

Segmentation and Annotation of Audiovisual Recordings based on Automated Speech Recognition

Stephan Repp¹, Jörg Waitelonis², Harald Sack², and Christoph Meinel¹

¹ Hasso-Plattner-Institut für Softwaresystemtechnik GmbH

² Friedrich-Schiller-Universität Jena

Abstract. Searching multimedia data in particular audiovisual data is still a challenging task to fulfill. The number of digital video recordings has increased dramatically as recording technology has become more affordable and network infrastructure has become easy enough to provide download and streaming solutions. But, the accessibility and traceability of its content for further use is still rather limited. In our paper we are describing and evaluating a new approach to synchronizing auxiliary text-based material as, e. g. presentation slides with lecture video recordings. Our goal is to show that the tentative transliteration is sufficient for synchronization. Different approaches to synchronize textual material with deficient transliterations of lecture recordings are discussed and evaluated in this paper. Our evaluation data-set is based on different languages and various speakers' recordings.

1 Introduction

Audiovisual recordings in terms of streaming media or video podcasts (vodcasts) are increasingly used for live distance and on-demand lecturing by universities and distance learning institutions. Independent in/from time and place, learners have access to libraries of recorded lectures, often being organized as knowledge bases that offer their content in a well ordered and categorized manner. But, how can appropriate information be retrieved in a large lecture video data base in an efficient way? Manual segmentation of video lectures into smaller units, each segment related to a specific topic, is an accepted approach to find the desired piece of information [7, 14, 17].

Traditional multimedia retrieval based on feature extraction cannot be efficiently applied to lecture recordings. Lecture recordings are characterized by a homogeneous scene composition. Thus, image analysis of lecture videos fails even if the producer tries to loosen the scene with creative camera trajectory. A promising approach is based on using the audio layer of a lecture recording in order to get information about the lecture content. Unfortunately, most lecture recordings do not provide optimal sound quality and thus, also the effectiveness of automatic speech recognition (ASR) for the extraction of spoken words suffers even if a speaker-dependent system is used. The raw results of an untrained ASR applied

to lecture audio streams are not capable for an accurate indexing. Today, in lectures often text-based presentations such as MS Powerpoint or Portable Document Format (PDF) are used to support the teaching. The best alternative is thus to synchronize slides with presenter speech, which can be extracted using a speech to text engine.

Section 2 of the paper shows current approaches and outlines related work, while section 3 introduces the new segmentation method. Section 4 gives an evaluation of the algorithm and Section 5 concludes the paper with a short summary and an outlook on future work.

2 Related Work

Using speech recognition for indexing the lecture videos is an often used and evaluated procedure [10, 7, 14]. Text matching is a widely recognized method [12] too. The Text matching method has a direct impact on other fields such as genetic sequence alignment [11]. The main focus of these algorithms is to parse large strings on a block level and to find correct matches to short strings (gene). Much research has been focused on the area of intelligent meeting rooms and virtual classrooms [9, 6]. They usually try to minimize error in the speech transcription or recognition stages. Chu and Chen presented an approach for cross-media synchronization [4]. They match audio recordings with text transliterations of the audio recordings based on dynamic programming methods [13]. It differs from our approach. In our case, the content of the presentation slides and the transliteration of audio recording do not match. Chu and Chen make use of explicitly encoded events for synchronization, instead of implicit automated synchronization.

Another non-analytic approach is to synchronize presentation slides by maintaining a log file during the presentation thus keeping track of slide changes. But, most available lecture recordings do neither support desktop recordings nor maintain a dedicated log file. In [17], they synchronize the book sections with word blocks of the transcript. Yamamoto use a sliding window system and a vector model. They measure the precision and the recall of the results. Chen and Heng [2] also propose a method for automatic synchronisation of speech transcripts with presentation slides. After matching the speech transcript with the slides, redundancy and noise are removed by a fitting procedure. Finally, transitions of slides are approximated by using a progressive polynomial fitting.

Our algorithm lies downstream of these related works, assuming speech to text transcription from an out-of-the box commercial speech recognition software. Further we show, how linear text segmentation algorithms [5, 3] can be applied to segment lecture video recordings and to map presentation slides to a particular point of time in the video recording. Additionally we suggest a new algorithm, that deals with achieving global synchronization between the transcript and the presentation text. In the synchronization problem, our transcript input is expected to contain a large amount of error from the speech to text process. Our evaluation is divided into two stages:

First, the transliteration generated from the ASR is segmented by standard linear text segmentation algorithms while the text boundaries are

assumed as slide transitions. How can standard linear text segmentation algorithm be able to segment the erroneous transcript into coherent segments, without any additional recourse like the slide? Second, the presentation slides are used as an additional resource for the segmenting procedure and for the slide transition detection. We have implemented a vector-space-algorithm based on a sliding window system, as shown in [8, 17, 1] and a new algorithm for noise removal.

3 Algorithm Description

Synchronization is not a trivial problem because the input data from the speech to text process naturally contains many sources for error and confusion. Errors in this process reside in the speech to text (STT) engines inherent inaccuracy, the inability for us to train accurately the engine, poor sound quality from the lecture videos, and poor enunciation by the lecturer. The synchronization process faces inconsistencies produced by the way that a professor generally lectures and authors his lecture slides. The slides are not a direct script of the lecture, merely an outline. As the lecturer speaks, he elaborates on each topic and discusses related areas, while only referring to key terms from the slides. As a result, the lecture transcript deviates greatly from the slides, and rarely directly matches a continuous sequence of the slide text.

For our algorithm we assume the monotonicity of the progression of the slides. The most real life presentations follow the schedule given in the presentation slides. But, if the presenter skips some slides (slide-gap), our algorithm is able to resynchronize and if the presenter steps back to a previous slide, our algorithm is also able to resynchronize, when the regular schedule continues.

Figure 1 depicts the overall system architecture.

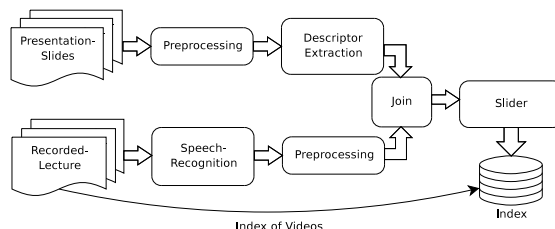


Fig. 1. Overall system architecture

Descriptor Extraction: The generation of keyword descriptors from the presentation slides is realised in the following way. The presentation slides are converted into the portable document format (PDF) and the plain text information is extracted from the PDF. To exclude nonrelevant words (stop words) the extracted plain text is processed by part-of-speech tagging. Words with high term frequency are removed.

Thus, only words with higher content separability remain in text. With term frequency and inverse document frequency lesser separating words can be identified and deleted. Generating the keyword stems (word stemming) to enhance the number of hits in the subsequent matching is not suitable, because stemming reduces the content separability of every word, which is unfavourable for further processing. Hence, for every set of presentation slides we obtain a revised list consisting out of slide numbers and keywords.

Speech Recognition: The lecture video recordings are analysed with an out-of-the-box speech recognition engine. This procedure is already discussed in detail in [16]. The engine needs a short neglectable training phase of 1-3 minutes for adapting the microphone to the ASR. In addition, the extracted slide descriptors complement the engine’s vocabulary, but these keywords are not trained by the speakers. The word accuracy is only about 20%-70% per transcript. The transcript consists of a list of words with the corresponding point in time, where the word was spotted in the speaker’s flow of words.

Join: The matching procedure works in the following way: (1) Extract all matches between the words from the generated transcript and the slides. This matching results in a tuple (time, slide number, word). (2) Use only the relevant matches for a word that matches at most four times. The parameter four was obtained during a training-phase. It is important to remove datasets with more than four hits per word, otherwise the step function will not be visible. Figure 2 shows the data before and after the filtering. The x-axis represents the time, the y-axis represents the slide numbers. After this procedure a new chronologically sorted list (time, slide number) is generated.

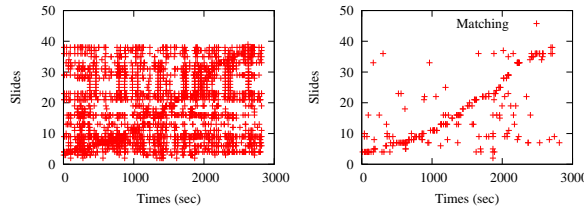


Fig. 2. Matching values before (left) and after filtering (right)

Slider: The purpose of the slider is the removal of the noise and implausible values. The slider is using a ‘bounded means filter’. Let N be the number of datasets, s_i the slide number of dataset i with $0 < i < N$ and $0 < s_i \leq S$, with S being the last slide number in the presentation. For the dataset i we define a foregoing environment of k datasets. For these k datasets of the i -th dataset’s environment we calculate a mean value \bar{x}_i as:

$$\bar{x}_i = \frac{1}{k} \sum_{j=0}^{k-1} s_{i-j}.$$

The mean value \bar{x}_i is compared to the successive mean, the mean of the $i + 1$ -th dataset. If a threshold l satisfies: $\bar{x}_i + l < s_{i+1}$ or $\bar{x}_i - l/2 > s_{i+1}$, the $i + 1$ -th dataset will be discarded and not used for further calculations. The lower threshold is less than the upper one, thus we get a monotone increasing slope. The algorithm depends on two parameters: the threshold l and the environment k . Parameter tuning was performed on a corpus. The best performance was achieved with $k=7$ values and a threshold of $l=5$ slides. The algorithm is implemented recursively: if there exists no match within the given threshold, the algorithm is recursively called with an increase of l .

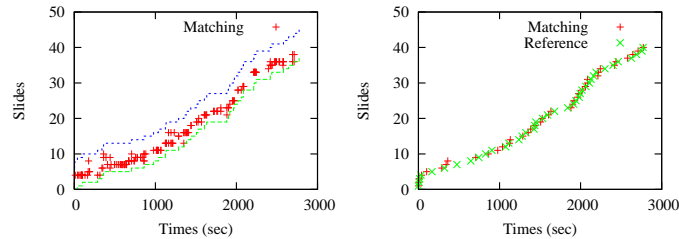


Fig. 3. Values after slider filtering (left), detected slide transitions (right)

Figure 3 illustrates the dataset after the use of the noise removal and the both threshold boundaries. Next, the slide steps of the datasets have to be determined. To do this the calculated mean values are used again. Every alteration of the means represents one step. Only one step is allowed per alteration. Thus, the slide transitions can be reproduced. Figure 3 shows the slide starting points and the manually determined reference values.

4 Evaluation

For evaluation we were using the recorded lectures series “Technische Grundlagen des WWW” and “Internet Security - Weaknesses and Targets” of Christoph Meinel from Hasso-Plattner-Institute in Potsdam (Germany) and the recorded lecture series “Semantic Web” of Harald Sack, Friedrich-Schiller-University of Jena (Germany). Our dataset includes two different speakers and two different languages (German and English). Table 1 shows a summary of the dataset’s content.

4.1 Measurement

Mean of Error Rates: The error rates are calculated as difference between the point in time of the reference and the point in time achieved by our algorithm for each slide transition in the course. The meaning of these values is the medial of all differences (i.e. the offset of the time-shift). A positive offset is a positive time shift and a negative offset is a

Table 1. Summary of the lecture series archive

Name	Speaker	Language	Accuracy	Lectures	Words	Duration	Transitions
WWW	Meinel	German	20-70%	24	258190	31h	1069
Semantic	Sack	German	20-55%	10	100437	12h	485
Security	Meinel	English	20-55%	14	130320	15h	448

negative time shift. For example $\bar{X} = -100s$ the starting point (or the constant component of the data-set) is 100 seconds after the real start-time of the slide. It is certain that a low offset (the mean \bar{X}) stands for higher accuracy of the algorithm. Addinially, we calculate the mean \bar{Y} of all absolute differences. The means of difference are calculated in the following way:

$$\bar{X} = \frac{1}{N} \sum_{i=1}^N (r_i - c_i) \text{ and } \bar{Y} = \frac{1}{N} \sum_{i=1}^N ABS(r_i - c_i)$$

with r_i denoting reference time and c_i calculated time of dataset i .

Standard Deviation: The standard deviation points out how much the data are spread out from the mean \bar{X} . If the datasets are close to the mean, the standard deviation is small and the algorithm matches the boundaries very well. Conversely, if many data points are far-off the mean, the standard deviation is large and the algorithm has higher error rates. Higher deviation of values are downgrading the search process. The users are not ready to accept overflowing dissimilarities of the starting time in the search results.

Precision and Recall: For our setting, precision and recall are not adequate for evaluation. The number of slides is well-known and any implemented algorithm would generate a hit for each slide. Thus, the recall would be always 100%. Another argument is the inconvenience of deciding, if a calculated time is a hit or not. This depends on the size of the delay between reference time and calculated time. The acceptable size of delay varies for every video recording and cannot be assigned in an objective way.

WindowDiff: The WindowDiff [15] measurement is used as a standard evaluation measure for text segmentation. WindowDiff does not take into account the time delay between the calculated time and the real boundary time. Even if a promising WindowDiff value is measured, a high discrepancy between the time of the real boundary and the calculated boundary can still exist. Figure 4 is showing texts with ten sentences (boxes) each. Each text box shows a calculated (calc) and a reference (ref) boundary. The calculated boundary in the first example is after the second sentence and in the second example after the fourth sentence. The x-axis represents the time, whereas the time in the first example has higher resolution than in the second example. The WindowDiff value of the second example is better than in the first example, but the time

delay in the second example is 10 seconds compared to 2.5 seconds in the first example. But, for video retrieval it is preferred to match the exact position in time, than to have an exact text segmentation position. Thus, WindowDiff is not a good choice for video segmentations measurement.

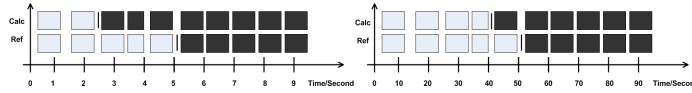


Fig. 4. WindowDiff Example 1. (left) and WindowDiff Example 2.(right)

4.2 Comparison of Algorithms

Our algorithm was evaluated in countercurrent with two kinds of algorithms:

First, we have compared it with standard linear text segmentation algorithms. The slide transition number is given and a sentence length of 10 words is defined. A slide transition is assumed to be a text boundary and can be detected by the C99 and the LCseg algorithm [3, 5]. In this way the number of segments has been provided to the algorithms. A stop-word list and a stemmer both for the German language have been adapted to C99 and LCseg. Additionally, a linear distribution (Linear) of the slide transition during the presentation time has been implemented. Second, a vector algorithm similar to [17, 8] has been implemented. A window slides over the stream of words and each window is matched with the word stems of each slide. The window size has been set to 140 words and the step to 20 words. After standard pre-processing (deletion of stop-words, word stemming, and matrix ranking) the similarity values (cosine measure) between each window and each slide have been calculated. A detected slide transition maps to the highest similarity between a window and the slide.

4.3 Result

Table 2 shows the mean over the three test corpora and the quantitative results are summarized in final column. The table shows that our RW07 algorithm yields to significantly lower mean and standard deviation than C99, LCseg, Linear and the Vector algorithm for the test corpora. It is not clear, whether the RW07 performs better than the other algorithms in the text segmenting measure WindowDiff. The difference is too low and it is not significant for any statement.

The vector algorithm matches a lot of slides correct. A disadvantage of this algorithm is the large amount of outlier hits (see figure 5). This mismatch results in the loss of the chronological sequence. Only one outlier can produce a large mean and a large standard deviation. The vector algorithm and the increasing slope algorithm (Slider) have been combined to avoid this interference in the result set.

Table 2. Mean, standard deviation, and WindowDiff of experimental results.

mean/s \bar{X} (\bar{Y})					
Course	LCseg/s	C99/s	Linear/s	Vector/s	RW07/s
WWW	-31 (674)	6 (215)	85 (204)	28 (311)	27 (52)
Semantic	-63 (742)	184 (321)	179 (293)	205 (663)	5 (89)
Security	-306 (577)	28 (189)	3 (164)	209 (639)	9 (113)
Summary	-100 (669)	54 (235)	89 (217)	111 (470)	18 (74)
Standard Deviation					
WWW	924	294	258	837	84
Semantic	1050	359	318	1393	134
Security	850	252	217	1211	163
Summary	947	311	272	1087	119
WindowDiff					
WWW	0.54	0.47	0.53	0.49	0.46
Semantic	0.54	0.50	0.52	0.52	0.54
Security	0.49	0.52	0.51	0.53	0.52
Summary	0.52	0.49	0.52	0.51	0.49

Therefore, the vector algorithm has been modified in the way that the tenth most similar value between a slide and the text windows have been used. The resulting tuple (slide-number, time) of a lecture serves as the input for the Slider. With this improvement an enhancement in mean

Table 3. Vector algorithm combined with Slider.

	mean/s \bar{X} (\bar{Y})	Standard Deviation/s	WindowDiff
RW07	18 (74)	119	0.49
Vector	111 (470)	1087	0.51
Vector + Slider	-1 (104)	224	0.53

and in standard deviation has been achieved.

Compared with RW07 the mean is better, but the more important standard deviation becomes worse. The deviation is approximately 3.5 min, compared to 2 min of the RW07.

5 Conclusion

We have presented a speaker and language independent segmentation and annotation algorithm for synchronization of lecture video recordings

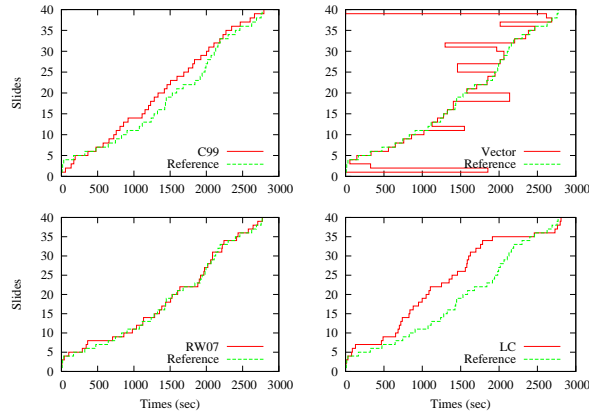


Fig. 5. Overview and error.

and text-based presentations. Keeping in mind that the average duration of a lecture video is about 45 to 90 minutes, a lecturer is speaking about a single slide approximately for only 105 seconds. If it is possible to match the correct slide in the video with an offset of 18 seconds and with a deviation of ± 119 seconds, then we will find a fuzziness area of ± 1 slide. For lectures, usually the topics of surrounding slides are very similar and therefore, the fuzziness area has no dramatic effect for the user. It is a significant improvement in accessing the content of video recordings.

The linear text based segmentation approaches (C99, LCseg) are not suitable for the application to erroneous transliteration of lecture video recordings. The vector algorithm is most suitable for a none sequential workflow, where a presentation slide might match to any given point in time of a lecture video recording. But, most lectures follow a sequential order of presentation slides.

The descriptors gained from the presentation slides can be used for MPEG-7 annotations. The application for our algorithm is not only limited to the use in sequentially ordered presentations. All applications with a chronology activity, as e. g. newscasts, theater-plays, or any kind of speech being complemented by textual data could be analysed and annotated with the help of our proposed algorithm. For further refinement, we plan to investigate the effects of adding features such as feature-words to our segmentation and annotation algorithm to achieve even more accurate results.

References

1. D. Beferman, A. Berger, and J. D. Lafferty. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210, 1999.
2. Y. Chen and W. J. Heng. Automatic synchronization of speech transcript and slides in presentation. In *Proceedings of the IEEE*

- International Symposium on Circuits and Systems (ISCAS)*, pages 568–571. Circuits and Systems Society, May 2003.
3. F. Y. Y. Choi. Advances in domain independent linear text segmentation. In *Proceedings of NAACL-00*, 2000.
 4. W.-T. Chu and H.-Y. Chen. Cross-media correlation: a case study of navigated hypermedia documents. In *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*, pages 57–66, New York, NY, USA, 2002. ACM Press.
 5. M. Galley, K. McKeown, E. Fosler-Lussier, and H. Jing. Discourse segmentation of multi-party conversation. In *ACL*, pages 562–569, 2003.
 6. R. Gross, M. Bett, H. Yu, X. Zhu, Y. Pan, J. Yang, and A. Waibel. Towards a multimodal meeting record. In *IEEE International Conference on Multimedia and Expo (III)*, pages 1593–1596, 2000.
 7. A. Haubold and J. R. Kender. Augmented segmentation and visualization for presentation videos. In *ACM Multimedia*, pages 51–60, 2005.
 8. M. A. Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, 1997.
 9. P. Hsueh and J. Moore. Automatic topic segmentation and labelling in multiparty dialogue. In *First IEEE/ACM workshop on Spoken Language Technology (SLT), Aruba*. IEEE Computer Society, 2006.
 10. W. Hürst, T. Kreuzer, and M. Wiesenhütter. A qualitative study towards using large vocabulary automatic speech recognition to index recorded presentations for search and access over the web. In *IADIS International Conference WWW/Internet (ICWI)*, pages 135–143, 2002.
 11. M. Li, B. Ma, and L. Wang. Finding similar regions in many sequences. *J. Comput. Syst. Sci.*, 65(1):73–96, 2002.
 12. G. Myers. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *J. ACM*, 46(3):395–415, 1999.
 13. H. Ney and S. Ortmanns. Progress in dynamic programming search for lvcsr. In *Proceedings of the IEEE*, 88(8):1224–1240, August 2000.
 14. C.-W. Ngo, F. Wang, and T.-C. Pong. Structuring lecture videos for distance learning applications. In *Proceedings of the Multimedia Software Engineering*, pages 215–222. ISMSE, December 2003.
 15. L. Pevzner and M. A. Hearst. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36, 2002.
 16. S. Repp and C. Meinel. Segmenting of recorded lecture videos - the algorithm voiceseg. In *Proceedings of the 1th Signal Processing and Multimedia Applications*, pages 317–322. ICETE, August 2006.
 17. N. Yamamoto, J. Ogata, and Y. Ariki. Topic segmentation and retrieval system for lecture videos based on spontaneous speech recognition. In *Proceedings of the 8th European Conference on Speech Communication and Technology*, pages 961–964. EUROSPEECH, September 2003.