Searching in Video Collections using Sketches and Sample Images – The Cineast System

Luca Rossetto, Ivan Giangreco, Silvan Heller,
Claudiu Tănase, and Heiko Schuldt

Databases and Information Systems Research Group
Department of Mathematics and Computer Science
University of Basel, Switzerland
{luca.rossetto, ivan.giangreco, silvan.heller,
c.tanase, heiko.schuldt}@unibas.ch

Abstract. With the increasing omnipresence of video recording devices and the resulting abundance of digital video, finding a particular video sequence in ever-growing collections is more and more becoming a major challenge. Existing approaches to retrieve videos based on their content usually require prior knowledge about the origin and context of a particular video to work properly. Therefore, most state of the art video platforms still rely on text-based retrieval techniques to find desired sequences. In this paper, we present Cineast, a content-based video retrieval engine which retrieves video sequences based on their visual content. It supports Query-by-Example as well as Query-by-Sketch by using a multitude of low-level visual features in parallel. Cineast uses a collection of 200 videos from various genres with a combined length of nearly 20 hrs.

1 Introduction

From all commonly used types of media, video offers the largest variety to express information. In the previous years, the amount of videos and thus the sheer size of video data accessible online has seen a significant growth. The same is true for the number of both video related usages and users. This development is expected to continue with even increased speed in the coming years. A result of this development is that finding a specific video (or a scene within a video) in this data deluge usually requires the collection to be properly annotated. However, it is not realistic to assume that every video creator is willing to provide or even capable of adding the required annotations during the upload process. Automating this process would require a highly sophisticated software which is able to understand the semantics and the context of any arbitrary video sequence, and it would have to anticipate any subsequent use of this video. While this is –especially constrained to certain situations and depending on the context– a non-trivial task for a human, it is highly difficult or even impossible for a machine. One possibility to overcome this problem is to employ content-based retrieval techniques. In what follows, the term "content" is used to refer to the

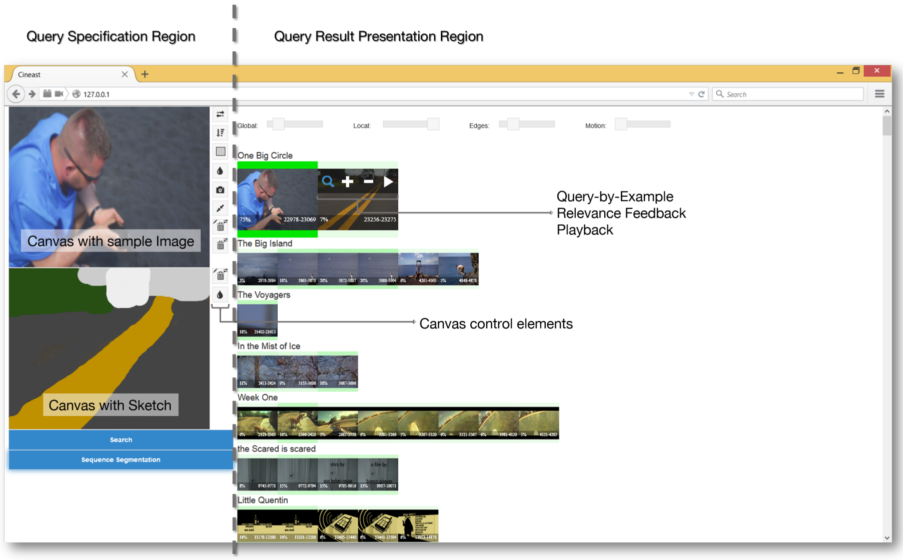

Fig. 1: Screenshot of the Cineast user interface with sequence segmentation

raw information such as pixels within frames, samples within audio tracks, and letters within subtitles as they are encoded, and not to the semantic meaning instilled within a video as interpreted by a human being.

Content-based search in video collections requires sophisticated retrieval support. This includes both the back-end of a video retrieval system and the user interaction at the front-end. In terms of the retrieval back-end, the content of the videos has to be effectively reflected by means of characteristic features and efficient search mechanisms have to be provided that are able to identify the most similar videos or video shots for a given query. In terms of the user interface, different query paradigms need to be supported that help users expressing their information need, i.e., to specify (parts) of the contents of the videos or shots a user is looking for.

In this paper, we present Cineast [2], a content-based video retrieval engine which retrieves video sequences based on their visual content. Cineast is built as a general-purpose video retrieval engine, with main usage in known-item search. It supports multiple query paradigms such as Query-by-Example (QbE) and Query-by-Sketch (QbS), as well as relevance feedback.

This paper briefly introduces the architecture of the Cineast system (Section 2) and discusses in Section 3 how the system can be used in a Query-by-Sketch and/or in a Query-by-example mode.

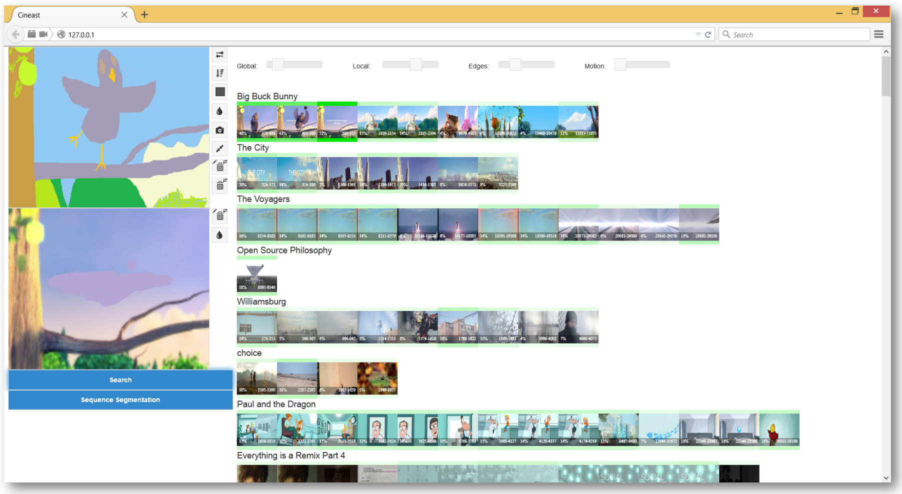

Fig. 2: Screenshot of the Cineast user interface with results grouped by video

2 Cineast

2.1 Architecture

From a conceptual point of view, the Cineast system can be divided into an on-line and an off-line part [2]. The on-line part deals with retrieval while the off-line part handles the feature extraction. Both parts are architecturally very similar. The primary component for both parts is the *feature module runtime* which manages the individual feature modules. These modules, in turn, are responsible for extraction and retrieval based on their individual feature vectors.

During feature extraction which is performed off-line, the input video is decoded and segmented into shots[1]. For every shot, a set of shot descriptors is generated. These shot descriptors only contain a part of the information present within the shot and serve as input for the feature modules. Shot descriptors include, among others, pixel-wise average or median over all frames, a most representative keyframe (which is the one frame with the smallest distance to the average image), the set of all frames of the shot, the set of all subtitle elements present during the shot, etc. The feature modules then use one or multiple of these descriptors to produce one or multiple feature vectors which are then stored in ADAM, a database system capable of managing feature data [1].

During retrieval, a set of shot descriptors is produced from the query provided by the user. The individual modules use these descriptors in a similar way as in the off-line extraction case to produce query vectors which are used to generate a list of results for every module. These result lists are combined into a final

[1] A shot is a continuously recorded sequence of frames. All frames within one shot are visually similar to each other.

ordered list of results by the feature runtime using a weighted average over the similarity scores.

2.2 Features

Currently, Cineast implements over three dozen features. These include global features, such as the average/median color, dominant shot colors, chroma and saturation, and a color histogram. Furthermore, it provides regional features, e.g., for color (color moments, color layout descriptor, color element grids), and for edges (partitioned edge image, edge histogram descriptor, dominant edge grid). In addition to the visual low-level features, Cineast also comes with motion features, i.e., directional motion histograms and regional motions sums, and text features applied to the subtitles. The features are grouped into categories that subsume multiple features.

2.3 User Interaction

Figure 1 shows the browser-based user interface of Cineast with the retrieved results (right hand side) for the given sketch. The left-hand side of the UI is used for query specification.

Query Specification: The left hand side of the UI contains two input canvases which can either be used for sketching or for providing query images acquired, for instance, from a webcam or the file system. This gives the user a wide range of input possibilities. Multiple canvases enable the user to specify multiple subsequent shots of a particular video sequence which opens up the possibility to search for longer, more visually diverse sequences while simultaneously increasing the chances of the correct video being high up in the result list. Both canvases additionally offer the possibility for a flow-field input to specify motion within a shot. Cineast also supports hybrid queries that consist of a sketch in one canvass and a sample image in the other. Moreover, sketches and sample images can even be combined in a single canvass by superimposing a sketch and a sample image (e.g., for adding further objects by means of a sketch on top of a query image).

Query Result Presentation: The right-hand side of the UI is used to display the retrieved video sequences. The smallest unit of information which is retrieved by Cineast is a shot, but since a query from multiple input canvases may return multiple shots, the results are grouped by video of origin as shown in Figure 2. The shots are ordered chronologically within each video while the displayed videos are ordered by the average score of their shots. The similarity score for each shot is displayed in the bottom left-hand corner as well as visually indicated by the color of its border.

Because certain features used for retrieval take more time for evaluation than others, partial results become available quite some time before the complete retrieval process is complete. Partial results are streamed to the front-end as soon as they become available. The front-end will then start displaying results as soon as they become available and re-order them as necessary which highly increases the usability of the interface. A loading icon in the lower left corner of the screen indicates that a query is still being processed by the back-end and more results might still be coming.

A click on the Sequence Segmentation button re-arranges the results. Shots are no longer grouped simply by video but a video is broken apart into multiple sequences as depicted in Figure 1. This is useful in cases where the query matches a certain video multiple times.

For each shot in the results, the user has the possibility to play the video at the corresponding position.

Query Refinement: In addition to displaying the retrieved video shots, a user can also take these shots as input for a next, refined query. This can be done in one of three ways. A shot can be used as a direct query input by clicking on the looking-glass symbol which appears when the mouse cursor enters the thumbnail. This will cause Cineast to search for shots which are similar to the one selected. It is also possible to drag a thumbnail from the results into one of the canvases and modify it by sketching to make it more closely resemble the shot one is looking for. Lastly, multiple shots can be selected from the result set to serve as positive and negative examples for relevance feedback, which is then triggered using a dedicated button.

System Configuration / Parameter Setting: The four sliders at the top of the result area control how the shot similarity is computed. Cineast uses many features with as many different measures of similarity. For reasons of usability, these features are grouped into four categories; global and local color similarity, edge similarity, and motion similarity. Depending on the properties of the query, a user might want to adjust these sliders to better reflect the intent of the search. Manipulating the sliders will re-rank the results in real time without involvement of the retrieval back-end.

2.4 Implementation

Cineast is written in Java and provides a network-accessible JSON-API. The user interface is browser-based. The server-side which is implemented in PHP serves as a proxy between the browser and Cineast as well as a server for static content such preview images. The client interface uses a JSON-streaming library[2] for the asynchronous processing of retrieved results.

[2] http://oboejs.com/

3 Cineast in Action

In order to show the Cineast system in action, two independent and unconnected machines should be used. The first one hosts the OSVC1 dataset [3], a collection of 200 creative commons videos. A user is able to explore the roughly 20 hours of video by browsing to select a video sequence to be searched for.

The second machine, which has to offer a touch- and pen-based input, runs Cineast. With this machine, users are able to search for the video sequence they have previously selected as their target.

Users of Cineast may sketch one or multiple scenes using color- and/or line-sketches and draw motion to describe their selected scene. Each canvas thereby describes a different shot. In case the scene in question was not among the results, a user has the possibility to refine a query using parts of the results. This can either be done using a retrieved shot directly as an input, thereby switching from a *Query-by-Sketch* to a *Query-by-example* modality, or by using it as a base for a new query image by dragging the preview into one of the canvases and modifying it using the sketching tools. Additionally, multiple results can be tagged as positive or negative examples and used for relevance feedback.

This set-up allows to demonstrate the highly efficient and effective query support for large video collections provided by the Cineast system. A video of the system in action can be found at: http://youtu.be/NjDqToONeZc.

4 Conclusion

In this paper, we have presented the Cineast video retrieval engine and we have shown how it can be interactively used to retrieve sequences of video based on sketches, sample images, and motion information.

Acknowlegements

This work was partly supported by the Swiss National Science Foundation in the context of the CHIST-ERA project IMOTION (20CH21_151571).

References

1. Ivan Giangreco, Ihab Al Kabary, and Heiko Schuldt. ADAM — A Database and Information Retrieval System for Big Multimedia Collections. In *Proceedings of the IEEE International Congress on Big Data 2014 (BigData 2014)*, Anchorage, USA, 2014. IEEE.
2. Luca Rossetto, Ivan Giangreco, and Heiko Schuldt. Cineast: A multi-feature sketch-based video retrieval engine. In *Multimedia (ISM), 2014 IEEE International Symposium on*, pages 18–23. IEEE, 2014.
3. Luca Rossetto, Ivan Giangreco, and Heiko Schuldt. OSVC – Open Short Video Collection 1.0. Technical Report CS-2015-002, University of Basel, 2015.