TalkMiner: A Lecture Video Search Engine

要　旨

Web配信されるレクチャー・ビデオを対象とした検索エンジンの設計と実装について述べる。検索可能なテキスト・インデックスを作成することで、YouTubeをはじめとするさまざまなWebサイトにあるレクチャー・ビデオの中で使われている資料を見つけることができる。このインデックスは、ビデオ中に表示されるプレゼンテーション・スライドのテキストから、また利用可能なタイトルや要約などの関連するメタデータから構築される。

ビデオの連続した流れの中でスライドを自動的に識別するためにはさまざまな困難が伴う。例えば、話者とスライドを同時表示するpicture-in-picture、カメラの切替り、スライド・ビルドなど、スライド画像を抽出する基本アルゴリズムだけでは対応が難しい。本稿ではこれらの課題に対応し、スライド識別の性能を改善する拡張アルゴリズムについて述べる。

アルゴリズムの評価と検索エンジンの有用性を試験するためにwww.talkminer.comを公開している。これまでにさまざまなサイトで公開されている17,000を超えるレクチャー・ビデオのインデックス作成を完了している。

Abstract

The design and implementation of a search engine for lecture webcasts is described. A searchable text index is created allowing users to locate material within lecture videos found on a variety of websites such as YouTube and Berkeley webcasts. The searchable index is built from the text of presentation slides appearing in the video along with other associated metadata such as the title and abstract when available.

The automatic identification of distinct slides within the video stream presents several challenges. For example, picture-in-picture compositing of a speaker and a presentation slide, switching cameras, and slide builds confuse basic algorithms for extracting keyframe slide images. Enhanced algorithms are described that improve slide identification.

A public system was deployed to test the algorithms and the utility of the search engine at www.talkminer.com. To date, over 17,000 lecture videos have been indexed from a variety of public sources.

Authors

John Adcock[*1]
Matthew Cooper[*1]
Laurent Denoue[*1]
Hamed Pirsiavash[*2]
Lawrence A. Rowe[*1]

[*1] FX Palo Alto Laboratory, Inc
[*2] University of California, Irvine

1. Introduction

Lecture videos or webcasts are increasingly available on the Internet. These webcasts may be class lectures (e.g., Berkeley webcasts, Kyoto OCW), research seminars (e.g., Google Tech Talks, PARC Forum, etc.), or product demonstrations or training. Many of these video lectures are organized around typical slide-based presentation materials (e.g.: PowerPoint).

Conventional web search engines can find a webpage with a lecture on it if the searcher includes “webcast" or “lecture" among the search terms, or searches on a website oriented towards video lecture content (e.g: AcademicEarth). This type of search can be successful if the hosting website includes a detailed textual description of the contents of the video, indexable with standard text-based search technology. But simply locating a topical video may not be satisfactory. Users, particularly students, need to find the precise points inside a video when an instructor covers a specific topic within a lecture (e.g., course requirements or discussion of a specific concept). Answering these needs requires a search engine that can analyze the content *inside* the webcast, expose it to search, and allow a searcher to seek to the interesting portion of the video.

TalkMiner proposes to address this need. It processes lecture videos to identify slide images, to extract text using optical character recognition (OCR), and to build a text search index from the information it recovers.

This video processing task can be challenging. Image quality of web videos is often poor, lighting is inconsistent, and the relative positions of the camera and projection screen are unpredictable. Also, many lecture videos employ production elements including cuts between multiple cameras and picture-in-picture compositing.

TalkMiner uses automatic video analysis methods developed to robustly identify slides in lecture video despite the wide variety of video capture practices and production techniques in use.

TalkMiner is not a video hosting system; it does not retain or serve a copy of the original video files. The videos are processed to identify their distinct slides and corresponding time codes, and a search index is constructed from text recovered from the slides. When a user reviews a lecture, the video is embedded from the original hosting website. As a result, storage requirements for our system are minimal. The current system has indexed over 17,000 lecture videos but only consumes approximately 20 GB of storage.

This report describes the design and implementation of the TalkMiner system. It is organized as follows. Section 2 describes the basic operation of the interactive search system. Section 3 reviews related work in this area. Section 4 details the implementation of the system including the back-end slide detection algorithms and the front-end web-based search system. We also review some experiments validating our slide detection scheme. Section 5 concludes the paper with a summary.

2. User Experience

The TalkMiner search interface resembles typical web search interfaces. The user enters one or more search terms and a list of talks that match those terms in the title, abstract, or presentation slides are listed as in Figure 1. Notice that search term matches are highlighted.

The information displayed for each talk on the search results page(s) includes a representative keyframe, the date and title of the lecture, and the channel (or source) of the talk. Other metadata shown includes the duration of the talk, the number of slides, the publication date, and the date of indexing by TalkMiner. A link is provided to the original page on which it was published.

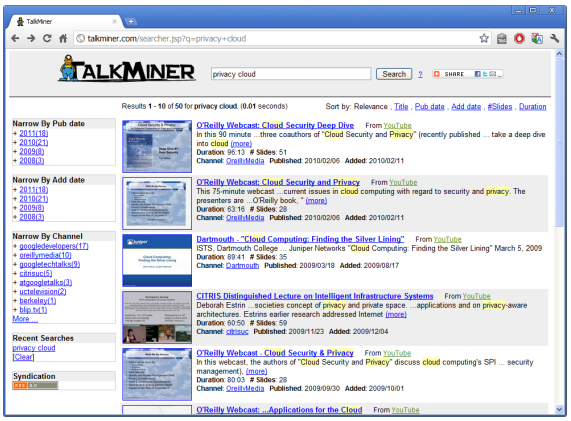

Figure 1. Search results page.

By default, talks are ordered by relevance to the query terms. Other sortable attributes include publication date, number of slides, and duration. The search results page also includes controls to filter the search results according to specific criteria (e.g., the year of publication, the channel, etc.).

From the search results page the user can click through to the detailed view for a specific talk, as shown in Figure 2. The right side of the interface contains the embedded video player above the lecture title and abstract. The left of the page displays detected slide thumbnails extracted from the video. Slide thumbnails are highlighted with a yellow border to indicate matches with at least one of the search keywords. Selecting a keyframe cues the video to that specific time point in the embedded player on the right.

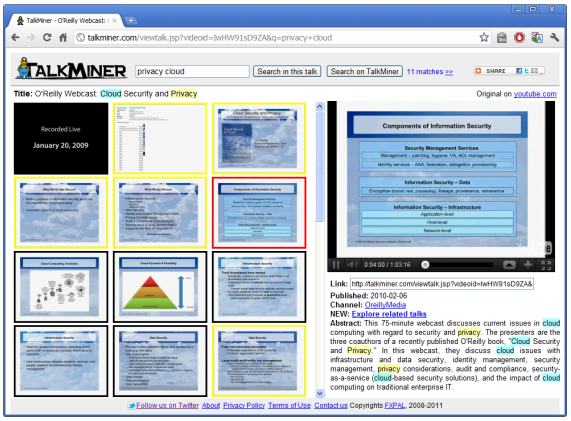

Figure 2. Viewing slides and using them to control the embedded player.

3. Related Work

Production of rich media lecture webcasts can be challenging. Collections of academic lecture material can be found on a few websites [22, 19, 2, 24] but the experience is rarely more advanced than a video with a basic seek and playback interface. When an enhanced presentation of the lecture is made available, it is generally the product of manual authoring that may require the original presentation materials (e.g.: PowerPoint). Research groups and commercial companies have created applications to produce high quality rich media lecture webcasts, but most solutions impose constraints to simplify content production [21].

Given a source lecture video, the first task is the detection of presentation slides. Slide time codes can be acquired by manual annotation, or for an automatic solution, time codes can be recorded by instrumenting the presentation system [26]. These solutions fail when non-preconfigured PCs, such as a guest lecturer's personal laptop, or unsupported presentation software is used. Another solution is to capture slide images by mounting a still camera in front of the projection screen or by installing an RGB capture device between the presentation PC and the projector [7, 23]. Solutions that use dedicated video cameras and room instrumentation can produce rich lecture records, but are notoriously expensive to install, operate, and maintain [1, 18, 5, 17].

Systems that process lecture videos or projector image streams typically use simple frame differencing techniques and assume that the location of the slides within the video frame is known and fixed, and that slides dominate the video frame [5, 18, 7, 23]. Haubold and Kender [9, 10] focused on multi-speaker presentation videos and developed an enhanced multimodal segmentation using the audio stream to detect speaker changes. The use of audio for segmentation has also been studied in [13] for

lectures and in more general purpose video retrieval systems [12, 20]. Our work also employs basic visual frame differencing as a first step, which we extend with both spatial filtering and speaker appearance modeling to reduce the number of non-slide images in the final set of keyframes.

After capture, the next step is to build a text index for retrieval. Systems that rely on instrumenting the presentation software or assume the availability of original presentation files can extract the original slide text [26, 14, 15] for indexing. Other work uses automatic speech recognition (ASR) to build a time-stamped transcript of the lecture for retrieval [20, 11, 12]. This can be error-prone for videos with low-quality audio or when speaker and vocabulary adaptation is not possible. Some research has employed natural language processing to improve the accuracy and searchability of ASR output [10, 13].

Optical character recognition (OCR) has also been used to recover the slide text for indexing [7] sometimes in combination with ASR [11]. Vinciarelli and Odobez [23] described techniques for improving OCR for slide retrieval with performance competitive with using the original slide text. These systems use images captured directly from the presentation projector. Such images are commonly higher resolution than webcasts, and include less non-slide content. The TalkMiner system accommodates generic, low quality production video with large amounts of non-slide content.

In summary, robustly producing a rich media lecture webcast is complicated and expensive. The simplest and least expensive approach to lecture capture, and by far the most widely used capture technology, is a single video camera pointing at the speaker and a screen on which presentation material is projected. TalkMiner aims to process this basic video input to automatically identify slides and their time codes, and create a useful text index and interactive browsing system.

4. System Description

This section details the design and implementation of TalkMiner including the system architecture, video indexing, and retrieval interface. An overview of the architecture appears in Figure 3. The system has two main components: the back-end video processing and the front-end web server. The back-end searches the web for lecture webcasts and indexes them. It executes algorithms to identify slide images and applies OCR to create the keyword search index.

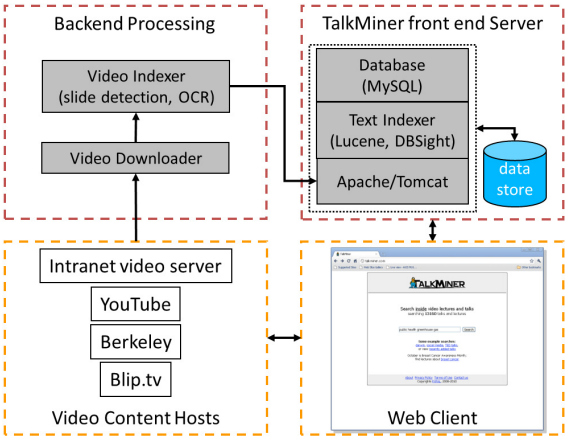

Figure 3. System architecture.

4.1 Aggregating Talks

The system gathers lecture videos from three sites: YouTube [25], U.C. Berkeley [3], and blip.tv [4]. The contributions by site are shown in Table 1.

TalkMiner finds and processes an average of 29 new videos per day. Videos are identified for download by parsing RSS feeds to which the

Table 1. Sources of indexed content.

Website	Videos Indexed
YouTube	13438
Berkeley	3583
Blip.tv	205
Total	17246

system is subscribed. Once downloaded, it takes roughly 15 minutes to process a 60 minute talk (i.e., 25 percent of the video duration). The system is generally limited by download speed rather than processing speed. Note that the 29 videos per day figure mentioned above is not the computational capacity of the system. Using the modest resources currently allocated to it, hundreds of videos can be processed daily.

New videos are registered for processing by creating records in the MySQL database (shown in the front-end portion of Figure 3). The MySQL database maintains talk metadata in a single table. Stored information includes the URL of the original media, the processing status of the video, the title, description, publication date, number of slides, duration, width and height of the video, the date the talk was added to TalkMiner, and the date it was indexed. Each record also contains the text recovered by OCR for the slides in the presentation.

4.1.1 Talk Classification

Of the many videos available online, only a relatively small proportion are good candidates for indexing by TalkMiner. The automatic crawling process while aimed at primarily lecture sources, does include some material which is not well suited to TalkMiner, such as interviews. Because the downloading process is the bottleneck of the system, there is an incentive to eliminate poorly suited content before investing the time and bandwidth to download the entire media file. We've implemented a classification system that determines suitability for inclusion in TalkMiner based on only the first few minutes of video. As a video is being downloaded it is tested. If it fails the test the download is abandoned.

To build this video classifier we used 200 slide-based videos and 100 non-lecture videos. We extracted features from the first 5 minutes of each video to train support vector machine (SVM) classifier to distinguish lecture and non-lecture content. The features used were: the total duration of detected stationary content, the number of stationary segments, the average length of stationary segments, and the minimum and average entropy of the projection of vertical edges. This last measure is a low-cost proxy for the presence of text. Detection of stationary content was performed using the frame differencing methods described below in Section 4.2. Using a leave-one-out method for evaluation we achieved 95% video classification accuracy. By treating the classification score as a confidence measure and rejecting classifications with low confidence, we achieved 98% classification accuracy with a 9% rejection rate.

4.2 Slide Detection

The principal goal of our analysis is to detect slides within the lecture videos. Slide images serve as ideal keyframes for lecture video for several reasons. Slides provide immediate visual context for the lecture allowing simple visual preview without audio or video playback. Slides contain text which is extracted by OCR to generate the index for text-based search. Finally, slides are commonly used by lecturers to organize their presentations and therefore delimit topically coherent portions of lectures.

The slide identification task is quite different from traditional video shot segmentation. The goal is to determine the temporal segment associated with a specific slide. In the simplest case, the slide will appear throughout the video segment. More generally, a slide segment may contain multiple shots of the same slide, possibly mixed with shots of the speaker, the audience, or other content. Detected slide keyframes organize both our interface and our indexing. Thus, we associate slide keyframes with corresponding time segments to facilitate browsing and interaction.

4.2.1 Frame Differencing

We initially used the frame difference-based analysis from ProjectorBox [7] for slide extraction for TalkMiner. We extract one frame per second from the source video, take the difference between subsequent frames, and apply a threshold to the differences to find changed pixels. If 1% of the pixels in the frame exceed the change threshold, a new keyframe is extracted. If this keyframe does not change for three seconds we save it as a slide. In other words we extract a keyframe for any video segment that is stable for at least 3 seconds. This basic approach produces satisfactory results for many videos, and works well for videos in which the slides fill the video. However, we have extended this approach to address several frequently occurring cases for which it produces poor results: videos with shots of the speaker that contain neither distinctive visual information for keyframe browsing nor useful text for indexing, videos with shots with secondary small “picture-in-picture" (PIP) streams, and videos with shots from the back of the room that include motion by the speaker or the audience.

In these cases, basic frame differencing often selects extraneous keyframes that may contain non-slide content or multiple copies of the same slide, or if portions of the frame exhibit continuous motion then slides can be missed as the scene is never stable for three seconds.

4.2.2 Spatial Filtering

Our first enhancement uses spatial information to improve the frame differencing. As before, we sample one frame per second from the original video and apply a blur to smooth the frames. We then determine the changed pixels, as above. We count the changed pixels in the central area of the frame, assuming that sufficient change in this block indicates a slide change. This is motivated by the observation that the projector screen on which slides are displayed typically occupies the center of the frame.

We then use connected components analysis to group changed pixels into regions within the frame. If the area of a connected component is below a threshold, we ignore that region as a spurious change. We next calculate the bounding box that encompasses all the remaining change regions (connected components) in the frame. If this bounding box is small in comparison to the frame size, the image is not selected as a new slide keyframe. Rather it is assumed that some other change in the frame such as an inserted view of the speaker, or speaker or audience motion is responsible for the pixel differences.

This spatial analysis correctly excludes many of the duplicated keyframes found using simple frame differencing due to picture-in-picture (PIP) video. Usually, PIP insertions occupy a small portion of the frame, and the bounding box for any motion within them is smaller still. Also, after eliminating small change areas, the requirement for stable content for at least three seconds in our inter-frame difference analysis no longer misses slide keyframes due to continuous motion within a PIP feed or in videos shot from the back of a room with visible audience motion. In sum, this approach improves both the precision and the recall of our slide detection.

4.2.3 Speaker Appearance Modeling

To reject spurious keyframes when the speaker appears in all or part of the frame, we automatically learn an appearance-based model to separate speaker views from slides.

We randomly sample 400 video frames and use both face detection and OCR results to group the frames as illustrated in Figure 4. Frames exhibiting motion, with detected faces, and without detected text are used to model speaker appearance. Stationary frames with detected text and those frames that are stable

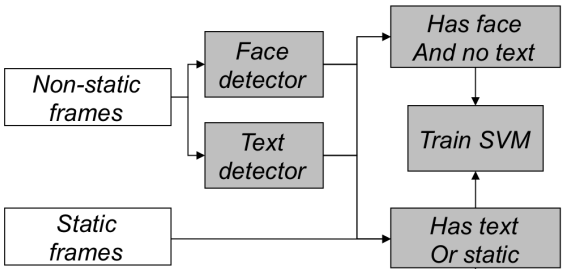

Figure 4. Appearance modeling for keyframe filtering.

for over five seconds are used to model slide appearance. Color histograms are computed from these two sets of frames and used to train a SVM classifier to distinguish the speaker frames and the slide frames. We use the entire frame to calculate the histogram features. As a result, the appearance of both the speaker and the lecture room or podium area are typically incorporated in the speaker training frames. We discard any candidate keyframes that are classified as members of this speaker/background class.

4.2.4 Build-up Material

Another common practice in slide presentations is the progressive build-up of a complete final slide over a sequence of partial versions. An example appears in Figure 5. Our basic difference-based slide detection tends to represent each distinct step in a build-up sequence with an individual keyframe. In general, we would rather show a single slide containing all of the content rather than each partial version.

We combine a sequence of slides that contain built-up content as illustrated in Figure 5. We first find the difference between two adjacent candidate slide keyframes. We compute the difference between the two keyframes and find the bounding box of any change. If this region in the first slide lacks significant edges (i.e.: was empty), and in the second slide contains strong edge content (i.e.: is no longer empty), we determine that this pair is part of a build-up sequence. This pairwise analysis is iterated, and the last keyframe of such a sequence is used to represent the entire build-up sequence. This creates a more compact visual index without losing any informative content or text. By using image analysis to judge whether the change has added content rather than relying on comparing the output of OCR on the two keyframes, we avoid problems with OCR accuracy and also handle the case where the added content is non-textual in nature.

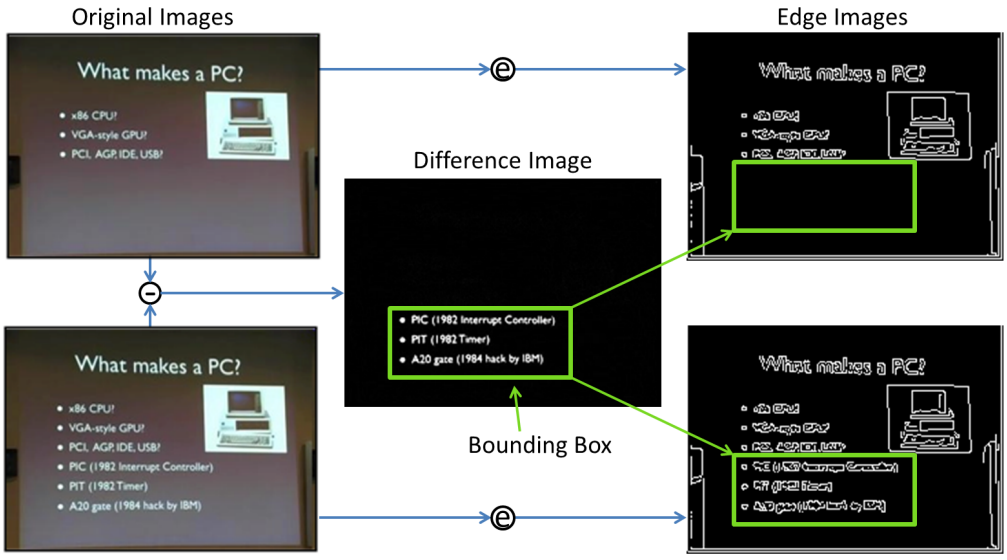

Figure 5. An example build-up sequence detection. The original slides are on the left. The bottom slide has material added to the top slide. In the center the difference between the slides is shown with a green bounding box around the changed region. On the right is the result of edge detection on the original images with the bounding box overlayed showing how the bottom image contains more edge pixels within the region.

4.2.5 Slide Detection Evaluation

The keyframe selection algorithm we have developed operates in the three phases described above. The first step is frame differencing with spatial filtering. Then we train our speaker appearance model and reject extraneous frames. Finally we apply the build-up detection to join together sequences of build-up slides.

To evaluate performance, we annotated 21 videos from our repository and manually marked the desired slide keyframes and segments. Then, we applied both the basic frame-differencing approach of [7] and the enhanced algorithm described above to this set of videos. Using the ground truth we measured precision and recall of the extracted slides. Precision decreases with over-segmentation and extraneous keyframes. Recall decreases with under-segmentation and missed keyframes. The F1 score is the geometric mean of precision and recall:

$$\text{Precision} = \frac{\#\text{ correctly detected slides}}{\#\text{ detected slides}}$$

$$\text{Recall} = \frac{\#\text{ correctly detected slides}}{\#\text{ ground truth slides}}$$

$$\text{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Table 2 summarizes the results of this evaluation. As noted previously, the main drawbacks of the basic frame differencing were duplicate keyframes mistakenly extracted due to intermittent motion and missed slides due to continuous small motion. The results using our extended algorithm show improvement in both precision and recall values, with substantial improvement in precision. The size of the resulting set of automatically extracted keyframes is much closer to the length of the ideal summary represented by the ground truth comprised solely of distinct slide keyframes.

Table 2. Experimental results for keyframe selection comparing frame difference-based keyframe selection (Basic) with our extended algorithm (Ext.).

	Basic	Ext	change
Ave Precision	47.5	76.9	84.3%
Ave Recall	91.7	94.0	3.3%
F1 Score	60.8	83.9	48.5%

	Basic	Ext	Truth
# Slides	106.4	63.7	52.7

4.3 Front End

The TalkMiner web-based front end, depicted in the top right of Figure 3, is implemented with Java server pages (JSP) running on an industry standard Apache/Tomcat combination. The indexing and search framework, implemented with DBSight [6], runs in the same Tomcat instance. At runtime, searches are performed with the DBSight web application using previously computed Lucene [16] indexes of the talk database. At indexing time, the DBSight framework provides a bridge between the information in the database and the Lucene system. Each row or presentation in the database corresponds to a document in the Lucene index, and search operations return a list of presentations with a common unique ID linking the search record in Lucene with the database record in MySQL. The search results list, depicted in Figure 1, is rendered by customized FreeMarker [8] web page templates. The query, sort order, and search filters are encoded as parameters of the search page URL, making searches easily bookmarkable. Each item in the search results list links to the detailed talk viewing page for that talk, as shown in Figure 2. This page is a JSP parameterized by the unique video id and the query string. This page retrieves the talk text and slide timing information from the MySQL database and renders the page including the slide thumbnails and embedded Flash video player. JavaScript actions in the page control the embedded video player in response to user

selection of slide thumbnails, and highlight matching slide thumbnails in response to text queries.

5. Conclusion

TalkMiner is a rich search and browsing system designed to enhance access to and usage of lecture webcasts. The system leverages existing online video distribution infrastructure to embed webcasts from numerous sources in a unified browsing and search interface. TalkMiner's automatic content analysis and interface technology enables users to rapidly locate specific information within videos of interest. TalkMiner's highly scalable design maintains a minimal computational and storage footprint. We have described algorithms for robust slide identification in a variety of ad-hoc video capturing scenarios. Text extracted from detected slides is used for index construction both across and within webcasts. TalkMiner assumes content captured according to common practices requiring neither dedicated lecture-capture systems nor any careful post-capture authoring. Thus, the system is capable of including a greater volume and greater variety of existing and newly created content at a much lower cost than would otherwise be possible.

We deployed our system publicly in 2010 at www.talkminer.com and continue to expand our collection of indexed content. We hope to explore interactive extensions to our system to enhance our indexing and video processing. We have outlined some initial explorations in this area, and expect to update TalkMiner further according to feedback from users and content creators in response to its public release.

This work is based on an earlier work: TalkMiner: a lecture webcast search engine, in MM '10 Proceedings of the international conference on Multimedia 2010, © ACM 2010 http://doi.acm.org/10.1145/1873951.1873986

6. TRADEMARKS

- Academic Earth is a trademark of Academic Earth LLC.
- Apache, Tomcat and Lucene are trademarks of the Apache Software Foundation.
- Blip.TV is a trademark of Blip Networks, Inc.
- Google, Google Tech Talk and YouTube are trademarks or registered trademarks of Google Inc.
- MySQL is a trademark of Oracle Corp. and/or its affiliates.
- PARC is a trademark of Palo Alto Research Center, Inc.
- PowerPoint is a trademark of Microsoft Corp.
- Flash is trademark of Adobe Systems Inc.
- All brand names and product names are trademarks or registered trademarks of their respective companies.

7. References

[1] G. Abowd. Classroom 2000: an experiment with the instrumentation of a living educational environment. IBM Systems Journal, 38(4):508-530, 1999.

[2] Academic Earth. http://academicearth.org.

[3] Berkeley Webcasts. http://webcast.berkeley.edu.

[4] Blip TV. http://www.blip.tv.

[5] P. Chiu, A. Kapuskar, S. Reitmeier, and L. Wilcox. Room with a rear view: Meeting capture in a multimedia conference room. IEEE MultiMedia, 7:48-54, 2000.

[6] DBSight. http://www.dbsight.net.

[7] L. Denoue, D. Hilbert, D. Billsus, and M. Cooper. Projectorbox: Seamless presentation capture for classrooms. In

World Conf. on E-Learning in Corporate, Government, Healthcare, and Higher Education, 2005.

[8] FreeMarker. http://freemarker.sourceforge.net.

[9] A. Haubold and J. R. Kender. Augmented segmentation and visualization for presentation videos. In MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia, pages 51-60, New York, NY, USA, 2005. ACM.

[10] A. Haubold and J. R. Kender. Vast mm: multimedia browser for presentation video. In CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval, pages 41-48, New York, NY, USA, 2007. ACM.

[11] A. G. Hauptmann, R. Jin, and T. D. Ng. Multi-modal information retrieval from broadcast video using ocr and speech recognition. In JCDL '02: Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries, pages 160-161, New York, NY, USA, 2002. ACM.

[12] A. G. Hauptmann and H. D. Wactlar. Indexing and search of multimodal information. In IEEE Intl. Conf. on Acoustics Speech and Signal Processing, volume 1, pages 195-198, 1997.

[13] T. Kawahara, M. Hasagawa, K. Shitaoka, T. Kitade, and H. Nanjo. Automatic indexing of lecture presentations using unsupervised learning of presumed discourse markers. IEEE Trans, on Audio, Speech, and Language Processing, 12(4):409-419, July 2004.

[14] A. Kushki, M. Ajmal, and K. N. Plataniotis. Hierarchical fuzzy feature similarity combination for presentation slide retrieval. EURASIP J. Adv. Signal Process, 2008:1-19, 2008.

[15] G. M. Liew and M.-Y. Kan. Slide image retrieval: a preliminary study. In JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries, pages 359-362, New York, NY, USA, 2008. ACM.

[16] Apache Lucene. http://lucene.apache.org

[17] Mediasite by Sonic Foundry. http://www.sonicfoundry.com/mediasite.

[18] S. Mukhopadhyay and B. Smith. Passive capture and structuring of lectures. In MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia (Part 1), pages 477-487, New York, NY, USA, 1999. ACM.

[19] Omnisio. http://www.omnisio.com/.

[20] D. Ponceleon, A. Amir, S. Srinivasan, T. Syeda-Mahmood, and D. Petkovic. Cuevideo: automated multimedia indexing and retrieval. In MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia (Part 2), page 199, New York, NY, USA, 1999. ACM.

[21] L. A. Rowe, D. Harley, P. Pletcher, and S. Lawrence. Bibs: A lecture webcasting system. Technical report, Center for Studies in Higher Education University of California, Berkeley, 2001.

[22] VideoLectures.NET. http://videolectures.net/.

[23] A. Vinciarelli and J.-M. Odobez. Application of information retrieval technologies to presentation slides. IEEE Trans. on Multimedia, 8(5):981-995, 2006.

[24] YouTube.EDU. http://www.youtube.com/edu.

[25] YouTube. http://www.youtube.com/.

[26] P. Ziewer. Navigational indices in full text search by automated analyses of screen recorded data. In Proc. E-Learn 2004.

Author's Introductions

John Adcock
FX Palo Alto Laboratory
Area of specialty: Electrical Engineering (Ph.D.), Multimedia Systems

Matthew Cooper
FX Palo Alto Laboratory
Area of specialty: Electrical Engineering (Ph.D.), Multimedia and Machine Learning

Laurent Denoue
FX Palo Alto Laboratory
Area of specialty: Computer Science (Ph.D.), Human Computer Interaction

Hamed Pirsiavash
University of California, Irvine
Area of specialty: Computer Science, Multimedia Processing

Lawrence A. Rowe
FX Palo Alto Laboratory
Area of specialty: Computer Science (Ph.D.), Multimedia Systems