Navidgator - Similarity Based Browsing for Image & Video Databases

Damian Borth, Christian Schulze, Adrian Ulges, and Thomas M. Breuel

University of Kaiserslautern,
German Research Center for Artificial Intelligence (DFKI),
67663 Kaiserslautern, Germany
{d_borth,a_ulges,tmb}@informatik.uni-kl.de
{christian.schulze}@dfki.de
http://www.iupr.org

Abstract. A main problem with the handling of multimedia databases is the navigation through and the search within the content of a database. The problem arises from the difference between the possible textual description (annotation) of the database content and its visual appearance. Overcoming the so called - semantic gap - has been in the focus of research for some time. This paper presents a new system for similarity-based browsing of multimedia databases. The system aims at decreasing the semantic gap by using a tree structure, built up on balanced hierarchical clustering. Using this approach, operators are provided with an intuitive and easy-to-use browsing tool. An important objective of this paper is not only on the description of the database organization and retrieval structure, but also how the illustrated techniques might be integrated into a single system.
Our main contribution is the direct use of a balanced tree structure for navigating through the database of keyframes, paired with an easy-to-use interface, offering a coarse to fine similarity-based view of the grouped database content.

Key words: hierarchical clustering, image databases, video databases, browsing, multimedia retrieval

1 Introduction

Nowadays, content-based image and video retrieval (CBIR/CBVR) are getting more and more into focus with the rapidly growing amount of image and video information being stored and published. Online portals providing image and video content like *flickr.com, youtube.com, revver.com, etc.* offer data in large amounts which creates the need for an efficient way of searching through the content. But the need for an efficient search and browsing method is not bound to those online portals. Archives of TV stations storing increasingly more digital content also need appropriate tools to find the material they are looking for. Additionally, with the already widely spread availability of digital acquisition

devices (still image and video cameras) it is easy for everybody to acquire large amounts of digital data in short amounts of time.

Currently the search for specific content in such collections is mostly done through a *query-by-text* approach, exploiting manual annotation of the stored data. This approach suffers from a few drawbacks which arise from the nature of this method. *First*: manual annotation is a very time consuming process which *second*: might lead to a rather subjective result, depending on the person doing the annotation. *Third*: the result of the query depends highly on the quality of the annotations. Visual content that has not been transcribed into the meta-data can therefor not be retrieved afterwards. *Fourth*: The result of a query can be manipulated by the type and number of tags associated with the visual data. This might mostly apply to online portals where currently the owners of uploaded content provide the meta-data. With such tag manipulation it is possible to assure that a specific content appears in most of the query results, which then in turn reduces the quality of the search result.

To prevent such drawbacks it is necessary to use different approaches for search in large multimedia databases. One of these approaches is the description of the database content in a specific feature space. Unfortunately, a major problem of this approach is the formulation of a query. This is bypassed by using the *query-by-example* approach [1], where a sample image or video is representing a query forming a visual concept for search.

Our approach is to build up a similarity based structure of the multimedia database (here we focus on video as content) to overcome the need for an appropriate search example. Doing this, the user is enabled to browse through the database by picking an entity as starting point which represents the query most. During browsing the user is able to zoom in and out of the database content with variable step size representing the similarity of appearance. Entities showing up during the navigation, giving a better representation of the users query, provide the opportunity to narrow down the selection of possible matches.

1.1 Related Work

With the development of multimedia retrieval/browsing systems quickly the problem of formulating a proper query arose. This lead to the definition of the so called *semantic gap* - 'a lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation.'[2]. Different approaches have been examined to bridge this semantic gap.[3–6] and lead to multiple approaches towards user interface design i.e. the RotorBrowser [7] or VideoSOM [8].

When we talk about similarity based browsing, we focus on the visual content of the video. This means we want to allow the user to find temporally independent shots of video with similar content, which is different to cluster-temporal browsing where causal relations in storytelling are used for browsing [9]. We purely utilize content based similarity between different videos in the video database for browsing the database. A similar approach can be found in [10, 11], where the concept of a similarity pyramid is introduced for browsing

large image databases. The idea of similarity pyramids was also applied to video databases in a system called ViBE [12].

Our main contribution is the direct use of a balanced tree structure for navigating through the database of keyframes, paired with an easy to use interface, offering a coarse to fine view on the grouped database content based on similarity.

2 Balanced Hierarchical Clustering

To find structures of strong visual similarity we use unsupervised learning methods like clustering. In particular we are not only interested in the pure cluster partitioning, even more we want to capture the relationship between different clustering levels i.e. the clustering structure in the video database including their cluster and their subcluster partitions. To achieve this we use a standard agglomerative hierarchical clustering [13], which runs through a series of partitions starting from singelton clusters containing a single image and terminating with a final cluster, containing all images of the database. This structure, usually represented as a dendogram, is postprocessed into a binary tree and used by our system for continuous browsing between different coarseness levels of similarity.

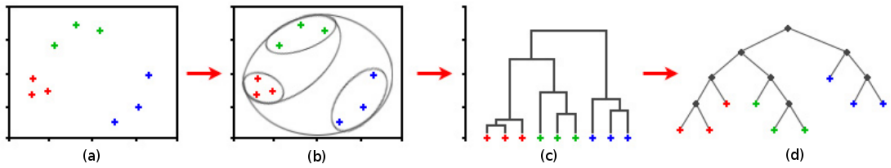

Fig. 1. A feature space representation of keyframes (a) leads to a hierarchical clustering (b). This clustering can be viewed as a dendogram with similarity measurement at y-axis (c) and will be postprocessed to a binary tree for our GUI (d)

2.1 Feature Extraction

Let the videos in the database be denoted as $X_1, ..., X_n$. Every video is represented by a set of keyframes $\{x_{i1}, ..., x_{im}\}$, resulting in a total keyframe set $\{x_1, ..., x_k\}$ for the entire video database. For every keyframe $x_i \in \{x_1, ..., x_k\}$ a feature vector $z_i \in \mathbb{R}^D$ is extracted.

A common way to extract keyframes for each video is to segment the data into shots and analyze the shots individually for representative keyframes. We use a divide-and-conquer approach that delivers multiple keyframes per shot in respect to its visual complexity. To achieve this, we compute MPEG-7 color layout descriptors [14] for each frame of the shot and fit a Gaussian mixture model to the feature set using k-means [15] in combination with the Bayesian

Information Criterion (BIC) for determining the number of clusters. [16]. The entire procedure is illustrated in more detail in [17].

The computation of the feature vector z_i for every keyframe $x_i \in \{x_1...x_k\}$ is based on our Baseline System definition [18]. There we are using Color and Texture features with a equally weighted early fusion (i.e concatenated). In particular, we use color histograms which are quantized to $8 \times 8 \times 8$ bins and also Tamura texture features [19] to form the feature vector.

2.2 Agglomerative Hierarchical Clustering

The first step in using conventional agglomerative hierarchical clustering algorithm is the computation of a distance matrix $D = [d(z_i, z_j)]$ where $i, j = 1...k$, with k the number of keyframes. The distance matrix D represents similarities between all pairs of keyframes and is used for successive fusion of clusters [13]. As distance function $d(z_i, z_j)$ serves the Euclidian distance.

The agglomerative hierarchical clustering creates a cluster $c_i = \{x_i\}$ for each keyframe $x_i \in \{x_1...x_k\}$, resulting in $C_0 = \{c_1...c_k\}$ disjoint singelton clusters, where $C_0 \subset C$. In this first step the distance matrix D is equal to the distances between the feature vectors of the keyframes. The cluster c_i c_j with the smallest distance $d(z_i, z_j)$ are fused together to form a new cluster $c_k = \{c_i, c_j\}$. After creating a new cluster, the distance matrix D has to be updated to represent the distance between the new cluster c_k to each other cluster in $C_0 \backslash \{c_k\}$. This recalculation of distances is usually done with one of the known linkage methods [11]. Considering the used linkage method the entire clustering structure will be more or less *dilating* i.e. individual elements not yet grouped are more likely to form new groups instead of being fused to existing groups. According to [11], the *complete linkage* method tends to be dilating and therefore resulting in more balanced dendograms compared to the *single linkage* method, which chains clusters together and therefore results in deep unbalanced dendograms. However, our goal is to use the resulting clustering structure for continuous similarity browsing of video databases. In order to achieve this, two points are important: `First`, the clustering must produce clusters with visual similar content and `Second`, the dendrogram produced by the clustering has to be as balanced as possible.According to our experience, *average linkage* produces the visually most similar clusters. Unfortunately, the clustering structure is not as balanced as desired for usable browsing. Therefore a modification of the average linkage method was needed to achieve the desired properties.

Viewing the resulting dendogram of the clustering as a tree structure, this structure will hierarchically organize keyframes into groups of visual similar content, thereby retaining the relationship between different coarseness levels of similarity and tree depth. Let S denote the set of all tree nodes. Each node of the tree $s \in S$ is associated with a set of keyframes $c_s \subset C \wedge c_s \notin C_0$ representing the cluster of keyframes. The number of elements in the cluster c_s is denoted by $|c_s|$. The children of a node $s \in S$ denoted by $ch(s) \subset S$ will partition the

keyframes of the parent node so that

$$c_s = \bigcup_{r \in ch(s)} c_r$$

The leaf nodes of the tree correspond to the extracted keyframes and are indexed by the set S_0. Each leaf node contains a single keyframe, so for all $s_i \in S_0$ we have $c_i = \{x_i\}$ with $|c_i| = 1$ implying that $|S_0| = |C_0|$. This notation is derived from the notations of [10–12].

Furthermore we define $D = [d(c_i, c_j)]$ as the updated distance matrix of distances between the pairwise different clusters c_i and c_j. The linkage method for calculating distances between clusters with $|c_i| > 1 \wedge |c_j| > 1$ is in our case the *average linkage* method enhanced by a weighted penalty, which depends on the amount of elements in both clusters

$$d(c_i, c_j) = \frac{1}{|c_i| * |c_j|} * \sum_{z_i \in c_i} \sum_{z_j \in c_j} d(z_i, z_j) + \alpha * (|c_i| + |c_j|)$$

We are naming this method: *balanced linkage* due to its tendency to form balanced trees with clusters of consistent visual properties. The weight can be set to $0 \leq \alpha \leq 1$, which either results in having no effect to the average linkage or totally forcing the algorithm to produce balanced trees without any visual similarity. The chosen α value was empirically evaluated and set to $\alpha = 0.01$.

2.3 Binary Tree Construction

The usually chosen representation of hierarchical clustering is a dendogram, which illustrates the fusions made at each successive stage of analysis. In a dendogram, the elements being clustered are placed usually at the bottom of the diagram and show fusions of clusters through connecting lines. Another representation for hierarchical clustering is using sets showing the elements being clustered in their feature space. Such sets represent one particular step in the clustering process and may contain subsets illustrating previous clustering steps.

Because we want to use the clustering structure for navigation, we postprocess the dendogram into a binary tree. The binary tree structure enables us to efficiently follow a keyframe from the root of the tree, which contains the entire database, to the leaf of the tree, which only contains the selected keyframe. With every *zoom in* step, the system is presenting a more similar subtree considering the selected keyframe and leaving out the frames, which are less similar to the selected keyframe. The *zoom out* step, lets the system present a tree containing more dissimilar keyframes, which might be useful for refinement of the query.

For the notation of the constructed binary tree, we refer to section 2.2. Let $T = S$ be the binary tree, then the nodes $n_i \in T$ are the fusion points where clusters c_i,c_j are being fused together to form a new cluster $c_k = \{c_i, c_j\}$.

An interesting outcome of the binary tree postprocessing is the creation of so called *content stripes*. These structures represent clusters within the binary

tree, in such a way that the keyframes of subclusters are ordered in stripes according to their similarity and therefore providing a more intuitive way of visualizing clusters at a particular level (Fig. 2). Content stripes replace the need to additionally compute spatial arrangemets for keyframes within a cluster like done with similarity pyramides [10, 11]. Therefore our method is not only able to construct a hierarchical database structure but also to build up a similarity based order within a cluster in one single step instead of separating these tasks.

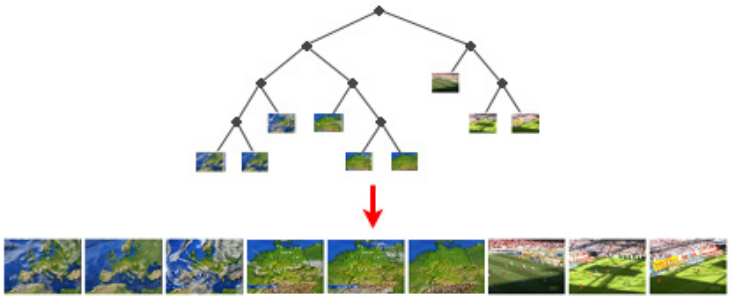

Fig. 2. A binary tree representation of a sample clustering, transformed to a content stripe displaying similarity clusters in an one-dimensional order

3 Graphical User Interface

Retrieval systems based on keyframes and best-match similarity tend to present a localized view of the database to the user, rather then providing an overview of the entire database. For uses who do not clearly know what exactly they are searching for, it would be more efficient to let them browse through the database and allow them to dynamically redefine their search query.

In this section we would like to present the Navidgator graphical user interface [1], which allows a user to easily and efficiently browse a video database in respect to his selected query. In our system a browsing process is initialized with starting at the root of the hierarchical clustered binary tree. First, the user has to select his first keyframe out of a randomly sampled set from the entire database to formulate his query [1]. This keyframe will represent the visual concept, which will guide the user while browsing. The user is also able to dynamically refine his visual concept in every point during browsing by selecting another keyframe.

Browsing itself is performed by the given zooming tools. The user can either *zoom-in* or *zoom-out* in the database. A *zoom-in* action will narrow down the available keyframe according to his visual concept and a *zoom-out* action will display a coarser level of the database to the user. For better usability the

[1] http://demo.iupr.org:8180/navidgator-tv (10.000 keyframes database)

interface provides a *multi-level-zoom-in* action and a *multi-level-zoom-out* action. Furthermore the user can perform a *max-zoom-in* action, which brings him straight to the most similar keyframes in the database or a *max-zoom-out* action, which brings him back to the root of the binary tree i.e. the top of the database. The depth of the database and the users current position are visualized by a vertical bar next to the zooming tools enabling an intuitive orientation. Additionally the user can utilize a click history to jump back to particular points of his browsing process. The Navidgator browsing interface is displayed in Fig. 3.

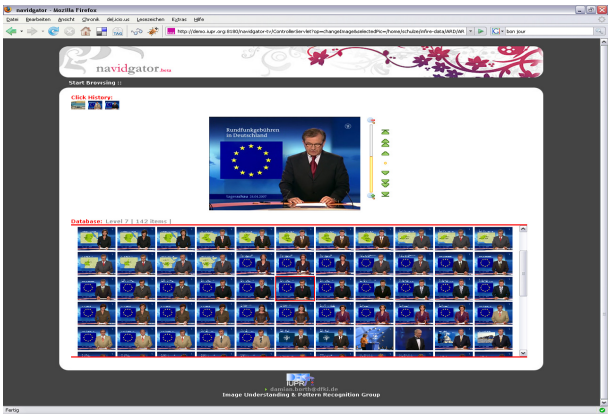

Fig. 3. The Navidgator browsing interface. The selected visual concept is displayed at the center. The lower area displays the cluster preview box, where the visual concept might be refined. Next to the selected keyframe the navigation tools are arranged

4 Conclusion

In this paper we have presented a new system for similarity-based browsing of multimedia databases. By using a balanced tree based on hierarchical clustering of the database content it is possible to supply users with an intuitive and easy-to-use browsing tool. We were able to improve search results by providing a set of navigation tools which support the decision tree like structure of the clustering. Because of our concept of an offline clustering and online retrieval we are able to efficiently perform a search on the entire database. The system offers coarse and detailed views on the database content with the opportunity to change the focus of search at any time. This enables the user to start navigation with a fuzzy visual concept and improve relevance incrementally while browsing.

Our future work will focus on advanced tree building and on dealing with growing databases, which will basically cover merging of new data into the binary tree. Additionally, to bridge the semantic gap and include high-level semantics we want to enhance the browser with an automatic tagging system like in [18]

References

1. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., Yanker, P.: Query by image and video content: the qbic system. Computer **28,Issue 9** (Sept. 1995) 23 – 32
2. Smeulders, A., Worring, M., Santini, S., Gupta, A., Jain, R.: Contentbased image retrieval at the end ofthe early years. IEEE transactions on pattern analysis and machine intelligence **22**(12) (2000) 1349–1379
3. Broecker, L., Bogen, M., Cremers, A.B.: Bridging the semantic gap in content-based image retrieval systems. In: Internet Multimedia Management Systems II. Volume 4519 of Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference. (2001) 54–62
4. Zhao, R., Grosky, W.: Narrowing the semantic gap-improved text-based web document retrieval using visual features. Multimedia, IEEE Transactions on **4**(2) (2002) 189–200
5. Zhao, R., Grosky, W.: Bridging the Semantic Gap in Image Retrieval. Distributed Multimedia Databases: Techniques and Applications (2003)
6. Dorai, C., Venkatesh, S.: Bridging the semantic gap with computational media aesthetics. Multimedia, IEEE **10**(2) (2003) 15–17
7. de Rooij, O., Snoek, C.G.M., Worring, M.: Mediamill: semantic video search using the rotorbrowser. [7] 649
8. Barecke, T., Kijak, E., Nurnberger, A., Detyniecki, M.: Videosom: A som-based interface for video browsing. IMAGE AND VIDEO RETRIEVAL, PROCEEDINGS **4071** (2006) 506–509
9. Rautiainen, M., Ojala, T., Seppanen, T.: Cluster-temporal browsing of large news video databases. IEEE Int. Conference on Multimedia and Expo **2** (2004) 751–754
10. Chen, J., Bouman, C., Dalton, J.: Similarity pyramids for browsing and organization of large image databases. SPIE Human Vision and Electronic Imaging III **3299** (1998)
11. Chen, J.Y., Bouman, C., Dalton, J.: Hierarchical browsing and search of large image databases. Image Processing, IEEE Transactions on **9, Issue 3** (March 2000) 442 – 455
12. Taskiran, C., Chen, J., Albiol, A., Torres, L., Bouman, C., Delp, E.: Vibe: A compressed video database structured for active browsing and search. IEEE Transactions on Multimedia **6**(1) (FEB 2004) 103–118
13. Johnson, S.C.: Hierarchical clustering schemes. PSYCHOMETRIKA **32**(3) (1967) 241–241
14. Manjunath, B., Ohm, J., Vasudevan, V., Yamada, A.: Color and texture descriptors. IEEE Trans. on Circuits Syst. for Video Techn. **11**(6) (2001)
15. McQueen, J.: Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability (1967) 281–297
16. Schwarz, G.: Estimating the dimension of a model. Annals of Statistics **6**(2) (1978) 461–464
17. Borth, D., Ulges, A., Schulze, C., Breuel, T.M.: Keyframe extraction for video taggging and summarization. In: Informatiktage 2008. (2008) 45–48
18. Ulges, A., Schulze, C., Keysers, D., Breuel, T.M.: Content-based video tagging for online video portals. In: MUSCLE/Image-CLEF Workshop. (2007)
19. Tamura, H., Mori, S., Yamawaki, T.: Textual features corresponding to visual perception. IEEE Transactions on Systems, Man, and Cybernetics **SMC-8**(6) (1978)