Automated Text Extraction And Indexing Of Video Presentation Recordings For Keyword Search Via A Web Interface

Martina Welte Thomas Eschbach Bernd Becker
Institut for Computer Science
Albert-Ludwigs-Universität Freiburg, Germany
{welte, eschbach, becker}@informatik.uni-freiburg.de

Abstract: University lecture recordings allow students to review lectures anytime and anywhere and thus potentially improve the quality of teaching. In this paper we address the problem of constructing a text search index for presentation recordings created with a frame grabbing tool and how to answer keyword search queries based on this index. Due to the problems of OCR based methods, an alternative approach to this problem is given. A platform independent access to the database was implemented. Our experimental results show the effectiveness and feasibility of the approach.

1 Introduction

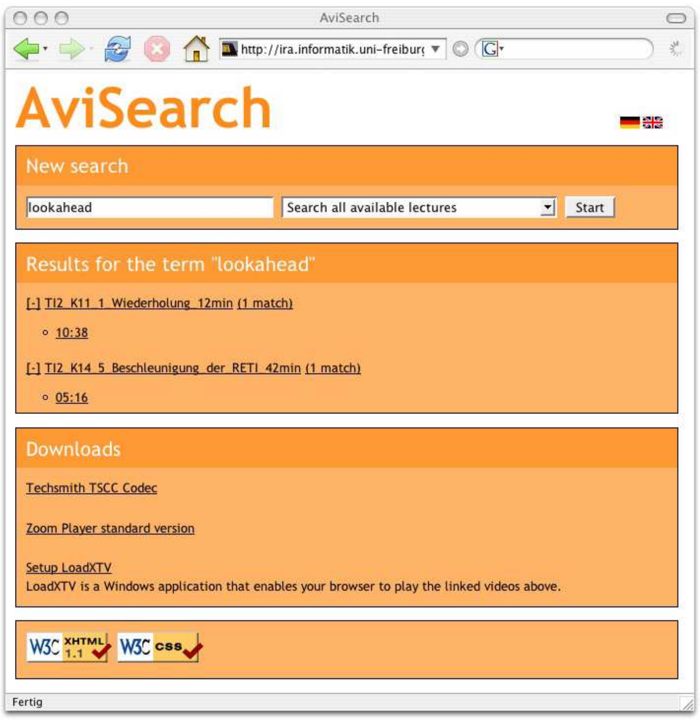
In the last few years the amount of multimedia material concerning university lectures has highly increased. The effort of the Albert-Ludwigs-Universität Freiburg to provide additional multimedia material [vir, uli, fmo] for a growing number of lectures is very popular among both students and lecturers [Bec05]. There are several tools available that are able to create recordings of lectures, such as *Camtasia Studio* [cam] by TechSmith, *Lecturnity* [lec] from imc, and *TeleTeachingTool* [ttt].

A common method to obtain a video recording of the lecture is to capture the screen of the presentation computer at frequent intervals and combine this with a voice recording of the lecturer; *Camtasia Studio* [cam] uses this approach. In contrast to classical, non-multimedia lectures for which only books or slides are provided, this method provides the possibility to reexamine the actual lecture, showing the slides together with the lecturer's explanation, (software) demonstrations and handwritten annotations anytime and anywhere. The tool allows to capture the entire screen without any restriction for the presenter on what he is showing to the students. In this *bitmap based* approach, the single frames of the video are stored as bitmaps and contain no further information of *what* is shown. Due to that, until now it was not possible to search for a given phrase other than to *watch* every presentation recording and look for an occurrence of the phrase.

A different approach is to store the single *objects* (like text blocks, graphics etc.) together with their position shown on a slide; this method is used by *Lecturnity* [HMO04]. The actual look of the slide is constructed based on this information just for playback. For this *object based* approach full text search can be easily implemented, since the text information is directly embedded in the file. The drawback of this method is that special software has to be used for both recording and playback and it is not trivial to convert existing *bitmap based* presentation recordings into these specialized *object based* formats in an automated way.

For this work, our goal was to remedy the drawback of *bitmap based* lecture recordings, providing a methodology of creating a text search index for a given set of presentation recordings and answering search queries based on this index. Our approach makes use of additional information that can be extracted from the slides used in the presentation. Notice that for this approach, an automatic and robust method to extract and index text for large numbers of already existing presentation recordings has to be created.

To satisfy the *anytime anywhere* concept, a web interface (alike Google [goo]) is provided. The result page shows the presentation recordings matching a given search query plus the time indices at which the matching text is shown. Figure 1 shows an exemplary result page using the Mozilla browser [moz]. By simply using the hyperlink on the result page, a playback of the presentation recording is automatically invoked. For a comfortable handling, the proprietary client *LoadXTV* is provided: The program parses the link on the result page, automatically downloads the video and invokes a playback starting at the given time.



AviSearch

http://ira.informatik.uni-freiburg

AviSearch

New search

lookahead

Search all available lectures

Start

Results for the term "lookahead"

[-] TI2_K11_1_Wiederholung_12min (1 match)

- 10:38

[-] TI2_K14_5_Beschleunigung_der_RETI_42min (1 match)

- 05:16

Downloads

Techsmith TSCC Codec

Zoom Player standard version

Setup LoadXTV
LoadXTV is a Windows application that enables your browser to play the linked videos above.

Fertig

Figure 1: The web interface shows the results of the search for the term “lookahead”, which was found in two videos at time 10:38 minutes resp. 5:16 minutes.

2 Approach

We now present our approach to process both the lecture recording and the slides used in this recording to determine the text shown on a video frame. We then discuss how to use this knowledge to build a text search index for the video file.

For the presentation, the lecturer usually uses a set of slides in either PDF[1] or Powerpoint format (in the following, the PDF format is assumed, since most other formats can be easily converted to PDF using free or inexpensive software). While the lecturer is giving his presentation, a screen capturing tool (e.g. Camtasia Studio) records both the shown slides and the audio. Figures 2 and 3 illustrate this procedure.

We now address the problem of building the keyword-time-relation describing which words are shown at a given time [Hür03]. There are several methods to extract this information from the video, e.g. a *OCR based* approach (Fig. 2, see also [LE00, Zie04]) and a *PDF based* approach (Fig. 3).

With the given resolution of the video (800×600 or 1024×768), many available OCR tools (e.g. SimpleOCR [sim], JOCR/GOCR [joc], ScreenOCR [scr]) are not able to extract all the technical terms correctly. In the case that the contrast between background and font color is low, the error rate of this *OCR based* approach soon gets unacceptable. To overcome this problem, we developed a *PDF based* approach (Fig. 3). This method is described in the remainder of the paper.

In the PDF based approach, we conceptually decompose the videostream into a sequence of bitmaps (in the following referred to as *video bitmaps*). To decide which slide was presented at a given point of time, each video bitmap has to be compared to every bitmap we obtained by converting the single PDF slides (in the following referred to as *PDF bitmaps*).

The PDF format provides the possibility to extract the raw text information for a single slide using free tools such as Xpdf [xpd]. By doing this, we can access the text visible on a slide, and can thus conclude the keyword-time-relation and build a corresponding search index based on this relation.

To do this in an efficient way, it is essential to restrict the comparison mentioned above to

[1] Portable Document Format

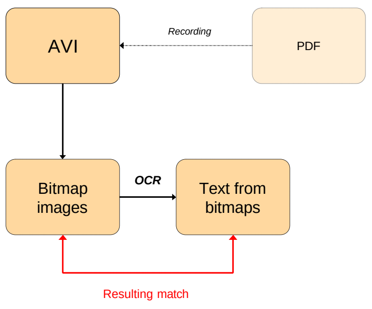


Figure 2: OCR based approach

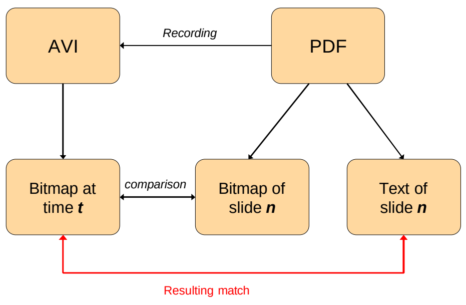


Figure 3: PDF based approach

video bitmaps that are likely to contain relevant information, such as video bitmaps that are placed immediately after a slide transition. For this, a two-step algorithm is used:

1. In each time interval, consider only one bitmap (in our implementation, this time interval was set to 1 second). Slides containing important keywords are unlikely to be presented for just one second or even less.

2. Compare every bitmap with its direct predecessor and drop it if they are too similar. (We will address the problem of comparing two bitmaps later.)

   Please notice that it is not necessary for these bitmaps to be *identical* to represent the same slide, since the lecturer may insert annotations during the presentation (e.g. add some markings and explanations).

Considering this "purged set" of video bitmaps, we now perform a one-by-one comparison to the set of PDF bitmaps. If a video bitmap at time t is most similar to the PDF bitmap of slide n, we consider these two bitmaps a *match*.

In the case that two subsequent video bitmaps were recognized as different although it turns out that they match the *same* slide (e.g. a slide that was heavily annotated during recording), our approach omits the latter frame, since it does not contain any additional keyword and its addition to the index would only enlarge both the search index and the result table.

We now describe how to measure the similarity between two bitmaps (video or PDF). When Adobe Acrobat is used to show the slides in the presentation, a small black margin is added to the original slides that can also be found in the video bitmaps; in contrast to that, Xpdf adds a white margin to the PDF bitmaps. Due to that, we first have to crop the boundary of the bitmaps accordingly. The bitmaps are then scaled down to a common size (in our case 256×192) and finally the average error between the two bitmaps is computed.[2]

If there is no PDF bitmap that has a reasonable low average error to the considered video bitmap, it is skipped since we assume that in this time interval no PDF slide but something else (e.g. a software demonstration or a picture illustration) was shown in the presentation.

3 Experimental Results

A prototype implementation *AVISearch* written in Perl 5 was given as part of [Wel05]. *AVISearch* implements a fully automated analysis of a given set of lecture recordings by invoking the following programs:

- An extension of the free video processor *VirtualDub* [vd] is used to extract a video bitmap for each time interval.

[2]The average error is defined as the average over the squared difference obtained by a pixel by pixel comparison of the two bitmaps; e.g. the difference of a black to a white pixel is 1.

- *ImageMagick* [im] is used to convert PDF slides into bitmaps, to crop and resize bitmaps and to compare two bitmaps using the average error method.

  The following experiments show the distribution of differences measured with the average error between frames within the same lecture recording. They were conducted on recordings of the annual undergraduate course "Technische Informatik" [ti1, ti2] held by Bernd Becker.[3] Table 1 shows the similarity of subsequent bitmaps extracted from representative presentation recordings.

  In the first column the lower end of the average error interval is given; each average error has the length of $2 \cdot 10^{-4}$. The second and the third column give the number of bitmaps which are similar to their predecessor within the given average error interval. For "demo1" and "demo2", recorded with a screen resolution of 1024×768, a slide transition was assumed if the average error value was larger than $10 \cdot 10^{-4}$. For "demo3" and "demo4", recorded with 800×600, we chose an average error value of $20 \cdot 10^{-4}$ to assume a slide transition, since the 800×600 resolution provides significantly less pixels than the 1024×768 resolution.

- *Xpdf* [xpd] is used to extract the text from the slides.

- Finally an efficient database is created with *Glimpse* [gli].

The web interface, realized as a Perl based CGI script, relays the search query to *Glimpse*, parses its output and creates both a readable web page and the links needed for *LoadXTV*.

The Windows application *LoadXTV* parses the link on the result page, automatically downloads the video using *wget* [wge] and invokes a playback with the *Zoom Player* [zoo] starting at the given time. Similar applications for other operating systems can be easily created.

Table 1: Similarity of successive bitmaps extracted from presentation recordings.

average error	1024×768		800×600	
$(\cdot 10^{-4})$	demo1	demo2	demo3	demo4
0	2155	3417	1200	1469
2	100	25	101	101
4	26	2	122	123
6	6	0	288	227
8	3	0	497	432
10	0	0	64	39
12	0	0	16	14
14	0	1	13	13
16	0	0	5	10
18	0	1	4	3
20	0	1	5	4
22	0	0	0	1
24	0	0	4	1
26	0	0	3	0
28	1	1	1	1
$\geq$30	52	68	39	30

[3]"demo1" was extracted from the presentation recording "TI2_Fanout_Verzoegerungszeiten_39min.avi", demo2 from "TI2_Flipflops_58min.avi", demo3 from "TI1_Addierer_CSA_Komplexitaet_39min.avi", and finally demo4 from "TI1_Addierer_CarrySave_41min.avi".

4 Conclusion

We have presented a robust and efficient method to extract and index keywords from videos recorded with *Camtasia Studio* in an automated way. An index over an existing set of videos of the Albert-Ludwigs-Universität Freiburg was created and stored in a database. A web site which provides free searchable access to these recordings was implemented, a simple click on a result invokes the playback of the according video at the correct time.

Acknowledgement

The authors would like to thank Tobias Nopper for his contributions to this paper.

References

[Bec05] B. Becker. *F-MoLL : Freiburg Mobilität in Lehre und Lernen*. Bernd Becker (Eds.), 2005.

[cam] *http://www.techsmith.com*.

[fmo] *http://f-moll.uni-freiburg.de*.

[gli] *http://webglimpse.net*.

[goo] *http://www.google.com*.

[HMO04] W. Hürst, R. Mueller, and T. Ottmann. The AOF Method for Production, Use, and Management of Instructional Media. *In Proceedings of ICCE 2004, International Conference on Computers in Education, Melbourne*, 2004.

[Hür03] W. Hürst. Indexing, Searching, and Skimming of Multimedia Documents Containing Recorded Lectures and Live Presentations. *ACM MM'03*, November 2003.

[im] *http://www.imagemagick.org*.

[joc] *http://jocr.sourceforge.net*.

[LE00] R Lienhart and W. Effelsberg. Automatic Text Segmentation and Text Recognition for Video Indexing. *ACM/Springer Multimedia Systems Volume 8*, 2000.

[lec] *http://www.im-c.de*.

[moz] *http://www.mozilla.org*.

[scr] *http://www.screenocr.com*.

[sim] *http://www.simpleocr.com*.

[ti1] *http://ira.informatik.uni-freiburg.de/teaching/ti-1-2001-ws/videos_ti1/records.html*.

[ti2] *http://porta.informatik.uni-freiburg.de/lectures/TI2/02-ss/Vorlesungsaufzeichnung/video/records.html*.

[ttt] *http://teleteaching.uni-trier.de*.

[uli] *http://uli-campus.de*.

[vd] *http://www.virtualdub.org*.

[vir] *http://www.viror.de*.

[Wel05] M. Welte. Suchen in Presentation-Recordings am Beispiel der Technischen Informatik. *Albert-Ludwigs-Universität Freiburg*, Sept 2005.

[wge] *http://www.gnu.org/software/wget/wget.html*.

[xpd] *http://www.foolabs.com/xpdf/*.

[Zie04] P. Ziewer. Navigational Indices and Full Text Search by Automated Analyses of Screen Recorded Data. *E-Learn 2004, Washington, DC*, Nov 2004.

[zoo] *http://www.inmatrix.com/zplayer/*.