are reported in detail in [3].

The user tests have shown that the selection of representative media items for the clusters is an important issue. As the clusters tend to be large (especially early in the browsing process) and the reduction of redundancy is a central goal, it is not possible to show all items of a cluster. In our application we have only very limited knowledge about the context and the user's information needs (we only know that we have clustered by feature X), thus this selection is difficult. In the current application we use representative frames that best capture the feature used for clustering (i.e. a frame that is as close as possible to the most salient appearance of the feature in the segment). We found that it confuses users, if the representative frame for one segment changes with the feature used for clustering. On the other hand, keeping the previously used representative frame will make it more difficult to judge the relevance of the segment given the current feature.

5. CONCLUSION

Multimedia content abstraction approaches such as browsing applications and video summaries are of growing importance for dealing with multimedia collections. They are complementary to search and retrieval approaches and focus on problems where the formulation of a query is difficult due to the available metadata and/or the user's knowledge of the content set.

Based on reviewing the available literature we have tried to find the most relevant aspects that discriminate multimedia content abstraction approaches. Despite the variability in terms of these aspects, the approaches show many similarities, which have served as an input to defining a general five step process for multimedia content abstraction. We have used this process definition as a basis for proposing a software framework for content abstraction. We have discussed the implementation of the process in terms of the framework and we have presented its main components. For feature independent components, the framework provides a complete implementation, for the feature dependent ones, only interfaces are defined.

We have demonstrated the practical usability of the framework by implementing a browsing tool for rushes video based on it. The browsing tool is an interactive application that allows performing iterative clustering and selection in order to filter the content down to a manageable set of relevant items. Clustering can be performed using the features camera motion, visual activity, audio volume, face occurrences, global colour similarity and object similarity. A number of representative frames are used to visualise a cluster. The frames are shown in a light table view which indicates the clusters by coloured areas around the images. The tool keeps a history of the browsing process which allows to undo and redo clustering and selection steps. The application has been evaluated on the TRECVID 2006 rushes data set.

In future, we will implement plug-ins for additional features that can be used for clustering. Within the browsing application we will experiment with other visualisations of the content set. As a further application of the framework, we intend to use it as basis for other multimedia content abstraction applications, i.e. for creating video skims.

6. ACKNOWLEDGMENTS

The authors would like to thank their colleagues at the Institute of Information Systems of JOANNEUM RESEARCH for their support and feedback, especially Werner Haas and Christian Schober.

The work presented here has been partially funded under the 6th Framework Programme of the European Union within the IST project "IP-RACINE - Integrated Project Research Area Cinema" (http://www.ipracine.org, IST-2-511316-IP). This work contributes to the activities of the IST research network of excellence K-SPACE (http://www.k-space.eu, FP6-027026) funded by the 6th Framework Programme of the European Union.

7. REFERENCES

[1] W. Bailer and P. Schallauer. The detailed audiovisual profile: Enabling interoperability between MPEG-7 based systems. In H. Feng, S. Yang, and Y. Zhuang, editors, *Proceedings of 12th International Multi-Media Modeling Conference*, pages 217–224, Beijing, CN, Jan. 2006.

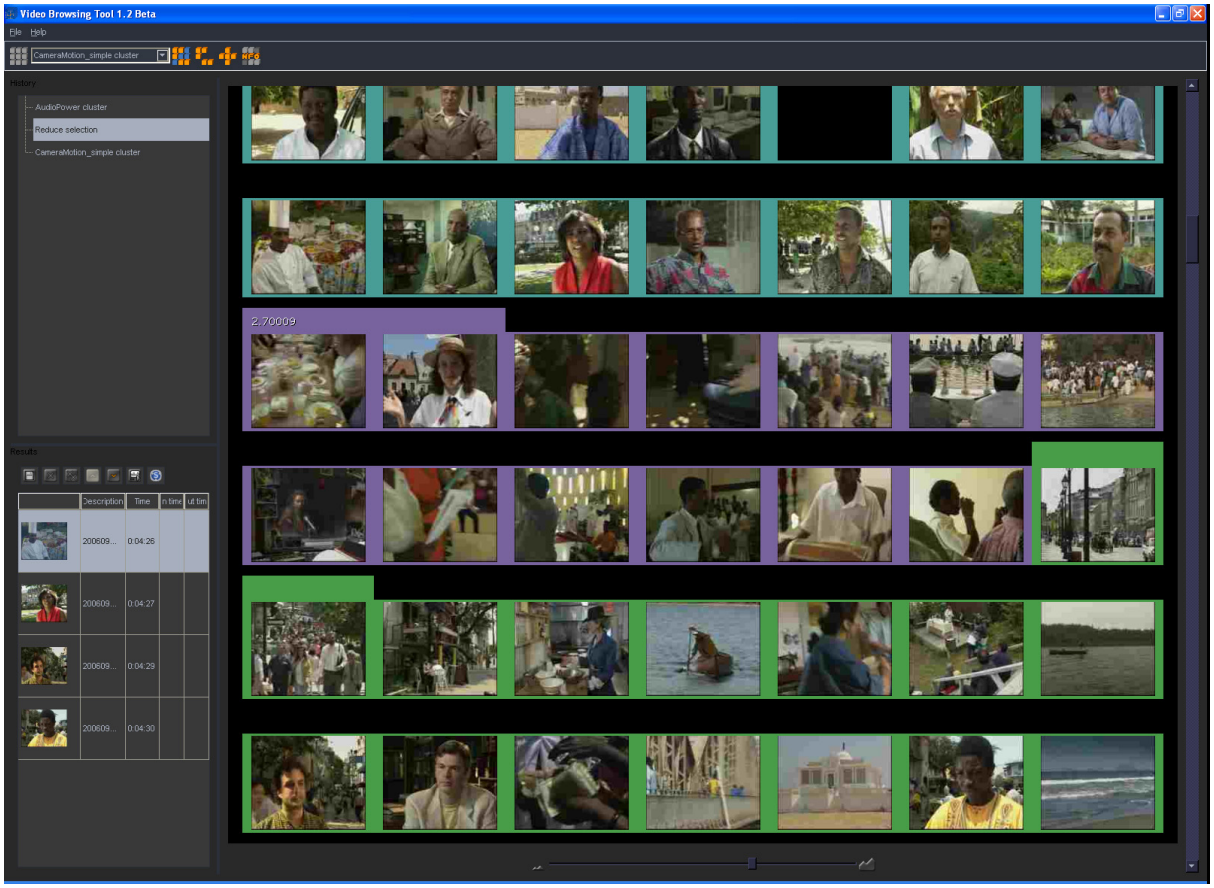

Figure 2: The user interface of the rushes browsing tool.

[2] W. Bailer, P. Schallauer, and G. Thallinger. JOANNEUM RESEARCH at TRECVID 2005 – camera motion detection. In *Proceedings of TRECVID Workshop*, pages 182–189, Gaithersburg, MD, USA, Nov. 2005.

[3] W. Bailer, C. Schober, and G. Thallinger. Video content browsing based on iterative feature clustering for rushes exploitation. In *Proceedings of TRECVID Workshop*, pages 230–239, Gaithersburg, MD, USA, Nov. 2006.

[4] I. Campbell and C. J. van Rijsbergen. The ostensive model of developing information needs. In *Proceedings of COLIS-96, 2nd International Conference on Conceptions of Library Science*, pages 251–268, Copenhagen, DK, 1996.

[5] P. Chiu, A. Girgensohn, W. Polak, E. Rieffel, and L. Wilcox. A genetic algorithm for video segmentation and summarization. In *Proc. IEEE International Conference on Multimedia and Expo*, volume III, pages 1329–1332, New York, NY, USA, 2000.

[6] M. G. Christel, A. G. Hauptmann, A. S. Warmack, and S. A. Crosby. Adjustable Filmstrips and Skims as Abstractions for a Digital Video Library. In *Proc. of the IEEE Forum on Research and Technology Advances in Digital Libraries*, pages 98–104, Baltimore, MD, USA, 1999.

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[8] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition, 2001.

[9] M. Irani, P. Anandan, and S. Hsu. Video indexing based on mosaic representations. *Proceedings of IEEE*, 86(5):905–921, 1998.

[10] Information technology-multimedia content description interface: Part 3: Visual. ISO/IEC 15938-3, 2001.

[11] R. Lienhart, S. Pfeiffer, and W. Effelsberg. Video Abstracting. *Communications of the ACM*, 40(12):54–63, Dec. 1997.

[12] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, pages 1150–1157, 1999.

[13] J. Luo, M. Boutell, and C. Brown. Pictures are not taken in a vacuum - an overview of exploiting context for semantic scene content understanding. *IEEE Signal Processing Magazine*, Mar. 2006.

[14] J. Oh and K. Hua. An efficient technique for summarizing videos using visual contents. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1167–1170, New York, NY, USA, 2000.

[15] S. Pfeiffer, R. Lienhart, S. Fischer, and W. Effelsberg.

Abstracting Digital Movies Automatically. *Journal of Visual Communication and Image Representation*, 7(4):345–353, Dec. 1996.

[16] D. Ponceleon. Hierachical brushing in a collection of video data. In *HICSS '01: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)*, volume 4, page 4045, Washington, DC, USA, 2001. IEEE Computer Society.

[17] Y. Rui and T. S. Huang. A unified framework for video browsing and retrieval. In A. C. Bovik, editor, *Image and Video Processing Handbook*, pages 705–715. Academic Press, New York, USA, 2000.

[18] P. Schügerl, R. Sorschag, W. Bailer, and G. Thallinger. Object re-detection using SIFT and MPEG-7 color descriptors. Technical report, JOANNEUM RESEARCH, 2006. Submitted to International Workshop on Multimedia Content Analysis and Mining (MCAM'07).

[19] M. A. Smith and T. Kanade. Video skimming for quick browsing based on audio and image characterization. Technical Report CMU-CS-95-186, Carnegie Mellon University, July 1995.

[20] H. Sundaram, L. Xie, and S.-F. Chang. A utility framework for the automatic generation of audio-visual skims. In *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*, pages 189–198, New York, NY, USA, 2002.

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

[22] S. Uchiashi, J. Foote, A. Girgensohn, and J. Boreczky. Video Manga: Generating Semantically Meaningful Video Summaries. In *ACM Multimedia (ACMMM'99)*, pages 383–392, Orlando, FL, USA, Nov 1999.

[23] Z. Xiong, R. Radhakrishnan, A. Divakaran, Y. Rui, and T. S. Huang. *A Unified Framework for Video Summarization, Browsing and Retrieval: with Applications to Consumer and Surveillance Video.* Academic Press, 2005.

[24] M. M. Yeung and B.-L. Leo. Video Visualization for compact representation and fast browsing of pictorial content. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(5):771–785, Oct. 1997.