This paper has been published at the seventh International Conference on Signal Image Technology & Internet-Based Systems, Dijon, France 2011

Lecture Video Indexing and Analysis Using Video OCR Technology

Haojin Yang, Maria Siebert, Patrick Lühne, Harald Sack, Christoph Meinel
Hasso Plattner Institute (HPI), University of Potsdam
P.O. Box 900460, D-14440 Potsdam
email: {Haojin.Yang, Maria.Siebert, Harald.Sack, Meinel}@hpi.uni-potsdam.de;
Patrick.Luehne@student.hpi.uni-potsdam.de

***Abstract*—The text displayed in a lecture video is closely related to the lecture content. Therefore, it provides a valuable source for indexing and retrieving lecture video contents. 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: Firstly, we developed a novel video segmenter for an automated slide video structure analysis. Having adopted a localization and verification scheme, we perform text detection secondly. We employ SWT (*stroke width transform*) not only to remove false alarms from the text detection, but also to analyze the slide structure further. To recognize texts, a multi-hypotheses framework is adopted, that consists of multiple text segments, OCR, spell checking and result merging processes. Finally, we implemented a novel algorithm for slide structure analysis and extraction by using the geometrical information of detected text lines. The accuracy of the proposed approach is proven by evaluation.**

***Keywords*-multimedia retrieval; video indexing; video OCR; video segmentation; e-learning; recorded lecture videos.**

I. 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 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. 2(a)).

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. 1. We developed a video segmenter based on fast connected component (CC) analysis and slide transition recognition. For text detection, we realized a localization and verification scheme: An edge-based multi-scale text detector serves to achieve a high recall rate with low computational time expenses. The verification consists of an SWT-based analysis to remove false alarms.

To segment text from its heterogeneous background, 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 of those text objects for slide title identification and text line classification. Operability and accuracy of our approach have been evaluated.

The rest of this paper is organized as follows: Section 2 presents related work, whereas the sections 3, 4, 5 and 6 describe our proposed framework in detail. We provide the evaluation and experimental results in section 7. Section 8 concludes the paper with an outlook on future work.

II. Related Work

Liška et al. propose an automated system for authoring hypermedia lecture recordings [5]. They use a VGA frame grabber to extract the slide frames and a common OCR tool on the extracted frames for text recognition. Instead of applying the video segmentation process, OCR is executed on every extracted slide frame. This is much less efficient concerning computation time and implies a large amount of redundancies. Furthermore, they don't make use of a text

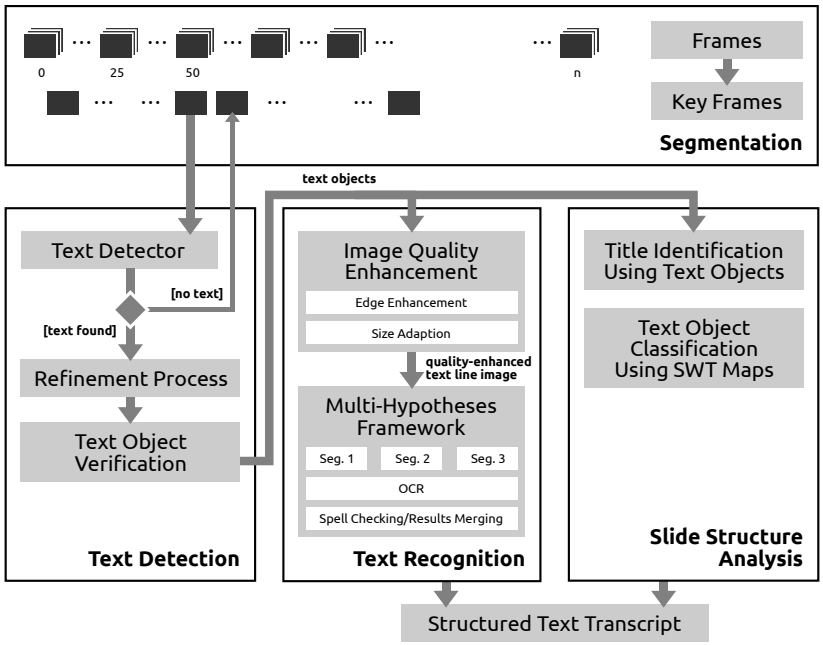


Figure 1. The entire system workflow. After having segmented frame sequences, text detection is performed on every single key frame in order to find text occurences. The resulting text objects are then used for text recognition and slide structure analysis.

localization process to identify text regions. Therefore, the recognition accuracy of their approach is much lower than our system's.

Hunter et al. propose a *SMIL* (Synchronized Multimedia Integration Language) based framework to build and index a multimedia presentation archive [11]. First, the *Real* format based video archive is already out of date. Although their system also records slide streams via video camera, they do not apply OCR analysis on it. Each speaker have to prepare a slide file in *PDF* format and requested to upload it to a multimedia server after the presentation. Then, the uploaded slide files have to be synchronized with the lecture video. The indexing is based on text resources of slide files. In contrast, since our system directly analyze the videos, thus we do not need to take care about the slide format and the synchronisation task is not required. Since their synchronisation algorithm is simply based on binary image difference matric, it might not work robustly when animated content-buildup is used in the presentation slide. Although the OCR based approach may introduce a few errors, however the recognized texts are still robust enough for the indexing task by performing a dictionary based filtering.

Yang et al. use the *Automatic Speech Recognition* (ASR) output for lecture video retrieval [10]. However, since ASR for lecture videos is still an active research area, most of those recognition systems have relative poor recognition rates—in contrast with OCR results—degrading the indexing and the retrieval yet.

Epshtein et al. propose a text detection method using *Stroke Width Transform* (SWT) for nature scene images [6]. 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 (see section 5).

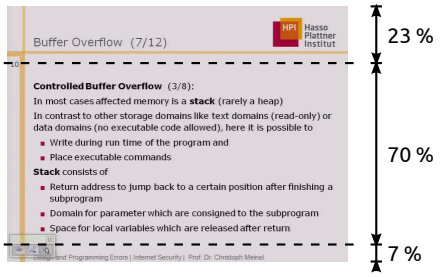
Chen et al. present a two-stage text detection method containing a localization and a verification step [3]. They use a fast detector to coarsely detect text while the verification algorithm identifies the coarse results using *Support Vector Machine* (SVM) and *Constant Gradient Variance* (CGV) features. However, the machine learning verification can only take a binary decision. It might not work robustly for the bounding boxes obtained from the localization stage that contain both text and non-text pixels.

Lienhart et al. propose an approach for text detection in both videos and images [4]. In their system, a complex-valued multilayer feed-forward network is trained for text detection. Localized text lines are refined by exploiting the temporal redundancy of text in videos. Their text line tracking algorithm processes every frame within the text line occurrence duration—therefore, their approach is less efficient concerning computation time.

III. Video Segmentation

By decomposing the video into a set of representative key frames, we are able to index the video. These selected frames

(a)



(b)

Figure 2. (a) An example lecture video. Video 2 shows the speaker giving his lecture, whereas his presentation is played in video 1. (b) Slide content distribution. Title and content act as indicators for actual slide transitions and are thus analyzed in the second step.

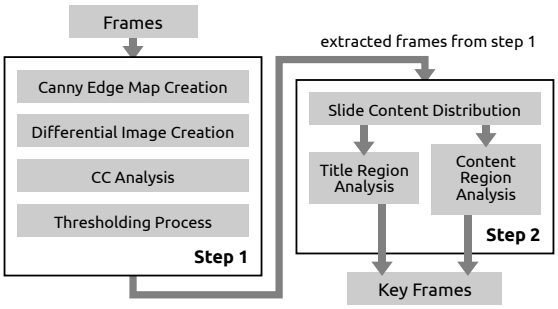


Figure 3. Segmentation workflow. (Step 1) Adjacent frames are compared with each other by applying *Connected Components Analysis* on their differential edge maps. (Step 2) Title and content region analysis ensure that only actual slide transitions are captured.

will be processed in other steps later, since they capture and encapsulate the entire video content.

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. 2(a) 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 [1]. One deficiency of this method is that the high-frequency image noise within video frames can still degrade the segmentation accuracy.

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 connected components (*CC*s). 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 Chang et al. [2].

Our segmentation algorithm consists of two steps (cf. Fig. 3): 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 differential edge map of two adjacent frames and perform the CC analysis on this map. 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 of a differential image exceeds T_{s1}.

The result of the first segmentation step is too redundant for indexing, since there are many changes within the same slide. 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. 2(b)). Let R_t and R_c denote the title and the content regions in Fig. 2(b) which account for 23 % and 70 % of the entire frame height respectively. If the amount of CCs in the differential edge image for R_t exceeds the threshold T_{s2}, a slide page transition is captured. Otherwise, in R_c, we detect the first text lines top-down and bottom-up respectively, and perform the corresponding CC differencing analysis on these text line images from two adjacent frames (cf. Fig. 4). If both of the two difference values exceed T_{s2}, a slide page transition is also captured. In our study, we set T_{s1} and T_{s2} to 20 and 5 respectively. All thresholds were learned on our test running.

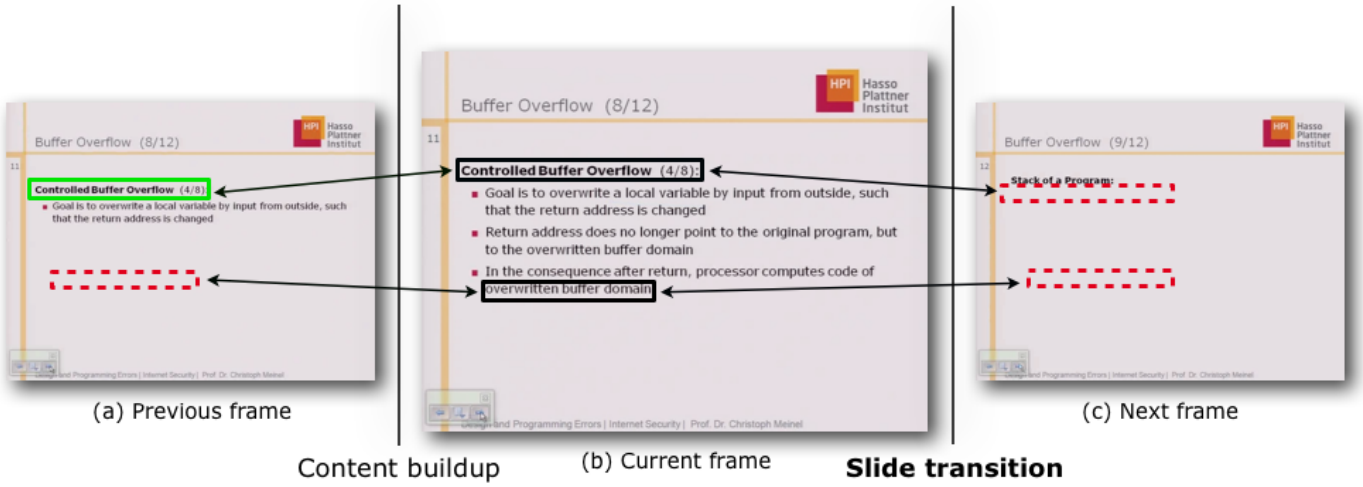


Figure 4. We detect the first text line top-down and bottom-up in R_c of current video frame and perform the CC differencing analysis on those text line sequences from adjacent frames. In frame (a) and (b), a same text line (top) can be found; Whereas in frame (b) and (c), the CC differences of both text lines will exceed the T_{s2}. A slide transition is therefore captured between frame (b) and (c).

IV. 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.

A. Text Localization

To localize text, we firstly produce a horizontal and a vertical edge map for an input image using the Sobel filter [9]. 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. Subsequently, we apply Otsu thresholding to binarize the dilation maps [7]. Due to the connectivity of horizontal and vertical character edges in text-like regions [3], a new image is created by covering the intersected regions of both binary maps. Ultimately, the initial bounding boxes are generated after performing CC analysis (cf. Fig. 5 (a)-(f)).

B. 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. 5 (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.

C. Text Verification

We apply a verification method based on SWT analysis. Epshtein et al. [6] have proved in their work that the stroke width feature is robust enough to distinguish text from other complicated image elements such as vegetation. Per pixel,

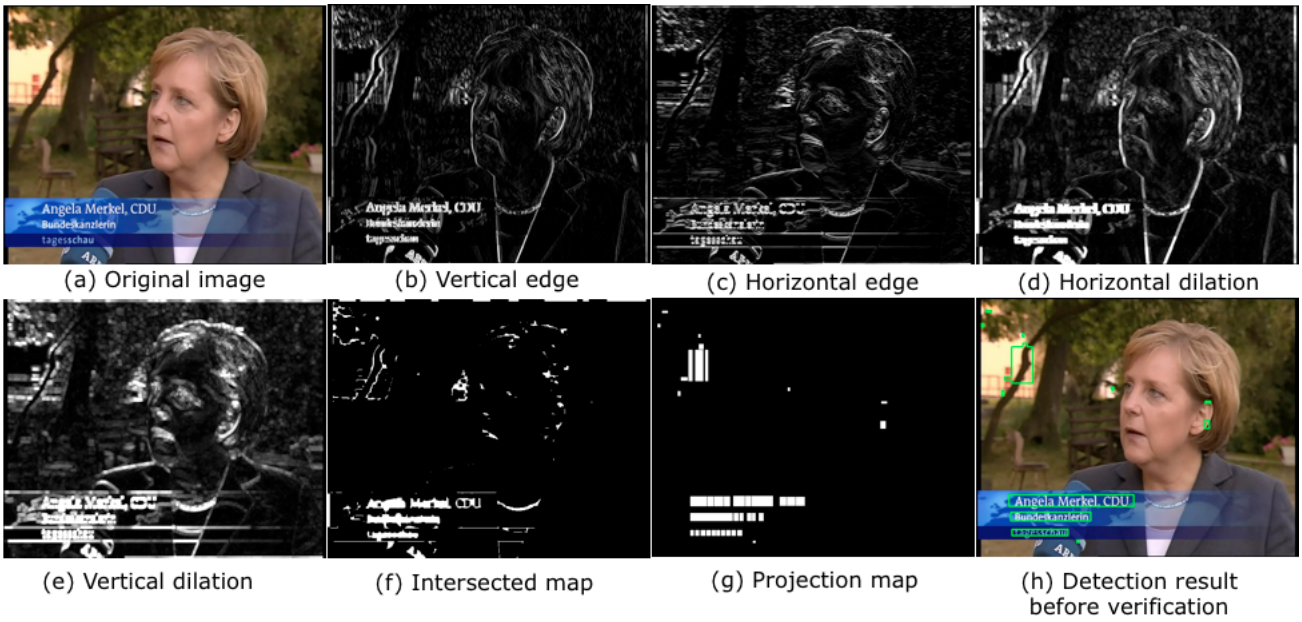


Figure 5. Workflow of the edge based text detection. (b) (c) Vertical and horizontal edge maps. (d) (e) Dilation maps of corresponding edge images. (f) Intersected image of binarized dilation maps. (g) After projection profiles. (h) Detection results without verification.

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. 6).

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 [6]. In oder to accommodate both bright text on dark background and vice-versa, they apply the algorithm twice for each gradient direction—this leads to a loss of performance. Anyway, it is really difficult or sometimes impossible to determine the correct gradient direction for images containing both kinds of gradient directions. 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 only apply SWT on those subimages in order to improve performance.

Hence, it is relatively easy to determine the gradient direction for such bounding box images: First of all, we convert the bounding box image to a grayscale image and binarize it subsequently using Otsu thresholding method. Then, we could obtain the text gradient direction by calculating the black pixel and white pixel ratio, where black pixels represent the background and white pixels represent the font respectively.

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 $MaxVar$,

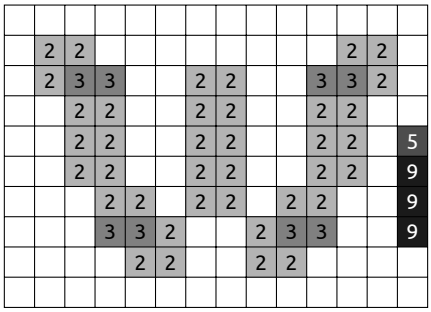


Figure 6. An example output image from *stroke width transform* (SWT) for character *w*.

- its mean stroke width exceeds a threshold range $MinStroke$ and $MaxStroke$.

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. [6] 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. 7 shows the results after verification which are the final output of the text detection.

V. Text Recognition

The detected text objects from the previous step serve as the input for the text recognition. Usually, texts in video

Figure 7. Refined bounding boxes from text detection. The results are automatically verified in order to remove false alarms.

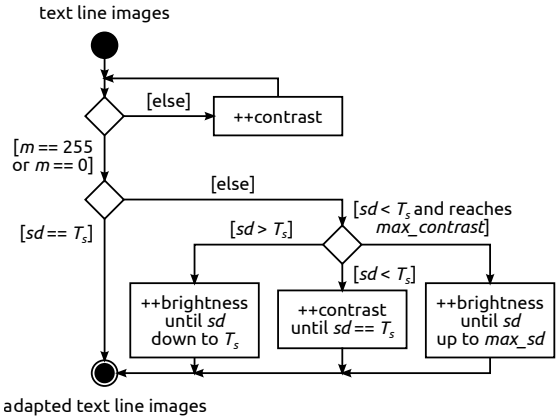


Figure 8. Workflow of the dynamic adaptive contrast and brightness adjustment. After applying the algorithm, the text appears black on a plain white background. m and sd denote the median and standard deviation of the image histogram, T_s is the threshold value for sd.

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 bicubic interpolation. Then, the texts will be segmented and recognized by a subsequent multi-hypotheses framework.

A. A Multi-Hypotheses Framework

1) Text Segmentation: We employ three methods in order to segment texts: the Otsu thresholding algorithm, Gaussian-based adaptive thresholding [12] 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 and vice-versa. The workflow of the algorithm is depicted in Fig. 8, 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. For our purpose, we chose a fixed value of 100 for the standard deviation threshold T_s.

2) Recognition and Spell Analysis: After the segmentation process, we generate three OCR results for each text line image by using a common 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.

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.

VI. Slide Structure Analysis

Observing the slide frames, we can realize that the title and substitle texts have more significancy 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 and text object classification methods.

A. 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 $MinW$,
- the x position of its top-left corner is lower than a threshold $MaxX$—has been set to $frameWidth * 0.77$,
- it is one of the three highest bounding boxes and has the uppermost position.

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.

B. Text Object Classification

After the title identification, the remaining text objects are classified into three classes: *subtitle/key point* (bold texts), *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 classfication is processed as follows:

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

VII. Evaluation and Experimental Results

To evaluate our system's performance, we restricted on testing the efficiency of the video segmentation, text detection, text recognition, and title identification algorithms. Our test data set consists of lecture videos and video frames from the *tele-TASK* project [13]. We randomly chose 20 lecture videos with varying layouts and font styles for the estimation of our segmentation algorithm, as illustrated in Fig. 9; 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 [14].

All experiments were executed on a Mac OS X, 2.4 GHz platform. Our system is implemented in C++. Overall, our test data set contains 2997 text lines, 12533 words, and 76099 characters. The video frame sizes vary between 640×480 and 1024×768 pixels.

A. 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 available at [14]. Since our algorithm is defined for slide videos (like presentations), videos embedded within slides are not considered in our evaluation.

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

	Recall	**Precision**	**F_1 Measure**	**seconds per frame**
Method in [6]	0.9	0.93	0.915	2.82
Our method	0.963	0.925	0.943	0.47

B. 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 correct match if the overlap with the ground truth exceeds 80 %. Table II shows the text detection results of our video test data set.

Furthermore, we compared our text detection algorithm with the method proposed in [6] by using our data set (cf. Table I). Although the achieved precision of our method is slightly lower compared to [6], we could achieve a more significant improvement in recall and processing time.

We also evaluated the title identification algorithm on the same test set; The achieved recall and precision are of 98 % and 94 %.

C. Evaluation of Text Recognition

We applied the open-source OCR engine *Tesseract*[1] 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.

Table II
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

VIII. 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

[1] http://code.google.com/p/tesseract-ocr/

Figure 9. Example frames of our test videos which have varying layouts and font styles for the estimation of our segmentation algorithm.

have proposed a new method for slide structure analysis using the geometrical information and stroke width of detected text lines.

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 further. The video segmentation algorithm will be extended so that the slide frames and the embedded video frames could be classified correctly. For the embedded videos, we will apply another video segmenter since the slide video segmenter is not suitable. In addition, we will ameliorate and evaluate the text line classification algorithm so that the automated creation of the lecture outline can take place. The OCR transcripts we obtain using our approach can be used in different manners: For instance, intelligent search algorithms and new recommendation methods for websites can be developed, based on the output of our software.

References

[1] J. Adcock, M. Cooper, L. Denoue, H. Pirsiavash, L. A. Rowe. "TalkMiner: A Lecture Webcast Search Engine," in *Proceeding MM '10 Proceedings of the international conference on Multimedia*, New York, NY, USA, 2010.

[2] F. Chang, C-J. Chen, and C-J. Lu. "A Linear-Time Component-Labeling Algorithm Using Contour Tracing Technique," in *Computer Vision and Image Understanding*, vol. 93, no. 2, January 2004, pp. 206–220.

[3] D. Chen, J.-M. Odobez, H. Bourlard. "Text Detection and Recognition in Images and Video Frames," in *Journal of The Pattern Recognition Society*, 2004, pp. 595–608.

[4] R. Lienhart, A. Wernicke. "Localizing and Segmenting Text in Images and Videos," in *Proc. of IEEE Transactions on Circuits and Systems for Video Technology*, 2002.

[5] M. Liška, V. Rusňák, E. Hladká. "Automated Hypermedia Authoring for Individualized Learning," in *8th International Conference on Information Technology Based Higher Education and Training*. Kumamoto, Japonsko: Kumamoto University, Kumamoto, Japan, 2007, p. 4.

[6] B. Epshtein, E. Ofek, Y. Wexler. "Detecting Text in Natural Scene with Stroke Width Transform," in *Proc. of Computer Vision and Pattern Recognition*, 2010, pp. 2963–2970.

[7] N. Otsu. "A Threshold Selection Method from Gray-Level Histograms," in *IEEE Transactions on Systems, Man and Cybernetics*, vol. SCM-9, no. 1, 1979, pp. 62–66.

[8] V. Schillings, C. Meinel. "tele-TASK—Teleteaching Anywhere Solution Kit," in *Proc. of ACM SIGUCCS 2002*, Providence (Rhode Island), USA, 2002, pp. 130–133.

[9] I. Sobel. "An Isotropic 3×3 Image Gradient Operator," in *Machine Version for Three-Dimensional Scenes, H. Freeman, Ed. N. Y.: Academic*, 1990, pp. 376–379.

[10] H-J. Yang, C. Oehlke, and C. Meinel. "A Solution for German Speech Recognition for Analysis and Processing of Lecture Videos," in *10th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2011)*, Sanya, Heinan Island, China, May 2011, pp. 201–206.

[11] J. Hunter, 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*, London, UK, 2001, pp. 415–428.

[12] R. C. Gonzalez, R. E. Woods. "Digital Image Processing," 2nd edn., Prentice Hall, Englewood Cliffs, 2002, pp. 602–608.

[13] "tele-TASK". Internet: http://www.tele-task.de [23.09.2011].

[14] Internet: http://www.yanghaojin.com/research/videoOCR.html [23.09.2011].