Lecture Video Indexing and Analysis Using Video OCR Technology

Haojin Yang, Harald Sack, Christoph Meinel

Hasso Plattner Institute (HPI), University of Potsdam
P.O. Box 900460
D-14440 Potsdam
{haojin.yang, harald.sack, meinel}@hpi.uni-potsdam.de

ABSTRACT: *Texts displayed in lecture videos are closely related to the lecture content. Therefore, they provide a valuable source for indexing and retrieving lecture videos. Textual content can be detected, extracted and analyzed automatically by video OCR (*Optical Character Recognition*) techniques. In this paper, we present an approach for automated lecture video indexing based on video OCR technology: first, we developed a novel video segmenter for the structure analysis of slide videos. Having adopted a localization and verification scheme, we perform text detection secondly. We apply SWT (*Stroke Width Transform*) not only to remove false alarms from the text detection stage, but also to analyze the slide structure further. To recognize texts, a multi-hypotheses framework is applied, that consists of multiple text binarization, OCR, spell checking and result merging processes. Finally, we have implemented a novel algorithm for extracting lecture structure from the OCR-transcript by using geometrical information and text stroke width of detected text lines. We use both segmented key frames and extracted lecture outlines for the further video indexing. The accuracy of the proposed approach is proven by evaluation.*

Keywords: Lecture Videos, Multimedia Indexing, Automated Multimedia Analysis

1 Introduction

Since recording technologies have become increasingly robust and easier to use in the last few years, more and more universities are taking the opportunity to record their lectures and publish them online in order to make them accessible for students. Hence, many lecturers in professional colleges use *Slide* presentations instead of blackboard writing today. This way of recording lectures leads to large amounts of multimedia data very quickly. Thus, finding lecture video data on the WWW (*World Wide Web*) or within lecture video portals has become a very important and challenging task.

We focus our research on such lecture recordings that combining two video streams: the main scene of lecturers which is recorded by using a video camera, and the second which captures the frames projected onto the screen during the lecture through a frame grabber (cf. Fig. 1).

The frame grabber output (slide stream) can be synchronized with the video camera automatically during the recording; Therefore, each key frame of slide videos can be linked to a video recording segment. In this way, indexing two-part lecture videos can be performed by indexing the slide videos only.

Content-based gathering within video data requires textual metadata that has to be provided manually by the users or that has to be extracted by automated analysis. For this purpose, techniques from common OCR—focusing on high-resolution scans of printed (text) documents—have to be improved and adapted to be also applicable for video OCR. In video OCR, video frames containing visible textual information have to be identified first. Then, the text has to be separated from its background, and geometrical transformations have to be applied before common OCR algorithms can process the text successfully.

In this paper, we propose an entire workflow for the structural segmentation of lecture videos, the video OCR analysis, and the slide structure extraction, as illustrated in Fig. 2.

Figure 1: An example lecture video. Video 2 shows the speaker giving his lecture, whereas his presentation is played in video 1.

We developed a slide video segmenter based on fast CC (*Connected Component*) analysis, and slide transition recognition. However, this segmenter is not suitable for video frames which are embedded in lecture slides with a different genre. Thus, we apply image intensity histogram feature and SVM (*Support Vector Machine*) classifier to distinguish slide frames and other video frames. For the latter case, we apply the commonly used video segmentation method based on the histogram differencing metric.
For text detection, we realized a localization and verification scheme: an edge-based text detector serves to achieve a high recall rate with low computational time expenses. The verification process consists of a SWT (*Stroke Width Transform*)-based analysis to remove false alarms.
For text recognition, we employ a multi-hypotheses framework consisting of multiple text segments, common OCR analysis, spell checking, and result merging processes.
We propose a novel slide structure analysis by using the detected text objects—we use the geometrical information and the corresponding stroke width value of those text objects for slide title identification, text line classification, and lecture outline extraction.
We have further integrated the video analysis engines into our lecture video portal. Therefore, the analysis process can be managed easily.
Operability and accuracy of proposed methods have been evaluated by using publicly available test data sets.
The rest of this paper is organized as follows: Section 2 presents related work, whereas the sections 3–8 describe our proposed framework in detail. We provide the evaluation and experimental results in section 9. Section 10 concludes the paper with a brief outlook on future work.

2 Related Work

Adcock et al. proposed a lecture webcast search system [1], in which they applied a slide frame segmenter to identify the distinct slide images. The OCR and lexical filtering processes are utilized, which in turn generate the indexable texts. Since they do not apply text detection and text segmentation processes on video frames, the character recognition accuracy of their approach is therefore lower than our system's. Moreover, by applying text detection process, we are able to extract the structured slide text lines, as e.g., title, subtitle, key-point etc. for the indexing task, and build a more flexible search method.
Wang et al. proposed an approach for lecture video structuring and textual analysis [6]. Their shot boundary detector computes the changes of text and background regions using thresholds to capture the slide transition. The final video structuring results are determined by synchronizing detected slide shots and external documents based on OCR texts.
Hunter et al. proposed a SMIL (*Synchronized Multimedia Integration Language*) based framework to build a digital archive for multimedia presentations [9]. Similar to [6], they also apply a synchronization process between recorded lecture video and the PDF slide file that has to be provided by presenters. In contrast to these two approaches, our system directly analyzes the videos, we thus do not need to take care about the slide format and the synchronization with an external document is not required. Since the animated content buildup of the slide has not been considered in [6] and [9], their systems might not work robustly when these effects occur in the slide videos. Furthermore, in [6],

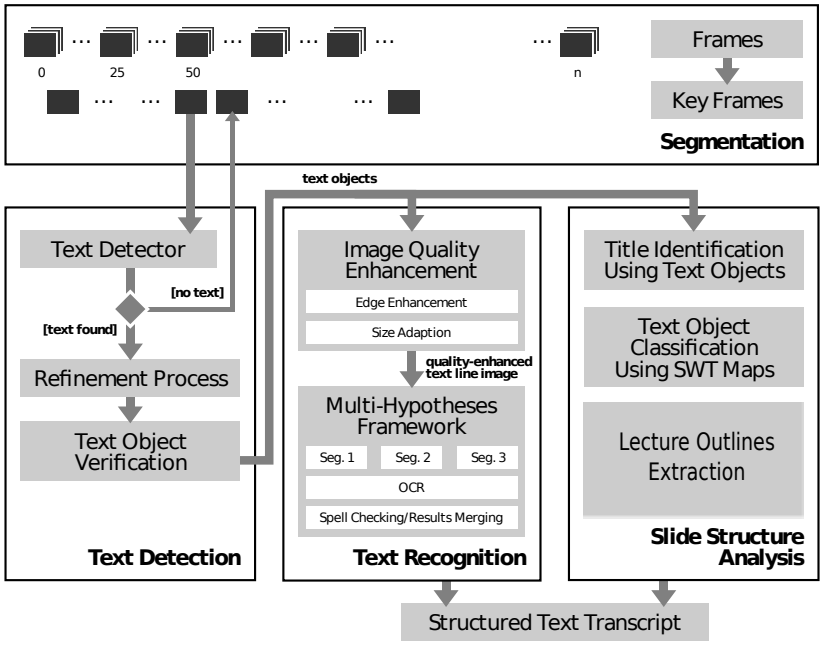


Figure 2: The entire system workflow. After having segmented frame sequences, text detection is performed on every single key frame in order to find text occurences. The resulting text objects are then used for text recognition and slide structure analysis.

the final structuring results are heavily dependent on the OCR results. It might be less efficient concerning computation time and implies redundancies, when poor OCR accuracy is obtained.

In the last few years, collaborative tagging has become a popular functionality in video portals. Some approaches, as e.g., Kim [10], Sack et al. [15] and Moritz et al. [5] apply manual video tagging to generate text-based metadata for efficient context-based video retrieval. The user tagging-based approaches work well for the recently recorded lecture videos. However, for videos which were created more than 10 years ago, there is no manually generated metadata available. And it is hard to attract the user's interest to annotate the outdated videos. Furthermore, regarding the quantitative aspect, the automated analysis is more suitable than the user annotation for large amounts of video data.

ASR (*Automated Speech Recognition*) is able to provide text transcripts of spoken language. Some approaches, as e.g., [20], [8] and [14] use the ASR output for lecture video retrieval. But often poor recording quality, speaker's dialects, or interfering noise lowers the quality of the achieved ASR results beyond further usefulness.

Text detection is one of the most important processing step of a video OCR framework. Most of proposed text detection methods make use of texture, edge, color, motion and some other text representative features (as e.g., stroke width feature) to discriminate text from background.

Texture features such as DCT (*Discrete Cosine Transform*) coefficients of grayscale images have been applied for text detection [13, 21]. DCT feature based approaches are well suited for JPEG encoded images as well as for MPEG encoded videos. However, in practice this feature is not robust enough to distinguish text from text similar textured objects such as, e.g., bushes, or other vegetation. For this reason DCT-based approaches are not suitable to handle text detection tasks in complex natural scenes.

Zhao et al. [22] introduced a classification approach for text detection in images using sparse representation with discriminative dictionaries. They applied wavelet transform to create an edge map and performed a block-wise classification to distinguish text and non-text regions by using two learned discriminative dictionaries. The obtained candidate text regions were further refined by projection profile analysis. However, the traditional machine learning-based verification can only deliver a binary decision and thus might not be robust enough to decide for the candidate bounding boxes that contain both text as well as non-text objects. Anthimopoulos et al. [2] have improved this drawback with a refinement process, which enables the machine learning classifier to refine the boundaries of the text images. This process enhanced the precision and the overall detection performance as well. But the evaluation results of the processing

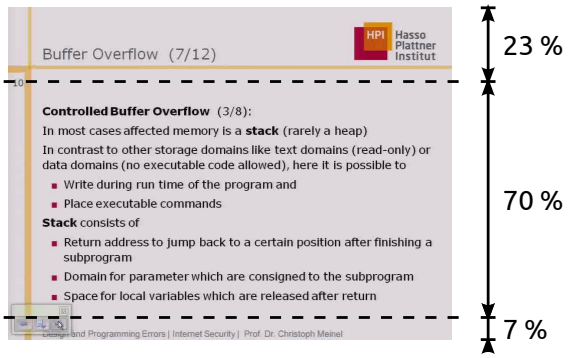


Figure 3: Slide content distribution. Title and content act as indicators for actual slide transitions and are thus analyzed in the second step.

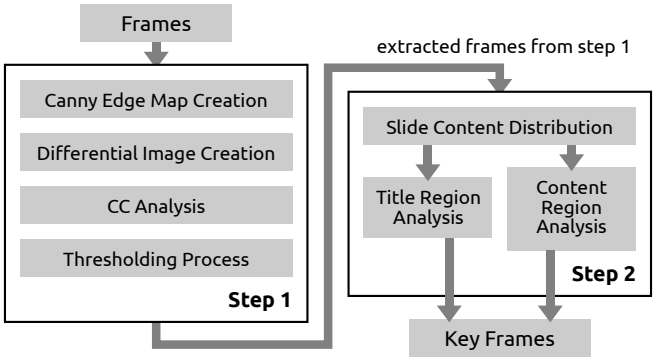


Figure 4: Segmentation workflow. (Step 1) Adjacent frames are compared with each other by applying *Connected Components Analysis* on their differential edge maps. (Step 2) Title and content region analysis ensure that only actual slide transitions are captured.

time reported in their work shows that the system was 6 times slower by applying this refinement algorithm.
Epshtein et al. propose a text detection method using SWT for nature scene images [4]. For each pixel, their algorithm computes the width of the most probable stroke containing the pixel. The output of the operator is a feature map having the same size as the input image while each pixel represents the corresponding stroke width value. We have extended the method in order to make it more efficient and suitable for our approach (cf. section 4).

3 Video Segmentation

By decomposing the video into a set of representative key frames, we are able to index the video. These selected frames will be processed in other steps later, since they capture and encapsulate the entire video content. In the next subsection, we will discuss our slide-video segmentation method.

3.1 Slide Video Segmentation

Regarding the characteristics of lecture videos, we can see that the retrieval of video contents can be achieved through the extraction of text information from slide videos: Fig. 1 shows an example lecture video that consists of two video streams showing the speaker and the current slide respectively. In such kind of videos, each part of the video can be associated to a corresponding slide. Hence, we carry out the analysis of the slide video stream in our study.
The segmentation process can be achieved by using frame differencing metrics which take both image structure and color distribution into account. The global pixel difference metric has commonly been used for determining slide segments. One deficiency of this method is that the high-frequency image noise within video frames can still degrade the segmentation accuracy.

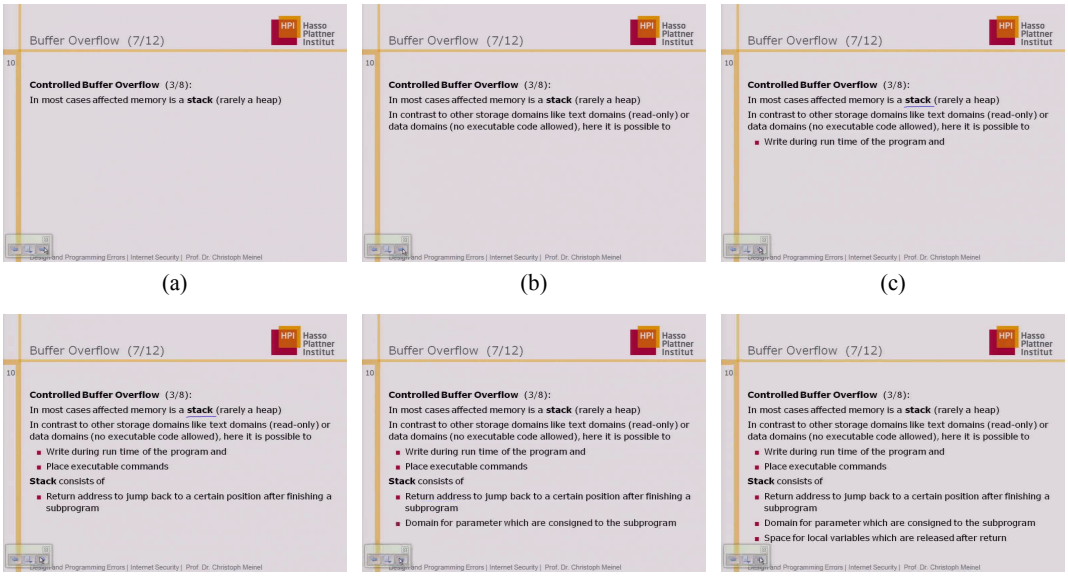


Figure 5. Segmentation results from the first step

Observing the slide video stream, we can see that video contents such as texts, figures, tables etc. can all be considered as a collection of CC (*Connected Component*). Therefore, the difference between two frames can be determined by calculating the difference of the amount of CCs. In this way, we are able to control the valid size of CCs so that the image noise can be removed from the comparison. We have investigated several CC algorithms and finally chose the most efficient one according to [3].

Our segmentation algorithm consists of two steps (cf. Fig. 4): in the first step, the entire slide video is analyzed. For reasons of efficiency, we do not perform the analysis on every video frame; instead, we established a time interval of one second. Therefore, our video segmenter considers only one frame per second (fps).

Subsequently, we create the pixel-based differential image from canny edge maps of two adjacent frames and perform the connected component analysis on the differential image. In this way, the number of differencing CCs are captured. In order to ensure that each newly emerging knowledge point or newly added figure within a slide can be detected, we have identified the segmentation threshold value T_{s1}. This means that a new segment is captured if the number of CCs from the differential edge image exceeds T_{s1}. Fig. 6 depicts the workflow of CC-based image differencing metric. The result of the first segmentation step is too redundant for indexing, since there are many changes within the

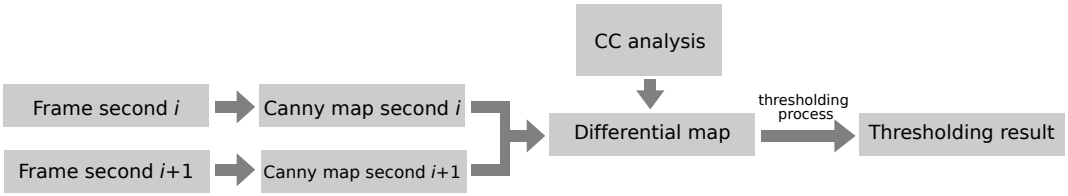


Figure 6. CC-based image differencing metric

same slide (cf. Fig. 5). Hence, the segmentation process continues with the second step that aims to find the actual slide page transition based on the frames detected in the first step.

After a statistical analysis of large amounts of slide videos, we defined the content distribution of the most commonly used slide style (cf. Fig. 3). Let R_t and R_c denote the title and the content regions in Fig. 3 which account for 23 % and 70 % of the entire frame height respectively. Then, we apply the same CC-based differencing metric for R_t. If the amount of CCs in the differential edge image for R_t exceeds the threshold value 1, a slide page transition is captured. This is because that any changes within the title region may result the slide transition. For instance, two slide titles

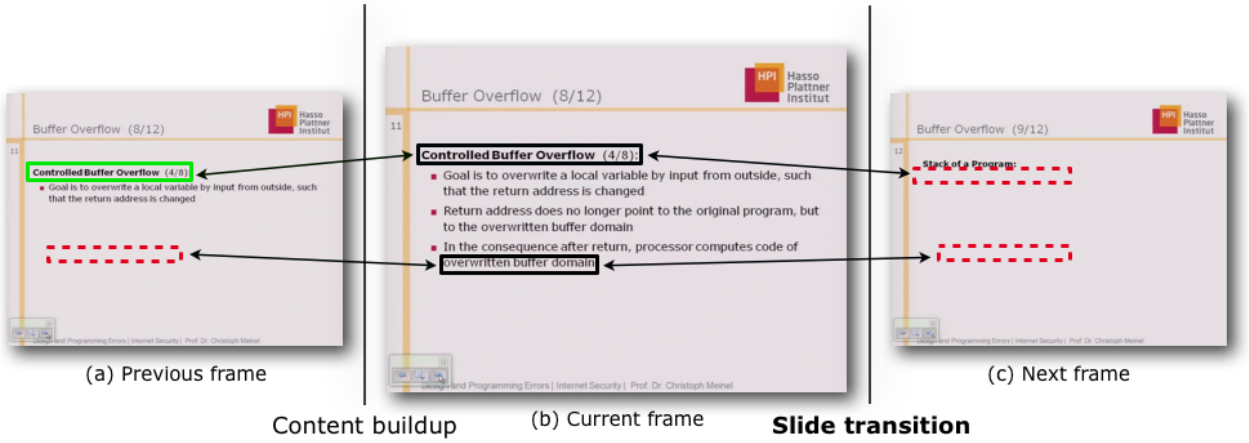


Figure 7: We detect the first text line top-down and bottom-up in R_c of current video frame and perform the CC differencing analysis on those text line sequences from adjacent frames. In frame (a) and (b), a same text line (top) can be found; Whereas in frame (b) and (c), the CC differences of both text lines will exceed the T_{s2}. A slide transition is therefore captured between frame (b) and (c).

might only differ from each other in a single chapter number. If no change from the title regions can be detected, we then detect the first and the last text lines in R_c respectively, and perform the corresponding CC-based differencing metric on these text line regions from two adjacent frames (cf. Fig. 7). If both of the two difference values exceed a threshold T_{s2}, a slide page transition is also captured. In general, the T_{s1} is defined to measure the CC-based differencing of the whole slide frame and the T_{s2} is defined for the single text line. In our study, T_{s1} = 20 and T_{s2} = 5 have proven to serve best for our training data, respectively.
Since our slide segmentation method is not suitable for videos with varying genres that were embedded in slides or are played during the presentation, we have improved the method with a SVM classifier. This will be presented in the next section.

3.2 Video Frame Classification

Regarding slide images, we can see that their content consists of texts, tables, figures etc., and comes with homogeneous background, which strongly differs to other video genres. In order to train an efficient classifier we have evaluated two features:

Histogram of Oriented Gradient (HOG) feature has been widely used in object genre classification domain [11]. To create HOG feature, first, the gradient vector of each image pixel within a local region is calculated. A histogram for all gradient vectors with *n* directions is subsequently created. In order to avoid sensitivity of HOG feature to illumination, the feature values are often normalized [11]. Hence, the distribution of gradients of slide images could be distinguishable from other video genres.

Image Intensity Histogram is a simple feature that describes the image complexity and grayscale distribution. We have also used this feature to train the SVM classifier comparing to HOG feature.

We have applied the commonly used segmentation method by calculating image histogram difference to extract key frames for non-slide video images. The evaluation results of two features are provided in section 9.
Fig. 8 shows the extracted video slides in the video player for navigation within lecture videos in our lecture video portal [18]. In order to give the user a visual guideline for the navigation, the slides are visualized in the form of a time bar. When sliding a mouse over the small squares, they are scaled up to a larger preview of the slide. By clicking the preview image the player will jump to the corresponding position in the video.

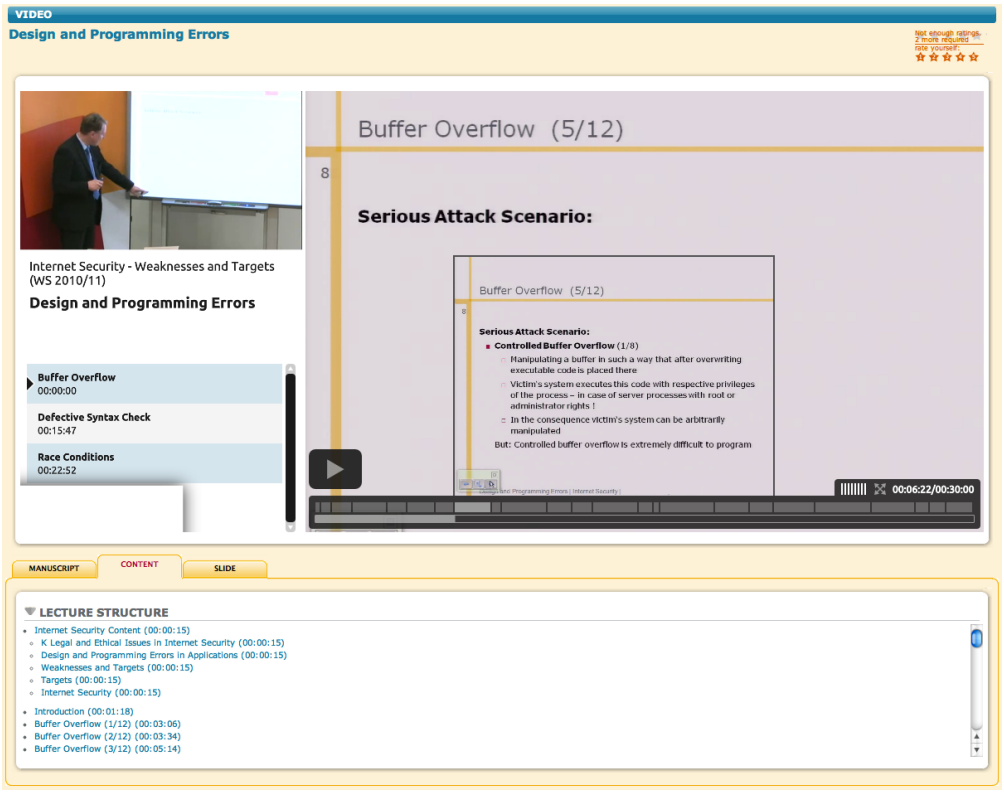


Figure 8: Visualisation of the segmented lecture slides and extracted lecture outlines (lecture structure) underneath the video player

4 Text Detection

Text detection is the first task performed for video OCR. Our approach determines whether a single frame of a video file contains text lines, for which it returns a tight bounding box eventually. For reasons of efficiency, text detection is only executed on the key frames we obtained during the video segmentation process.
In order to manage detected text lines efficiently, we have defined a class *text object* with the following properties: bounding box location (the top-left corner position) and size, start time, end time, text content, and average stroke width. If a text line has successfully been detected in a frame, a text object is created. The occurrence duration of each text object is determined by reading the time information of the corresponding segment. After the first round of text detection, a text object verification procedure ensures the validity of our results in order to reduce false alarms. The output of this text detection system is an XML-encoded list serving as input for the text recognition and slide structure anlysis that we discuss in section 5 and 6.

4.1 Text Localization

To localize text, we firstly produce a horizontal and a vertical edge map for an input image using the Sobel filter [17]. Then, we adopt two morphological dilation operators to extend the vertical edges horizontally and the horizontal edges vertically. Let $MinW$ and $MinH$ denote the detected minimal text line width and minimal character height. We define two rectangle kernels:$1 \times MinW$ and $6 \times \frac{MinH}{2}$ for vertical and horizontal dilation operators respectively. Since video texts with a height smaller than 5 pixels are hard to read for both, human and OCR engines, we assume that the minimal height $MinH$ of the recognizable text is 5 pixels for the original scale of the image. Furthermore, each detected text line has to contain at least 2 characters. Thus, $MinW$ is calculated by $MinW = 2 \times MinH$.
Subsequently, we apply Otsu thresholding to binarize the dilation maps [12]. Due to the connectivity of horizontal and vertical character edges in text-like regions, a new image is created by covering the intersected regions of both binary

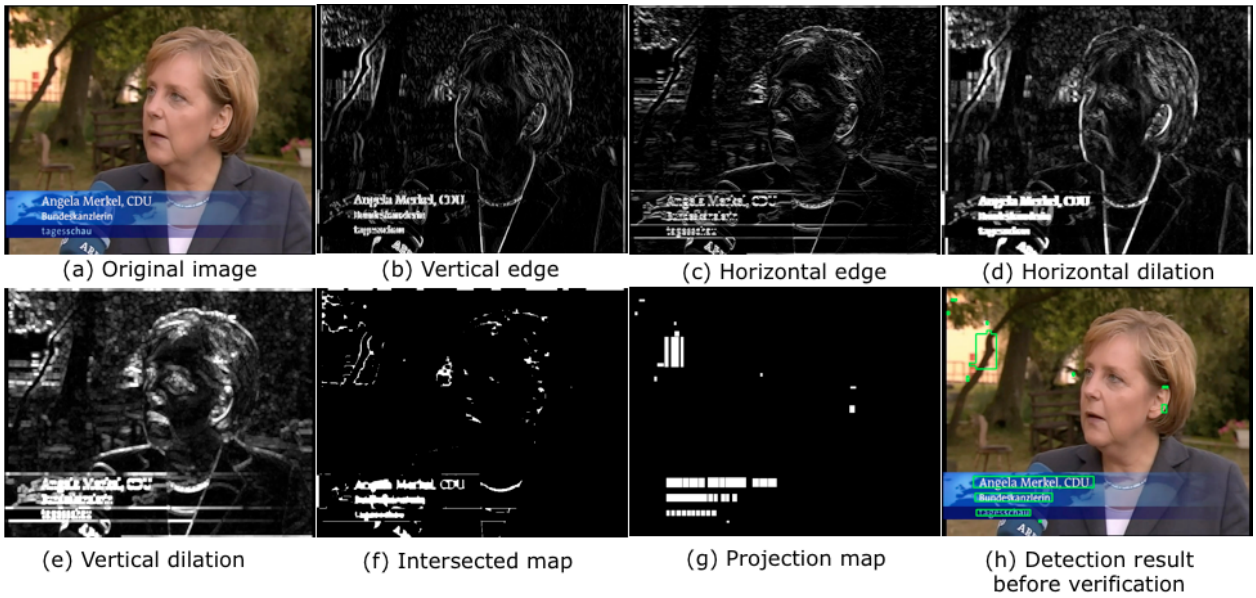


Figure 9: Workflow of the edge based text detection. (b) (c) Vertical and horizontal edge maps. (d) (e) Dilation maps of corresponding edge images. (f) Intersected image of binarized dilation maps. (g) After projection profiles. (h) Detection results without verification.

maps. Ultimately, the initial bounding boxes are generated after performing CC analysis (cf. Fig. 9 (a)-(f)).

4.2 Text Region Refinement

Although the text detection process from the previous step has a very high detection rate, the initial bounding boxes are still not accurate enough for OCR engines and also some false alarms may be produced. To reject such falsely detected regions, we apply a refinement procedure based on projection profiles.

We firstly process the horizontal projection on the edge maps of input images. A horizontal line is discarded when its projection value is lower than a given threshold; in this way, the multi-line bounding boxes are segmented to single lines. Subsequently, we relocate the bounding boxes. Then, we perform the vertical projection in a similar way; yet it might split text lines into single characters. Thus, we have to merge neighboring characters—having a neighbor distance less than the $MinH$ of this text line—horizontally together. We execute this refinement process iteratively until the bounding box list remains unchanged (cf. Fig. 9 (g)-(h)).

Due to the refinement process, we obtain more precise bounding boxes, and false regions which have a low edge density have been rejected successfully. However, there may still remain few false objects, such as human faces, foliage, barriers etc., which are often confused with text patterns. In order to remove them, the process continues with a text verification procedure.

4.3 Text Verification

We apply a verification method based on SWT analysis. Epshtein et al. [4] have proven in their work that the stroke width feature is robust enough to distinguish text from other complicated image elements such as vegetation. Per pixel, the algorithm computes the width of the most likely stroke containing this pixel. The output of SWT is a feature map where each pixel contains the potential stroke width value of input image pixels (cf. Fig. 10).

In order to use the method for our approach, we have extended it in following manners: first, Epshtein et al. describe that the SWT analysis depends on the edge gradient direction [4]. In oder to accommodate both bright text on dark background and vice-versa, the analysis process has to be applied twice to cover each gradient direction, which leads to a loss of performance. In general, it is difficult to determine the correct gradient direction for the whole image. However, our two-stage analysis approach solves this issue: In the first stage, we have detected the single-line text

bounding boxes already. Subsequently, a corresponding subimage is created for each bounding box. We then only apply SWT on the subimages.
Hence, it is relatively easy to determine the gradient direction for subimages of slide frames: first of all, we convert the subimage to a grayscale map and binarize it subsequently using Otsu thresholding method. Then, we could obtain the text gradient direction by calculating the ratio of black and white pixels, where we assume that the background pixel number is more than the text pixel number.
We use the following constrains to verify the bounding boxes. A box is discarded if:

- its stroke width variance exceeds a threshold range $MinVar$ and $MaxVar$,
- its mean stroke width exceeds a threshold range $MinStroke$ and $MaxStroke$.

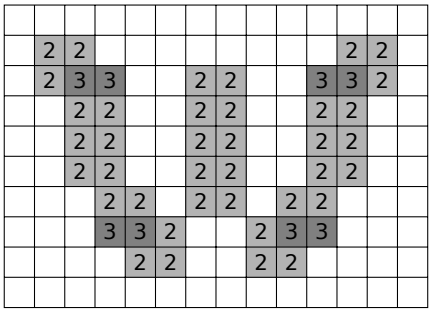


Figure 10. An exemplary SWT image for character w.

For some boxes with a size larger than $MaxH$—which may contain both text lines and other image objects—we apply the detection algorithm proposed by Epshtein et al. [4] on them. In this way, the detection recall can in turn be increased during the verification process. In our study, $MinVar$ and $MaxVar$ have been set to 100 and 500, while $MinStroke$ and $MaxStroke$ have been set to 1 and 10, respectively. These values were also learned from our training data. Fig. 11 shows the results after verification which are the final output of the text detection.

Figure 11: Refined bounding boxes from text detection. The results are automatically verified in order to remove false alarms.

Figure 12: Exemplary resulting images of dynamic contrast/brightness adaption. Contrary to the originals, the revised images are suitable for being processed by OCR engines, as the background has been removed.

5 Text Recognition

The detected text objects from the previous step serve as the input for the text recognition. Usually, texts in video frames have poor resolution, and often, the image background is of heterogeneous complexity. However, common OCR engines are designed for high-resolution scans of printed texts having a homogeneous plain background.
To enable an efficient way of processing the detected texts, the text images have to be further refined in an image quality enhancement process. To resolve the problem of having low-resolution images, we enlarge text line images with fonts smaller than 40 pixels text height using cubic interpolation. Then, the texts will be segmented and recognized by a subsequent multi-hypotheses framework.

5.1 Text Segmentation

The global thresholding methods use a single threshold for the entire document and local thresholding methods which assign a threshold for each small local region of the processed document. We employ three methods in order to segment texts: the Otsu (global) thresholding algorithm, Gaussian-based adaptive (local) thresholding [7] and our adaptive image contrast and brightness adjustment.
By applying the dynamic contrast and brightness adjustment, the grayscale text line images are converted to an image showing black font on almost plain white background or vice-versa. The workflow of the algorithm is depicted in Fig. 14, where m and sd denote the median and standard deviation of the image histogram, with T_s as the threshold value for sd. First, the image contrast is adapted until m equals 255 or 0. Subsequently, sd is adapted until it reaches its threshold T_s. If sd exceeds T_s, the image's brightness is gradually increased—otherwise, its contrast is increased until sd reaches T_s. When the image contrast reaches its maximum while sd is still smaller than T_s, we increase the image brightness until sd reaches its maximum.
In order to achieve the best recognition accuracy, we have optimized the threshold T_s by using our training dataset. The training set consists of video text frames with varying genre and resolutions. The correctly recognized character rate has been applied to indicate the recognition performance. Fig. 13 shows the evaluation results. For our purpose, we chose a fixed value of 120 for the standard deviation threshold T_s.
Fig. 12 shows several result images from our trials. The adapted text line images will be recognized using OCR software.

5.2 Recognition and Spell Analysis

After the segmentation process, we generate three OCR results for each text line image by using a standard OCR engine. Then, we apply spell checking on every OCR string. In order to adapt the spell checker to our data more accurrately, we extend its standard dictionary with technical terms from our working domain. The spell-checking and result-merging processes are based on the following constraints:

- First, a OCR hypothesis will be selected as the final result if it contains the major amount of correct words.
- Second, if two or more hypotheses have the same amount of correct words, the one having the lower word count will be selected.
- If the hypotheses have the same amount of correct words and also the same word count, then all correct words will be merged in the final result according to their indices.

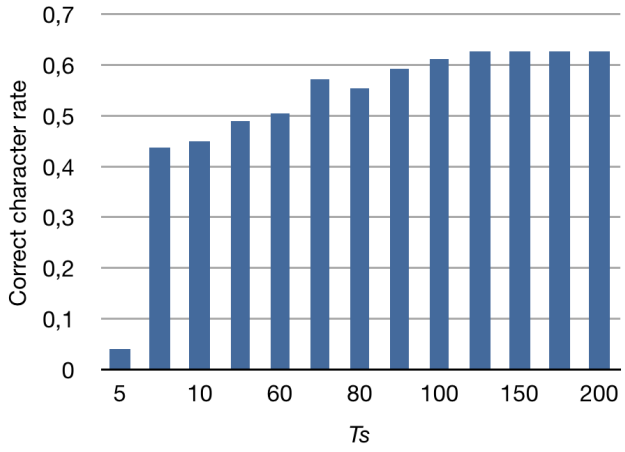


Figure 13. Evaluation results of threshold T_s of the dynamic adaptive contrast and brightness adjustment algorithm

Our experimental results show that the multi-hypotheses framework improved the word recognition rate of about $5\,\%$ in contrast with the using of single segmentation method for our lecture video test set.

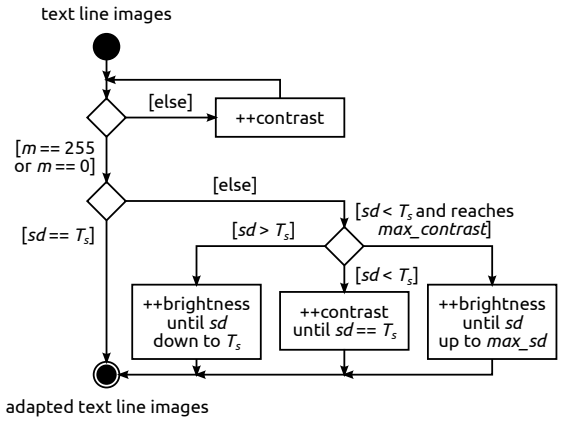


Figure 14: Workflow of the dynamic adaptive contrast and brightness adjustment. After applying the algorithm, the text appears black on a plain white background or vice-versa. m and sd denote the median and standard deviation of the image histogram, T_s is the threshold value for sd.

6 Slide-Content Structure Analysis

Observing the slide frames, we can realize that the title and subtitle texts are more significant than the body of the slide—they summarize each slide, whereas bold texts in the slide body are often conclusions or key points.
On the one hand, a classified (structured) OCR transcript enables a more flexible search algorithm, and on the other hand we are able to extract the lecture structure so that an automatically generated lecture outline can be provided for the students further. A timed lecture outline allows the user in turn to explore the video content on the video portal. In this section, we will discuss our title identification, text object classification, and lecture outline extraction methods .

6.1 Title Identification

The title recognition procedure is intended to identify potential title text lines within the key frames. It is based on the detected text objects' geometrical information. A potential title object is identified if:

- it is in the upper third of the image,

- its width is larger than the minimal word width,
- the x position of its top-left corner is lower than a threshold $MaxX$, which has been set to $frameWidth * 0.77$,
- it is one of the three highest bounding boxes and has the uppermost position,
- the text line has more than three characters.

If a text object satisfies the above conditions, it will be labeled as a title line. The process repeats these steps for the remaining text objects in order to find the next potential title lines; however, title lines within the same frame must have similar height and stroke width. For our purpose, we allow up to three title lines for each slide frame.

6.2 Text Object Classification

After the title identification, the remaining text objects are classified into three classes: *subtitle/key point* (bold fonts), *normal content* and *footline*. The classification is based on the stroke width s_t and the height h_t of text objects. Let s_{mean} and h_{mean} denote the average stroke width and the average height of text objects within a slide frame respectively. The classification is processed as follows:

$$\begin{aligned} \textit{subtitle/key point} \text{ if } & \mathrm{s_t} > \mathrm{s_{mean}} \wedge \mathrm{h_t} > \mathrm{h_{mean}} \\ \textit{footline} \text{ if } & \mathrm{s_t} < \mathrm{s_{mean}} \wedge \mathrm{h_t} < \mathrm{h_{mean}} \wedge \mathrm{y} = \mathrm{y_{max}} \\ \textit{normal content} & \text{ otherwise} \end{aligned}$$

6.3 Lecture Outline Extraction

The lecture outline can be both utilized for manual navigation and to provide indexable metadata for search engine. Our lecture outline extraction algorithm begins with the text line objects which have been obtained from the previous text object classification step that have a text line type of *title* or *subtitle-key point*. We first sort out unsuitable text objects by applying a spell checking procedure. A valid text object must consist out of more than three characters, in which at least one of them must be noun.

After the filtering processes, all title lines within the same slide are merged together by using their coordination positions. The subtitle-key points will be assigned to the corresponding title object subsequently. Slide frames that have a similar title will be merged in order to avoid the redundancy, which might affect the indexing process. The similarity measure is determined by using the word- and text line-edit distance. Finally, a structured lecture outline is extracted according to their time occurrences. Fig. 8 shows an example of extracted lecture outlines underneath the video player. All extracted titles and subtitle-key points with their corresponding timestamps are provided. Timestamps serve as links to jump to the associated position in the lecture video.

7 Automated Analysis Framework for Lecture Video Portal

In the last section we have discussed our video OCR framework. In this section, we will further present how have integrated the automated video analysis framework into the lecture video portal.

7.1 Framework Overview

The major components of the framework are the analysis management engine, video analysis engine, and result storage/visualization engine. The analysis management and result storage/visualization engines are associated with the video portal server, while the video analysis engine is associated with the multimedia analysis server. Fig. 15 shows the framework architecture.

7.2 An Automated Analysis Workflow

The analysis management engine handles the start of the analysis and reports the processing status. As shown in Fig. 16, we can start both, the OCR and ASR analysis for each video in the *staff* page of the lecture video portal. In order to manage each analysis request efficiently, we have defined a class "analysis job object" with the following properties: media id, date time, media URL, analysis type and language. Once the analysis for a lecture video is started, a job object will be created and encoded in the XML format. The XML request will then be sent to the analysis engine on the multimedia analysis server.

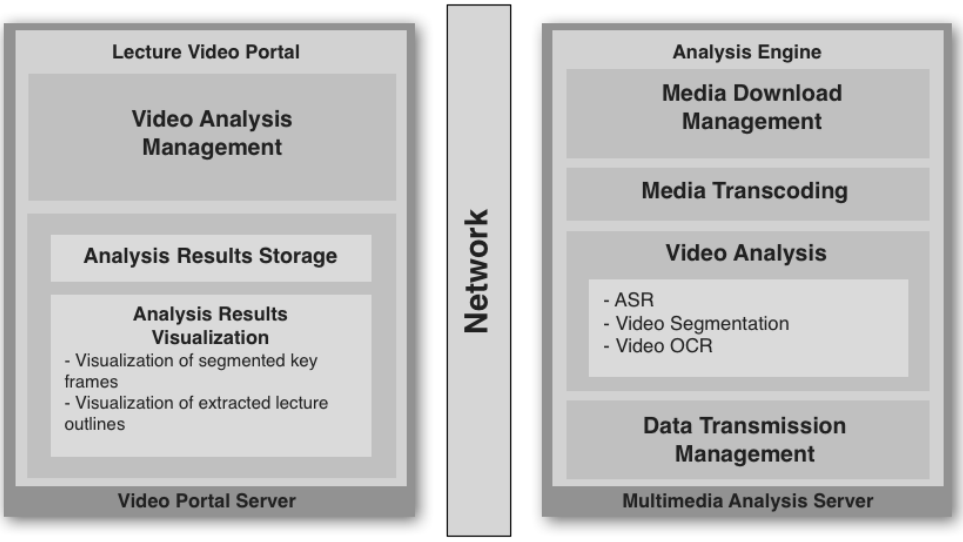


Figure 15. Architecture of the multimedia analysis framework

The analysis engine manages four major processes: media download, video transcoding, video analysis for each request, and analysis result transmission to the video portal server. The analysis engine is designed fully configurable. Furthermore, it is suitable to be extended and work with a multi-process workflow.
Once a video is downloaded, a transcoder will convert it to a predefined format with a suitable visual quality which is most appropriate for the video segmentation and OCR analysis.
We have also applied ASR (*Automated Speech Recognition*) analysis for the lecture video. We extract the audio track from lecture videos with a format of 16KHz and 16 Bit which meets the requirements of the speech recognition software.
After the analysis, the results will be automatically sent to the destination place on the portal server. Subsequently, we send HTTP-POST requests for saving the corresponding results to the portal database. The GUI (*Graphic User Interface*) elements of the slide segments and lecture outlines have been developed based on the plugin architecture in the video portal platform. Therefore, once the analysis result data is saved, the corresponding GUI elements will be created automatically by refreshing the web page.
For our purpose, we apply this framework to analyze the lecture videos from our institute. However, using it for the videos from other sources (e.g., YouTube[1], Berkeley-Webcast[2] etc.), we would only need to adapt the configuration file of the analysis engine.

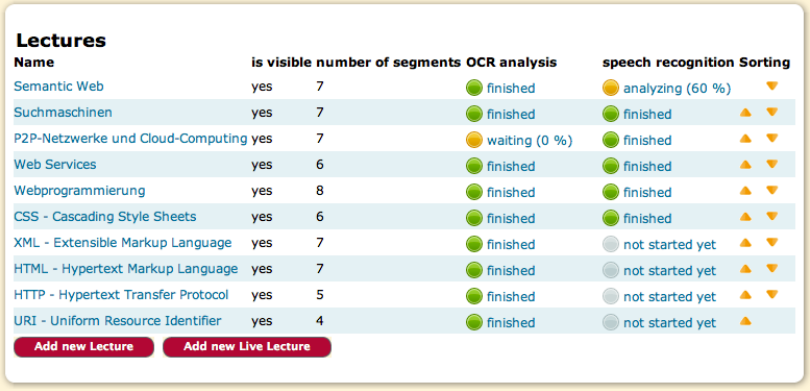
8 Keyword Extraction

Keywords can briefly summarize a document and are widely used for information retrieval in digital libraries. However, manually keyword annotation of multimedia data such as videos, is very time-consuming and expensive when considering costs. Therefore, the development of an automated method for extracting keywords from lecture videos is highly desirable and especially meaningful for a lecture video portal. In this section we will discuss the conception of the keyword extraction workflow in our video analysis framework.

8.1 Keyword Extraction From OCR and ASR Data

In our framework, the keywords are extracted from OCR and ASR transcripts. In the first step, all OCR and ASR words will be arranged to an appropriate video segment. This enables us to efficiently calculate the weight factor for

[1] *http://www.youtube.com*
[2] *http://webcast.berkeley.edu*



Lectures

Name	is visible	number of segments	OCR analysis	speech recognition	Sorting
Semantic Web	yes	7	finished	analyzing (60 %)	
Suchmaschinen	yes	7	finished	finished	
P2P-Netzwerke und Cloud-Computing	yes	7	waiting (0 %)	finished	
Web Services	yes	6	finished	finished	
Webprogrammierung	yes	8	finished	finished	
CSS - Cascading Style Sheets	yes	6	finished	finished	
XML - Extensible Markup Language	yes	7	finished	not started yet	
HTML - Hypertext Markup Language	yes	7	finished	not started yet	
HTTP - Hypertext Transfer Protocol	yes	5	finished	not started yet	
URI - Uniform Resource Identifier	yes	4	finished	not started yet	

Add new Lecture | Add new Live Lecture

Figure 16. Management page of the lecture video portal for the multimedia analysis

each keyword in the later stage. Since our system only considers nouns as the keyword, we thus capture nouns from the transcripts by applying Stanford POS Tagger (*Stanford Log-linear Part-Of-Speech Tagger*) [19] secondly.
Subsequently, we use the *stemming algorithm* to capture words with variant forms but a common meaning. This process leads to improve recall rate of the retrieval.
Since the OCR transcript might contain spelling mistakes, we have to perform a dictionary-based filtering for sorting out the error words. In order to avoid the out-of-vocabulary issue, the standard dictionary will be extended by adding technical terms of our lectures.
Then, the weight factor of each remaining word is calculated by TFIDF (*Term Frequency Inverse Document Frequency*) scores [16], which is described as following:

Term Frequency represents the appearance frequency of a given term in the given document. In our case, each video segment is considered as a local context. Therefore, the OCR and ASR transcripts of each video segment will be considered as one document. The term frequency is then calculated by dividing the total word count of a segment document by the term appearance count.

Inverse Document Frequency measures whether a term is common or rare across all documents. To our purpose, the IDF is calculated by using the total number of segments and the number of segments containing the term.

TFIDF ranks candidate keywords according to their statistical frequencies, which dose not reflect the location information of the keyword, while the location might be important for ranking keywords extracted from the web pages or lecture slides. In our framework, the OCR text line objects have been classified into several classes (cf. section 6). Therefore, we are able to apply the classified text line types to improve the keyword weighting result.

8.2 Utility of Extracted Keywords

The segmented key frames can help to guide the users with navigation within the video, while they summarize each video. The extracted keywords from each segment can further provide a more abstract information to the users. This enables the user to get the orientation about which lecture segment within the video he is interested in, efficiently.
To enrich the web page with more meta data, for example for better indexing of the web site or better accessibility, we can also use the keywords. The top 100–150 keywords could be illustrated for each video segment according to their weights. The users are able to click on the keyword labels, and the distribution of the corresponding keyword in the video could be illustrated subsequently. This feature can serve as visual guideline for the user to navigate through a lecture video. Furthermore, more meta data can be provided for the web search engine crawlers so that the video can be found easier.

Table 1. Experimental Results of Video Segmentation

Video URL [1]	Detected slides count	Slides count in ground truth	Correctly detected slides
.../5410/	56	51	49
.../5365/	41	36	36
.../5478/	23	24	22
.../4619/	49	48	48
.../4884/	54	51	51
.../5082/	70	66	66
.../5131/	78	77	77
.../4479/	27	26	26
.../4913/	21	20	19
.../4468/	72	78	72
.../5314/	51	59	51
.../4207/	18	18	18
.../4586/	31	29	28
.../4490/	40	39	39
.../5281/	27	25	24
.../5318/	34	30	30
.../5327/	18	17	17
.../4713/	70	65	65
.../4206/	20	18	18
.../4847/	60	60	60
	Recall	Precision	F_1 Measure
Total	0.975	0.948	0.964

[1] http://www.tele-task.de/archive/lecture/overview...

9 Evaluation and Experimental Results

To evaluate our system's performance, we restricted on testing the efficiency of the video segmentation, text detection, text recognition algorithms. Furthermore, a SVM classifier has been trained and evaluated with HOG and image intensity histogram features. We also evaluated the lecture outline extraction method by using test data.
Our test dataset consists of lecture videos and video frames from the *tele-TASK* project [18]. We randomly chose 20 lecture videos with varying layouts and font styles for the estimation of our segmentation algorithm, as illustrated in Fig. 17; in order to evaluate the text detection and recognition procedures, we used 180 video test frames including respective manual text annotations. The test data is available at [3].
All experiments were executed on a Mac OS X, 2.4 GHz platform. Our system is implemented in C++. Overall, our test dataset contains 2997 text lines, 12533 words, and 76099 characters. The video frame sizes vary between 640×480 and 1024×768 pixels.

9.1 Evaluation of Video Segmentation

The overall number of detected segments is 860, of which 816 were detected by our segmenter correctly. The achieved segmentation recall and precision are 98 % and 95 % respectively. The detailed evaluation data for each video is shown in Table 1. Since this experiment is defined to evaluate our slide segmentation method, videos embedded within slides are not considered in our evaluation.

9.2 Evaluation of Text Detection

In order to evaluate our text detection methods, we selected 180 key frames from the segmentation results. Then, we applied the following, commonly used evaluation methodology: We distinguish between a pixel-based evaluation, where the percentage of overlapping pixels in the ground truth and the experimental results are used to determine recall and precision, and an evaluation based on text bounding boxes. For the latter one, a bounding box is deemed to be a

[3] *http://www.yanghaojin.com/research/videoOCR.html*

Table 2. Text Detection Results

	Recall	Precision	F_1 Measure
Pixel-based	0.977	0.996	0.986
Bounding-box-based	0.963	0.925	0.943

Table 3. Text Detection Comparison Results Based on Bounding-Box-Based Evaluation Method

	Recall	Precision	F_1 Measure	seconds per frame
Method in [4]	0.9	0.93	0.915	2.82
Our method	0.963	0.925	0.943	0.47

correct match if the overlap with the ground truth exceeds 80 %. Table 2 shows the text detection results of our video test data set.
Furthermore, we compared our text detection algorithm with the method proposed in [4] by using our data set (cf. Table 3). Although the achieved precision of our method is slightly lower compared to [4], we could achieve a more significant improvement in recall and processing time.

9.3 Evaluation of Text Recognition

We applied the open-source OCR engine *tesseract-ocr*[4] for the text recognition. The recognition evaluation is based on the results of our preceding text detection, using test frames containing English texts.
Overall, we achieve recognizing 92 % of all characters and 85 % of all words correctly. To some extent, this is due to some text passages are set in special graphical and artistic fonts and are therefore difficult to read by standard OCR libraries.

Figure 17: Exemplary frames of our test videos which have varying layouts and font styles for the estimation of our segmentation algorithm.

9.4 Evaluation of The SVM Classifier

For the evaluation of the slide frame classifier 2597 slide frames and 5224 none-slide frames have been collected in total to train the SVM classifier. All slide frames are selected from a large amount of lecture videos. To create a none-slide frame set, which has varying image genres, we have selected additional 3000 images from [5]. In addition, about 2000 none-slide video frames have been used, which are extracted from the lecture video database. The test set is build from 240 slide frames and 233 none-slide frames, which differ from the training set.

[4] *http://code.google.com/p/tesseract-ocr/*
[5] *http://www.flickr.com*

Table 4. Feature Comparison Results

	Recall	Precision	F_1 Measure
HOG feature	0.996	0.648	0.785
Image intensity histogram feature	0.91	0.93	0.92

In our experiment, we have used RBF (*Radial Basis Function*) as kernel for the SVM classifier. Parameters have been determined by cross-validation function. We have used the HOG feature with 8 gradient directions, whereas the local region size has been set to 64×64. For calculating an image intensity histogram, 256 histogram bins have been created corresponding to 256 image grayscale values. Then, the normalized histogram values have been applied to train the SVM classifier. We have evaluated the normalization factor (cf. Fig. 18), which has proven to serve best when set to 1000.

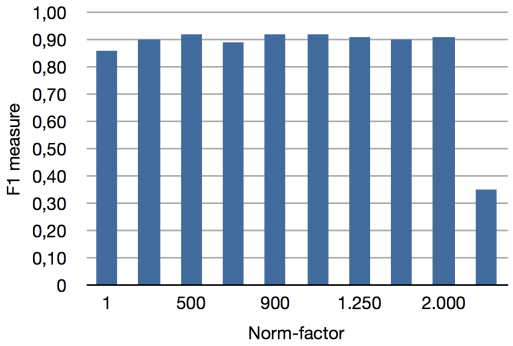


Figure 18. Evaluation results of the normalization factor of image intensity histogram feature

The results of HOG and image intensity histogram features are illustrated in Table 4. Both of two features achieve a good recall rate for slide frame recognition. However, comparing with HOG feature, the intensity histogram feature can achieve a significant improvement in precision. Moreover, the processing speed is also much faster.

9.5 Evaluation of Lecture Outline Extraction Method

We have also used the randomly selected 180 unique slide frames for the evaluation of lecture outline extraction method. The percentage of correctly retrieved outline words in the experimental results and the ground truth are used to calculate the recall and precision rate.
To evaluate the outline extraction method, we determine not only whether the outline type is correctly distinguished by slide structure analysis. We also calculate the word accuracy for each detected outline object. Therefore, the accuracy of our outline extraction algorithm is based on the OCR recognition rate.
We have defined recall and precision metrics for the evaluation as follows:

$$Recall = \frac{number\ of\ correctly\ retrieved\ outline\ words}{number\ of\ all\ outline\ words\ in\ ground\ truth}$$

$$Precision = \frac{number\ of\ correctly\ retrieved\ outline\ words}{number\ of\ retrieved\ outline\ words}$$

Table 5 shows the experimental results. We have performed the evaluation for the title and subtitle-key point, respectively. The results of subtitle-key point extraction method still have a significant improvement space. This is due to that some key points are written in the different color and do not have geometrical differences to the normal text font. Moreover, some subtitles have event a lower height than normal texts.
However, the extracted titles are already robust for lecture video exploring. As the upcoming improvement, using a better OCR engine and/or enhancing the text segmentation results will further improve both, the text recognition and outline extraction results.

Table 5. Evaluation of Lecture Outline Extraction Method

	Recall	**Precision**	F_1 **Measure**
Title	0.86	0.95	0.9
Subtitle-key point	0.61	0.77	0.68

10 Conclusion and Future Work

In this paper, we have presented an effective, robust system for indexing and analyzing lecture videos. We have developed and evaluated a novel slide video segmenter and a text localization and verification scheme. Furthermore, we have proposed a new method for extraction of timed lecture outlines using the geometrical information and stroke width of detected text lines. The video exploring and indexing can be performed by using both, slide shots (visual information) and extracted lecture outlines (textual information).
For our lecture video test set, the achieved segmentation accuracy is $96\,\%$. Furthermore, more than $94\,\%$ of all text regions and $96\,\%$ of all slide title locations are correctly detected. $85\,\%$ of all words are recognized accurately.
As an upcoming improvement, implementing context- and dictionary-based post-processing will improve the text recognition rate. The slide text classification method can be further improved by using font color features.
Since it is desired that the segmented slide shots and extracted lecture outlines become frequently used functionalities within our tele-teaching portal, the usability and utility need to be evaluated within a user study.

References

[1] John Adcock, Mattthew Cooper, Laurent Denoue, and Hamed Pirsiavash. Talkminer: A lecture webcast search engine. In *Proc. of the ACM international conference on Multimedia*, MM '10, pages 241–250, Firenze, Italy, 2010. ACM.

[2] Marios Anthimopoulos, Basilis Gatos, and Ioannis Pratikakis. A two-stage scheme for text detection in video images. *Journal of Image and Vision Computing*, 28:1413–1426, 2010.

[3] F. Chang, C-J. Chen, and C-J. Lu. A linear-time component-labeling algorithm using contour tracing technique. *Computer Vision and Image Understanding*, 93(2):206–220, January 2004.

[4] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scene with stroke width transform. In *Proc. of Computer Vision and Pattern Recognition*, pages 2963–2970, 2010.

[5] Ch. Meinel F. Moritz, M. Siebert. Community tagging in tele-teaching environments. In *Proc. of 2nd International Conference on e-Education, e-Business, e-Management and E-Learning*, 2011.

[6] T-C. Pong F. Wang, C-W. Ngo. Structuring low-quality videotaped lectures for cross-reference browsing by video text analysis. *Journal of Pattern Recognition*, 41(10):3257–3269, 2008.

[7] R. C. Gonzalez and R. E. Woods. Digital image processing. In *2nd edn., Prentice Hall, Englewood Cliffs*, pages 602–608, 2002.

[8] Alexander Haubold and John R. Kender. Augmented segmentation and visualization for presentation videos. In *Proc. of the 13th annual ACM international conference on Multimedia*, pages 51–60. ACM, 2005.

[9] J. Hunter and S. Little. Building and indexing a distributed multimedia presentation archive using smil. In *Proc. of ECDL '01 Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries*, pages 415–428, London, UK, 2001.

[10] Hyun Hee Kim. Toward video semantic search based on a structured folksonomy. *Journal of the American Society for Information Science and Technology*, pages 478–492, 2011.

[11] B. Triggs N. Dala. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

[12] N. Otsu. A threshold selection method from gray level histogram. *IEEE Transactions on System, Man, Cybernetics*, 19(1):62–66, 1978.

[13] Xueming Qian, Guizhong Liu, Huan Wang, and Rui Su. Text detection, localization and tracking in compressed video. In *Proc. of International Conference on Signal Processing: Image Communication*, pages 752–768, 2007.

[14] Stephan Repp, Jörg Waitelonis, Harald Sack, and Christoph Meinel. Segmentation and annotation of audiovisual recordings based on automated speech segmentation and annotation of audiovisual recordings based on automated speech recognition. In *In Proc. of 8th Int. Conf. on Intelligent Data Eng. and Automated Learning*, pages 620–629, Birmingham, UK, 2007.

[15] Harald Sack and Jörg Waitelonis. Integrating social tagging and document annotation for content-based search in multimedia data. In *Proc. of the 1st Semantic Authoring and Annotation Workshop*, Athens , USA, 2006.

[16] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. In *INFORMATION PROCESSING AND MANAGEMENT*, pages 513–523, 1988.

[17] I. Sobel. An isotropic 3×3 image gradient operator. In *Machine Version for Three-Dimensional Scenes*, pages 376–379. H. Freeman, Ed. N. Y.: Academic, 1990.

[18] teletask. http://www.tele-task.de.

[19] Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *In Proc. of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL 2003)*, pages 252–259, 2003.

[20] Haojin Yang, Christoph Oehlke, and Christoph Meinel. A solution for german speech recognition for analysis and processing of lecture videos. In *Proc. of 10th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2011)*, pages 201–206, Sanya, Heinan Island, China, 2011. IEEE/ACIS.

[21] Haojin Yang, Maria Siebert, Patrick Lühner, Harald Sack, and Christoph Meinel. Automatic lecture video indexing using video ocr technology. In *Proc. of International Symposium on Multimedia (ISM)*, pages 111–116, 2011.

[22] Ming Zhao, Shutao Li, and James Kwok. Text detection in images using sparse representation with discriminative dictionaries. *Journal of Image and Vision Computing*, 28:1590–1599, 2010.