vitrivr – A Flexible Retrieval Stack Supporting Multiple Query Modes for Searching in Multimedia Collections

Luca Rossetto Ivan Giangreco Claudiu Tănase Heiko Schuldt
Department of Mathematics and Computer Science
University of Basel, Switzerland
{luca.rossetto | ivan.giangreco | c.tanase | heiko.schuldt}@unibas.ch

ABSTRACT

vitrivr is an open source full-stack content-based multimedia retrieval system with focus on video. Unlike the majority of the existing multimedia search solutions, vitrivr is not limited to searching in metadata, but also provides content-based search and thus offers a large variety of different query modes which can be seamlessly combined: Query by sketch, which allows the user to draw a sketch of a query image and/or sketch motion paths, Query by example, keyword search, and relevance feedback. The vitrivr architecture is self-contained and addresses all aspects of multimedia search, from offline feature extraction, database management to frontend user interaction. The system is composed of three modules: a web-based frontend which allows the user to input the query (e.g., add a sketch) and browse the retrieved results (vitrivr-ui), a database system designed for interactive search in large-scale multimedia collections (ADAM), and a retrieval engine that handles feature extraction and feature-based retrieval (Cineast). The vitrivr source is available on GitHub under the MIT open source (and similar) licenses and is currently undergoing several upgrades as part of the Google Summer of Code 2016.

Keywords

Content-based multimedia retrieval; multimedia search

1. INTRODUCTION

The continuous increase of multimedia content on the Internet requires new ways of storing, organizing, and searching within collections of photos or videos. Especially for searching in large multimedia collections, a one-size-fits-all approach which relies on just one query paradigm is no longer sufficient as user intentions in different applications tend to be very diverse. However, most currently available multimedia search solutions mainly focus on tag-based keyword searches.

Multimedia information retrieval (MMIR) is an ever growing field in research concerned with the extraction of semantic information from large collections of multimedia content. MMIR has enormous potential for application in search engines, yet current commercial search engines are essentially metadata retrievers that do not consider the visual content and are vulnerable to tag abuse or copyright infringement. We consider visual information and sketching valuable ways of describing video content that are currently only slightly scientifically understood and completely absent from the software landscape.

Compared to content-based image retrieval (CBIR) which has recently gained widespread popular awareness thanks to initiatives like Google Brain[1] or the ImageNet challenge[2], content-based video retrieval (CBVR) has so far attracted only relatively little scientific support. The state-of-the-art in video retrieval research has been, with a few exceptions, reduced to CBIR through the use of keyframing. This fact is also reflected in the open source community: according to Wikipedia[3] there are at least 10 open source CBIR engines, but there is no open source CBVR system we are aware of. Motion is an important mode in video content which is considerably more challenging to exploit than images. In particular, sketching for motion has remained a research-only question limited to small video datasets [5].

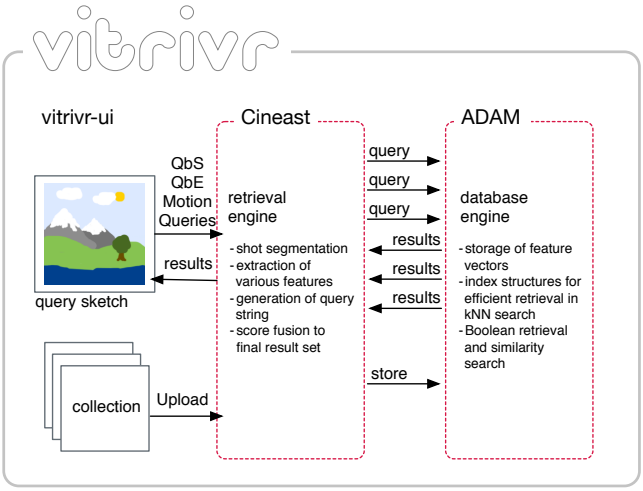

Figure 1: Architecture of vitrivr

MM '16, October 15 - 19, 2016, Amsterdam, Netherlands

ISBN 978-1-4503-3603-1/16/10...$15.00

DOI: http://dx.doi.org/10.1145/2964284.2973797

[1] https://research.google.com/teams/brain/
[2] http://image-net.org/challenges/LSVRC/
[3] https://en.wikipedia.org/wiki/List_of_CBIR_engines

The vitrivr stack is an open source search system that supports scalable Query by sketch, in the form of color or motion sketches, over video collections spanning hundreds of hours of content. vitrivr is available for download from https://vitrivr.org.

The remainder of the paper is structured as follows: in Section 2, we briefly summarize the functionality of vitrivr. The complete vitrivr architecture is presented in Section 3, the query modes are described in Section 4, and Section 5 lists the features used. Section 6 briefly surveys related work. Information on how to download and deploy the vitrivr stack are given in Section 7. Section 8 concludes.

2. MAIN FUNCTIONALITY

The vitrivr stack is a content-based multimedia retrieval engine with focus on video. Its modular architecture makes it however easy to process different types of media as well. The main capabilities of the vitrivr stack not found in this combination in other retrieval systems are:

- Content-based retrieval in multimedia documents
- Multiple complementary query modes
- Distributable across multiple machines
- Scalable to large multimedia collections

3. THE VITRIVR STACK

The vitrivr stack is comprised of three main components: the feature database ADAM [3], the retrieval engine Cineast [8], and the browser-based vitrivr frontend. Figure 1 summarizes the architecture of vitrivr discussed in the following in more detail.

3.1 ADAM

ADAM is a database system that is able to store and retrieve multimedia objects by seamlessly combining aspects from databases and information retrieval. ADAM applies both the relational database model and the vector space model. The latter considers multimedia objects as vectors in a high-dimensional feature space and defines the similarity between two objects by the distance in the spanned space. For structured data, ADAM queries can use Boolean filter predicates and make use of traditional database B-tree index structures; for ranking the elements of a collection according to a similarity score, a similarity retrieval can be performed using the Vector Approximation-File (VA-File) [11].

ADAM is implemented in PostgreSQL 9.3. We have extended PostgreSQL to support the storage of feature vector data using a new data type. Furthermore, we have added the VA-File as a new index structure to PostgreSQL and we have adapted the query execution to support Boolean retrieval and nearest neighbor retrieval at the same time and in combination.

ADAM is able to scale to multimedia collections of multiple million objects with still performing well below a few seconds: In our evaluation with 14 million feature vectors each having 144 dimensions, ADAM returns results on average in 0.55 seconds for the 100 most similar objects [3].

3.2 Cineast

Cineast is a content-based retrieval engine which forms the main query processing component of the vitrivr stack. Its primary focus is on video retrieval, but its modular architecture allows for easy extensions towards other types of multimedia such as images or audio. Cineast is implemented in Java and supports multiple query modes [7] such as *Query by Example* (QbS), *Query by Sketch* (QbS) which includes not only color but also motion sketches and *Relevance Feedback* (RF). Section 4 offers more details on these query modes. Cineast provides a simple networked JSON API which can be used to query the system.

Cineast can logically be divided into two parts, one being concerned with feature extraction from multimedia documents and the population of the metadata storage system (*offline phase*) while the other takes care of interpreting user queries and retrieving relevant documents (*online phase*). Both phases use a very similar software architecture employing a multitude of independent feature modules in parallel. A feature module is a self contained unit responsible for extracting feature vectors from multimedia and query documents and comparing such vectors. An overview of available modules is given in Section 5.

Given a query and a connection to the ADAM database system, each module is capable of producing a list of scored documents which are similar to the query by the metric employed by the module. These lists of scored documents resulting from the feature modules are combined by the retrieval runtime using a late-fusion approach. Additional details on this method are provided in [8].

3.3 Frontend

The vitrivr frontend is browser-based and thus offers a high degree of flexibility and customization. A web server is used to provide the static content such as multimedia files and the components of the UI itself while also serving as a proxy (implemented in PHP) between the UI and Cineast.

Queries are specified using one or multiple canvases which can be used to either sketch a query or use an existing input image which can be added via drag and drop. Imported images can also be modified using the sketching tools to closer approximate or further refine a query.

The frontend uses Oboe.js[4] to stream query results. This enables the system to start displaying partial results while the retrieval backend is still processing the query. Whenever new results are ready, the result display is updated and the results are re-ranked dynamically. The user has the possibility to influence the ranking by adjusting the weights for different measures of similarity. Changing these weights, even after a query has been processed, changes the order of the displayed results accordingly.

Putting the frontend in a browser enables access from a wider range of devices and also facilitates scaling, storage and retrieval across multiple servers or deployment in the cloud.

4. USAGE & QUERY MODES

4.1 Offline mode

The feature extraction process enables vitrivr to make a document searchable. This process is time consuming and hence performed offline. To perform the extraction task, the Cineast command line interface can be used. Using the `extract` command followed by a folder path, Cineast will

[4]http://oboejs.com

Figure 2: Screenshot of the vitrivrUI.

extract features from the video file within the folder as well as any available subtitle files. For larger collections, it is also possible to add custom extraction logic which directly invokes the internal routines. More flexible extraction via the API will be provided in the future.

4.2 Online mode

Once deployed on a multimedia collection, the frontend (see Figure 2) serves as querying and result display interface. The user can specify a query by sketching with colors or drawing motion paths (QbS) and she can re-use a result as a query (QbE) or import a frame into the drawing area. Furthermore, using the buttons, she can provide feedback on the retrieved results and refine the query. The interaction modes are detailed below.

Query by Sketch is the main interaction mode with the vitrivr system. In this mode the query is represented by a drawing, created by using the sketching tools present in the UI and/or dragging one of the results into the sketching area. The system retrieves shots that visually match the closest with the query. As the query is passed to Cineast, feature extractors process the query image and obtain query feature vectors. Each vector queries for k nearest neighbors in ADAM, which thanks to its indexes speeds up this operation. Returned results and their distances are aggregated using a 2-step hierarchical score fusion leading to the final list of results presented to the user.

Query by Example retrieves shots from the collection similar to a specific shot, which the user can select from the results list. QbE also makes use of the indexing mechanisms of ADAM for retrieving near neighbors and is faster than QbS since it bypasses feature extraction.

Relevance Feedback enables more complex result filtering by allowing several shots in the results list to be marked as *relevant* or *non-relevant*. The system retrieves results that are simultaneously similar to the relevant shots and dissimilar with the non-relevant ones.

5. FEATURES

This section provides a brief overview of the feature modules currently[5] available in the Cineast retrieval engine.

Global Features

- Average / Median color
- Dominant shot colors
- Color Histogram

Regional color features

- Color moments: channel-wise statistical moments over regional partitions (uniform grid, angular radial partitioning) of an aggregation over all frames of a shot
- Registered color grid: grid of quantized colors registered during retrieval
- Color Layout Descriptor
- Color element grids: grids containing partial color information in various representation (average saturation, variance of hue, etc.)
- Subdivided color histogram: color histograms of image partitions

Regional edge features

- Partitioned edge image: regional ratios of edge- and non-edge pixels
- Edge Histogram Descriptor
- Dominant edge grid: regional dominant edge direction quantised into 5 categories.

Motion features

- Directional motion histograms: regional normalised histograms of motion quantized into 8 directions.
- Regional motion sums: regional sums of the lengths of all motion vectors.

[5]as of Q2 2016

6. RELATED WORK

MMIR has been an active research topic for decades. The influential paper by Flickner et. al. on the QBIC system [2] demonstrated the feasibility of image retrieval with shape, texture and sketch. However, it was only years later (around 2002) when the first open source fully integrated CBIR systems like imgSeek[6] came along.

For CVBR, ongoing benchmarks such as TRECVID [6] focus on very large scale video datasets an mostly deal with non-interactive retrieval. A common approach involves keyframing video shots, which reduces the problem to CBIR, thus throwing away all motion information. On the side of interactive video search, some research prototypes offer query by sketch capabilities, but this approach is far from mainstream [10]. Aside from vitrivr, none of these research prototypes nor the very few published systems that support motion sketch-based video retrieval [1, 5] have evolved into open source software.

7. DOWNLOAD AND DEPLOYMENT

The vitrivr stack can be downloaded from our project website https://vitrivr.org. To facilitate the deployment, we offer a docker[7] image containing an ADAM database prepopulated with features from a video collection comprised of creative commons videos [9] together with the relevant thumbnails.

To run the vitrivr stack, the following components have to be present on a system:

- Docker (unless the database is installed natively)
- Java 8
- a PHP-capable web server

8. FUTURE WORK

Future work is planned in all three areas covered by the vitrivr stack, i.e., at the frontend, query processing, and data storage layer.

ADAM We are currently in the process of migrating the vitrivr stack from ADAM to its successor, called ADAM$_{pro}$ [4], in order to improve the scalability to even larger multimedia collections and to ensure flexibility in the retrieval plans. ADAM$_{pro}$ is a new storage system capable of storing and efficiently retrieving feature data of arbitrary dimensionality as well as structured metadata. It supports a large variety of index structures (VA-File, Locality-Sensitive Hashing, Spectral Hashing, etc.) and allows a flexible combination of multiple indexes and data sources.

Cineast currently focuses mainly on video. We plan to change this by adding additional feature modules for visual, auditory and spatio-temporal modalities. Additionally, a distributed extraction system is currently being developed which will facilitate feature extraction on large collections across many machines.

Frontend The vitrivr frontend will be extended by a natural-language interface as well as additional query modes for speech and sound. Additional functionality is also planned to facilitate exploratory searches.

[6]http://server.imgseek.net
[7]https://www.docker.com

9. ACKNOWLEDGMENTS

This work was partly supported by the Swiss National Science Foundation (SNSF), project IMOTION (contract no. 20CH21_151571).

10. REFERENCES

[1] S.-F. Chang, W. Chen, and H. Sundaram. VideoQ: A Fully Automated Video Retrieval System Using Motion Sketches. In *Proc. Int. Workshop on Applications of Computer Vision, WACV 1998*, pages 270–271, 1998.

[2] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, et al. Query by Image and Video Content: The QBIC system. *IEEE Computer*, 28(9):23–32, 1995.

[3] I. Giangreco, I. Al Kabary, and H. Schuldt. ADAM — A Database and Information Retrieval System for Big Multimedia Collections. In *Proc. Int. Congress on Big Data (BigData Congress)*, pages 406–413, Anchorage, USA, 2014.

[4] I. Giangreco and H. Schuldt. ADAM$_{pro}$: Database Support for Big Multimedia Retrieval. *Datenbank-Spektrum*, 16(1):17–26, 2016.

[5] R. Hu, S. James, T. Wang, and J. Collomosse. Markov Random Fields for Sketch based Video Retrieval. In *Proc. Int. Conf. on Multimedia Retrieval, ICMR 2013*, pages 279–286, 2013.

[6] P. Over, G. Awad, M. Michel, J. Fiscus, W. Kraaij, A. F. Smeaton, G. Quéenot, and R. Ordelman. TRECVID 2015 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. In *Proc. of TRECVID 2015*. NIST, USA, 2015.

[7] L. Rossetto, I. Giangreco, S. Heller, C. Tănase, and H. Schuldt. Searching in Video Collections Using Sketches and Sample Images–The Cineast System. In *International Conference on MultiMedia Modelling, MMM 2016*, pages 336–341, Miami, USA, 2016.

[8] L. Rossetto, I. Giangreco, and H. Schuldt. Cineast: A Multi-Feature Sketch-based Video Retrieval Engine. In *Proc. Int. Symposium on Multimedia, ISM 2014*, pages 18–23, Taichung, China, 2014.

[9] L. Rossetto, I. Giangreco, and H. Schuldt. OSVC — Open Short Video Collection 1.0. Technical report, (CS-2015-002), University of Basel, 2015.

[10] K. Schoeffmann, M. A. Hudelist, and J. Huber. Video Interaction Tools: A Survey of Recent Work. *ACM Computing Surveys (CSUR)*, 48(1):14, 2015.

[11] R. Weber, H.-J. Schek, and S. Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In *Proc. Int. Conf. on Very Large Data Bases, VLDB 1998*, pages 194–205, New York, USA, 1998.