

ProVeR: Probabilistic Video Retrieval using the Gauss-Tree

Christian Böhm Michael Gruber Peter Kunath Alexey Pryakhin Matthias Schubert
Institute for Informatics
University of Munich
D-80538 Munich, Germany
{boehm,gruber,kunath,pryakhin,schubert}@dbs.ifl.lmu.de

Abstract

Modeling objects by probability density functions (pdf) is a new powerful method to represent complex objects in databases. By representing an object as a pdf, e.g. a Gaussian, it is possible to represent very large and complex objects in a compact and still descriptive way. In this contribution, we propose ProVeR a prototype search engine for content-based video retrieval which represents a video as a set of Gaussians. The Gaussians are managed by the Gauss-tree, an index structure allowing the efficient processing of probabilistic queries. ProVeR provides even non-expert users with an intuitive method for efficient, content-based retrieval of videos containing similar shots and scenes.

1 Introduction

Video clips are an important type of multimedia data. Due to recent technical advances, the amount of video data that is available in digital formats has increased enormously. In this contribution, we propose ProVeR, a system for probabilistic video retrieval that is based on uncertain object representations. ProVeR focuses on the following scenario: Given a database of movies or video clips, we want to retrieve all movies from the database that are likely to contain a given query scene. For this type of scenario, there are various applications. For example, a company wants to determine whether a given video podcast or shared video file contains scenes from any copyright protected movie or video clip. In this scenario, the company would store all of its movies in the database and automatically check whether the scenes in the video podcast match any scenes in the database. Another example is a database of news programs recorded on various days from various tv stations. A user can retrieve all news programs that are likely to contain a given video clip featuring a particular event. Since most news programs use videos which are provided by video

news agencies, it is very likely that the news programs dealing with similar topics contain similar news clips. From a technical point of view, video data consists of a sequence of images (so-called frames). To allow similarity search on video clips, each frame is represented by a feature vector. ProVeR allows the use of color histograms, color moments and texture features. The sequence of frames is segmented into so-called shots. Each shot corresponds to a set of subsequent frames that were taken from the same perspective. Thus, the change between subsequent frames in a shot is rather small while in most cases the change between images from different shots is rather large. Each shot is now summarized by a Gaussian distribution function over the employed feature space. The Gaussians representing the shots of all stored video clips are now indexed employing the Gauss-tree [1]. The Gauss-tree is a dedicated index structure for efficiently managing Gaussian distributions. The Gauss-tree is used to speed up queries w.r.t. identification uncertainty. This uncertainty model is used to answer the question whether a given exact or uncertain observation matches any object that is stored within the database. In ProVeR, we want to retrieve the shots in the database that most likely match a given query shot. Since for many practical applications a query will be composed by more than one shot, ProVeR can answer multiple shot queries, by segmenting a query frame sequence into several shots and process a query for each query shot separately. For the complete query, ProVeR afterwards averages the result probability over these single shot queries.

2 System Architecture and Implementation

Figure 1 illustrates the client/server architecture of our prototype. The server manages a video repository that contains video data that can be queried by the clients. Whenever a video is added to the repository by the management module, the video decoder module computes a summarization in form of a set of Gaussians. This step is performed for several different features, like e.g. color histograms. To

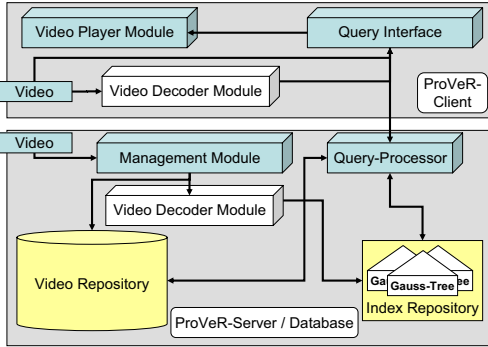


Figure 1. Architecture.

support efficient query processing, each summarization for each representation is stored in a separate Gauss-tree. During query time, the user can choose between these feature representations. The server also manages a list of clients that are connected to its query processor module. A client processes query videos given by a user. For each query video, a summarization is generated by the client-side video decoder module. The calculated summarization is sent to the server which returns references to the k most likely videos in the repository. The videos in the result can be viewed by the video player module of the client. An important part for the server as well as the client is the video decoder module. It generates a set of Gaussians for a given input video. A video is decoded into a sequence of single images. While decoding, feature vectors for several different image representations are calculated. In the next step, a shot detection is performed for each representation. The sequence of feature vectors which corresponds to a single shot is then aggregated by a Gaussian. Thus, the input video is described by a set of Gaussians.

ProVeR is implemented in Java 5.0 and its feature extraction is based on the Java Media Framework (JMF) 2.1. The videos are associated to several different representations. We extract color histograms, color moments and texture descriptions on a per-frame basis. The color histograms use the HSV color space which is divided into 32 subspaces. Additionally, we compute the color moments for the HSV color space. To capture the structural nature of the images, we also calculate the Haralick texture features [2].

3 Practical Benefits

The ProVeR client starts with a list of known multimedia servers. Initially, the user chooses one of the available servers from the list. The client establishes a connection to the multimedia database on the server. In order to perform a query, the user has to supply a video file. While decoding the video, ProVeR extracts the image representations mentioned in Section 2. Depending on the selected representa-

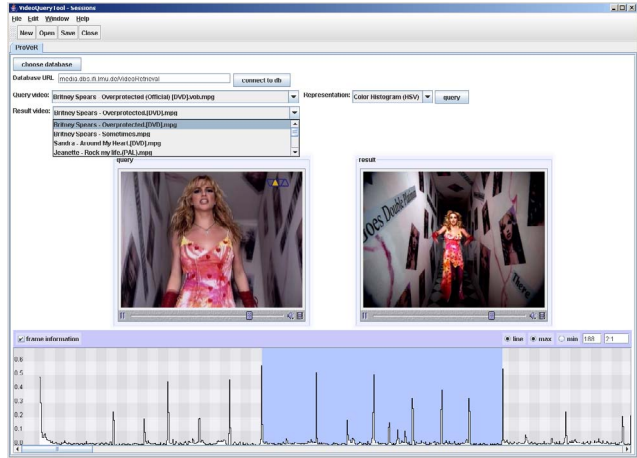


Figure 2. Screenshot of ProVeR.

tion, the client displays a distance graph in the bottom of the window (cf. Figure 2), providing information about the shot structure of the video. A valley in the distance graph indicates a sequence of similar images, which usually form a shot. To specify a query, the user selects either a single shot or a sequence of subsequent shots. To display the content of the selected query frames, ProVeR offers a preview consisting of a frame for each shot. For each selected shot, the client transforms the corresponding sequence of feature vectors into a single Gaussian. Thus, the sequence of selected shots is summarized by a set of Gaussians. This set is sent to the server as a query. Since the client sends only aggregated information to the server, the user doesn't have to share the original video data. Besides, this helps to save a lot of transmission bandwidth. Additionally, the decentralized approach also saves CPU time on the server. The server processes the query and returns a list of video repository references which contains the k most likely videos corresponding to the query. The user can browse through this result list and play a video file by selecting it, in which case it is streamed to the client. To conclude, ProVeR is a search engine for content-based video retrieval that offers an intuitive access to the shots and scenes contained in large video repositories. To allow efficient and effective retrieval, ProVeR represents shots as Gaussians which are stored in several Gauss-trees, one for each representation.

References

- [1] C. Böhm, A. Pryakhin, and M. Schubert. "The Gauss-Tree: Efficient Object Identification of Probabilistic Feature Vectors". In *Proc. 22nd Int. Conf. on Data Engineering (ICDE'06)*, Atlanta, GA, US, 2006.
- [2] R. M. Haralick, S. K., and D. I. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 3(6):610–621, 1973.