Research Report

Automatic Text Extraction From Video For Content-Based Annotation and Retrieval

Jae-Chang Shim, Chitra Dorai, Ruud Bolle
IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - China - Haifa - T. J. Watson - Tokyo - Zurich

Automatic Text Extraction from Video for Content-Based Annotation and Retrieval

Jae-Chang Shim, Chitra Dorai and Ruud Bolle
IBM Thomas J. Watson Research Center
P.O. Box 704, Yorktown Heights, New York 10598
{jcshim,dorai,bolle}@watson.ibm.com

Abstract

Efficient content-based retrieval of image and video databases is an important emerging application due to rapid proliferation of image and digital video data on the Internet and corporate intranets and exponential growth of video content in general. Text either embedded or superimposed within video frames is very useful for describing the semantic content of the frames, as it enables both keyword and free-text based search, automatic video logging, and video cataloging. Extracting text directly from video data becomes especially important when closed captioning or speech recognition is not available to generate textual transcripts of audio, or when video footage that completely lacks audio needs to be automatically annotated and searched based on frame content.

Towards building a video query system, we have developed a scheme for automatically extracting text from digital images and videos for content annotation and retrieval. In this paper, we present our approach to robust text extraction which can handle complex backgrounds in video frames, deal with different font sizes, font styles, and font appearances such as normal and inverse video. Our algorithm results in segmented characters from video frames that can be directly processed by an OCR system to produce ASCII text. Results from our experiments with over 5,000 frames obtained from twelve MPEG-1 video streams demonstrate the good performance of our system in terms of text identification accuracy and computational efficiency.

Key words: Video, digital images, content-based search and annotation, automatic text extraction, image segmentation, character recognition.

1 Introduction

The ongoing proliferation of digital image and video databases, due to many new applications such as multimedia authoring, video conferencing, telepresence, and video-on-demand systems

has led to the increasing demand for systems that can query and search large video databases efficiently and accurately for desired video clips [1, 2, 3]. In practice, most video data is currently annotated manually, using keywords, through a preview of the video. However, this practice is often bedeviled by inadequate choice of keywords, incomplete description of the video content, and the subjective biases of the annotator, adversely affecting accuracy in the search and retrieval phase. Moreover, manual annotation is extremely time consuming, expensive, and unscalable in the face of ever growing video databases. Therefore, automatic extraction of video frame and segment descriptions is desirable in order to annotate and search large video databases.

Video sequences contain a rich combination of images, sound, motion, and text. Text is abundant in videos with credits and titles, especially at the beginning and end of a video. In news videos, text is often used as captions to mention the location of the event being described, the time stamp, the name of the anchor person or narrator, brief topic titles, phone numbers, etc. During the broadcast of sports, statistics about the game or players are often superimposed on the video in textual form, instead of being spoken aloud. Video commercials ensure that the product name, key product information and other shopping information are printed as readable text for consumers. Some exciting application areas where text extracted from video is extremely useful include retrieval of pertinent videos based on product identifiers from large video warehouses for product comparison and online retail/shopping, generation of electronic catalogs, and annotation and search of image and video libraries.

When video text is captured, it not only provides keywords for search but also aids in highlighting events which can then be used for summarizing a video. Automatic identification and logging of the beginning and end of key events can be accomplished using the extracted captions. Text can also be used to enable video categorization, cataloging of commercials, and efficient video browsing.

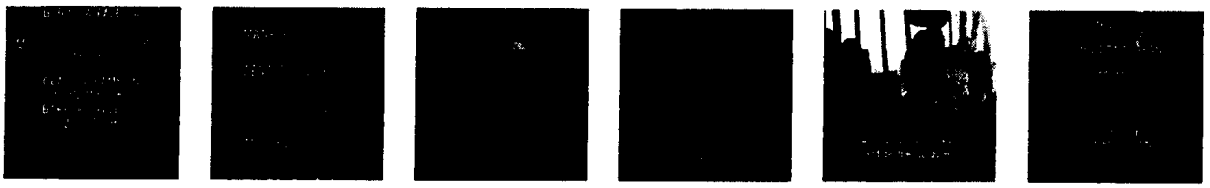

Figure 1: Text in video appears in different contexts, backgrounds, and font sizes.

Our primary goal in this paper is to present a novel computational scheme to extract the textual information present in images and video frames. Our algorithm employs a combination of region segmentation and feature-based progressive refinement techniques to handle the uncontrollable variations in text font size, style, gray level contrast, and complex image backgrounds in which the text is embedded or superimposed. Our technique does not make any

assumptions about the resolution of the video frames and appearance of text in normal or inverse video modes. The algorithm is general enough to handle text that appears as part of a scene as well as superimposed text, and robust enough to handle the noise and artifacts introduced by block-based MPEG encoding of digital video. We also present a mechanism to exploit the temporal relation of the text appearing over multiple consecutive frames for better performance of a text extraction system, including ours. Figure 1 shows the various contexts in which text conveying important information appears in videos. These images of size, 352×240 were obtained by decoding MPEG-1 streams of the videos.

2 Previous Work

The focus of a lot of research on text recognition has been optical character recognition in printed and hand-written documents. Such systems have matured over the years in specific application domains such as postal address recognition, processing of tax forms [4, 5], and license plate recognition [6]. These approaches employ strong heuristics about the illumination, image background, text location, font size, and font style that are specific to their application domains. They cannot be easily applied to extracting text superimposed on videos in very different footage types such as news story videos, movies, sports videos, commercials, etc.

Approaches to extracting text from video can be broadly classified into two categories: (i) methods that decompress video streams into individual frames and isolate the text regions in each of the frames independently and (ii) methods that decompress video streams into individual frames and not only analyze individual frames for text but also utilize the *temporality* of video sequences and perform interframe analysis to improve text segmentation results. To the first category belong several approaches that employ a combination of classical image segmentation techniques such as thresholding and filtering techniques to analyze video frames. Yu and Jain [7] analyze color still images and video frames and employ connected component analysis to locate text regions. Their analysis results in only locating the bounding blocks of the text regions, which therefore entails human involvement to recognize the characters. Wu and others [8] employ texture segmentation techniques to digitized video frames to obtain bitmaps of text that can be directly sent as input to OCR systems. However, texture-based text analysis can be very sensitive to font sizes and styles. Ohya et al. [9] focus on extracting text in still images only and assume that text present is almost upright, non-textured, and monochrome, and the characters are unconnected. Kannangara and others [10] have developed a system that analyzes C-SPAN video frames to extract captions using a combination of thresholding and vertical projection profiling techniques. The system is specialized to handle text that appears in C-SPAN videos and therefore uses specific knowledge about its location, image background, and contrast.

A few techniques work on image sequences derived from MPEG-1 video streams. Smith

and Kanade [11] verify the consistency of the presence of same text cluster over many frames to improve text location accuracy. This approach focuses only on locating text regions and the bitmaps containing text are retained as they are without obtaining the character outlines or separating the individual characters. The method relies on human identification of the character strings from the bitmaps. Our system avoids this, by segmenting the individual characters automatically and providing it as input to an OCR system. Yeo and Liu [12] extract only captions in videos and restrict the presence of text within a frame to a pre-specified set of locations (frame lines). They also assume that text frames span only a small portion of a shot and cannot handle long text sequences such as scrolling titles, credits, etc. Their technique locates only the regions containing text and does not achieve recognition. It also suffers severely when there are rapidly moving objects in the frames. Lienhart [13] proposes a technique that can handle text generated by video character generators. Scene text which is embedded as part of the scenery in the frame is not handled. Text segmentation results are enhanced by using interframe analysis. The system cannot handle text present in normal and inverse video modes.

3 Text Extraction from Video

Text extraction and recognition comprises of obtaining an image (scanned from a document or decoding a MPEG video clip), segmenting the image and extracting regions containing text only (sometimes referred to as text location), analyzing the text regions into blocks, lines, words, and characters, and finally recognizing the characters using OCR systems to output the text strings contained in the image. Text can appear in video anywhere in the frame and in different contexts. It appears as either *scene text* or as *superimposed text* [13]. Text that appears as part of the scene and is recorded with the scene is referred to as scene text and its presence can be in the scene as part of street and shop name boards, or on a person's clothing. It is difficult to extract scene text reliably due to unconstrained nature of its appearance. On the other hand, superimposed text is intended to carry and stress important information in video. It is typically generated by video title machines or graphical font generators in studios. Our system is designed to extract superimposed text and scene text that possesses typical text attributes. We do not assume any prior knowledge about frame resolution, text location, and font styles. Some common characteristics of text are exploited in our algorithm including monochromaticity of individual characters, size restrictions (characters cannot be too small to be read by humans or too big to occupy a large portion of the frame), and horizontal alignment of text (preferred for ease of human reading).

3.1 Locating Text and Extracting Characters: Our Approach

The input to our system is a sequence of gray level images obtained by decompressing MPEG-1 encoded video sequences. The primary goals of our system are (i) isolating regions that may contain text characters in an image from other image content, (ii) separating each character region from its surroundings, and (iii) verifying the presence of text by consistency analysis. "Image" and "frame" are used interchangeably in this paper.

3.1.1 A New Generalized Region Labeling (GRL) Algorithm

A basic process that is used repeatedly in our system for text extraction is that of labeling the pixels in an image based on a given criterion (e.g. gray scale homogeneity) using contour traversal, thus partitioning the image into multiple regions, then grouping pixels belonging to a region by determining its interior and boundaries, and extracting region features such as its MBR (minimum bounding rectangle), area, mean gray level, etc. We have developed a fast and efficient algorithm that uses chain-code to perform these tasks collectively. This new generalized region labeling (GRL) algorithm works on all types of images. It is fast, avoids recursion and external buffering, and extracts boundary and other region features along with segmenting regions efficiently. We refer the reader to [14] for more details.

3.1.2 Candidate Text Region Extraction

The objective of this first step is to remove the background from an input gray scale image where the background is interpreted as containing regions such as faces of the speakers, players, and other non-text scene content. The GRL algorithm is employed to extract homogenous regions from the input image. The criterion used to group pixels into a region is that the gray level difference between any pair of pixels within the region cannot exceed ± 10. By using the GRL algorithm to segment the image into nonoverlapping homogenous regions, we have obtained complete region information such as its label, outer and inner boundaries, number of holes within the regions, area, average gray level, gray level variance, centroid, and the MBR. We compared the performance of the GRL technique with that of split-and-merge and hybrid linkage techniques. The GRL algorithm is faster than the others on the average when tested over several images of different sizes, with the segmentation results being nearly the same.

Having obtained a number of homogenous regions in the image, non-text background regions are removed based on their size. Since text fonts are usually not larger than 24×32 in a 320×240 image, a region is removed if the width and height of its MBR are greater than 24 and 32 respectively. This size constraint can be adaptively modified depending on the image size. Thus, by employing a constraint that emphasizes the spatial proportion of text characters rather than the area which is often used by others, large regions of homogeneity which are unlikely to be text are effectively removed.

Within the remaining candidate regions, characters can sometimes be fragmented into multiple regions because of gray level variations in regions surrounding the characters. In order to group multiple touching regions into a single character region, we generate a binary image from the labeled region image where all the regions which do not satisfy the size constraint are marked "0" and the remaining regions are marked with "1". This binary image is processed using the GRL algorithm to obtain new connected regions. With the creation of a binary image, followed by a relabeling step, many small connected fragments of a whole candidate text region (that is likely to be a single character) are merged together.

3.1.3 Text Region Refinement

In this stage, the basic idea is to apply appropriate criteria to extract character segments within the candidate regions. Within a region, characters can be present embedded in a complex background and since OCR systems require text to be printed against a clean background for processing, the second stage attempts to remove the background within the regions while preserving the text. Since character outlines in these regions can be degraded and merged with the background, a local thresholding operation [15] is performed in each candidate region to separate the text from its surroundings and from other extraneous background contained within its interior. For each region $\mathcal{R}$, the local threshold $\mathcal{T}_l$ is determined as follows:

1. Compute the average gray value of the region and use it as an initial threshold $\mathcal{T}_l$.
2. Partition the region into two sub regions $\mathcal{R}_1$ and $\mathcal{R}_2$ based on $\mathcal{T}_l$.
3. Compute the average gray values t_1 and t_2 for $\mathcal{R}_1$ and $\mathcal{R}_2$ respectively and compute a new threshold, $\mathcal{T}_l' = (t_1 + t_2)/2$.
4. Use $\mathcal{T}_l'$ to re-partition $\mathcal{R}$ into two sub regions $\mathcal{R}_1$ and $\mathcal{R}_2$.
5. Repeat steps (3) and (4) until $\mathcal{T}_l'$ does not change between successive iterations. Use this as the final local threshold for the region.

Once thresholds are determined for all candidate regions, we compute positive and negative images, where the positive image contains region pixels whose gray levels are above their respective local thresholds and the negative image contains region pixels whose gray levels fall below their respective thresholds. Observe that the negative image will contain candidate text regions if that text appears in inverse video mode in the input. All the remaining processing steps are performed on both positive and negative images and their results are combined at the end of the last stage.

We further sharpen and separate the character region boundaries by performing a region boundary analysis. This is necessary especially when characters within a text string appear connected with each other and they need to be separated for accurate text identification. This

is achieved by examining the gray level contrast between the character region boundaries and the regions themselves. For each candidate region $\mathcal{R}$, a threshold $\mathcal{T}$ is computed:

$$\mathcal{T} = (\sum_k \mathcal{I}_{cb_k} + \sum_l \mathcal{I}_{i_l})/(N_{cb} + N_i) \quad (1)$$

where $\mathcal{I}_{cb_k}$ is the gray level of the pixel k on the circumscribing boundaries of the region and $\mathcal{I}_{i_l}$ is the gray level of the pixel l belonging to $\mathcal{R}$ (including interior and region boundary), N_{cb} is the number of pixels on the circumscribing boundaries of the region and N_i is the number of pixels in the region. A pixel is defined to be on the circumscribing boundary of a region if it does not belong to the region but at least one of its four neighbors (using 4-connectivity) does. Figure 2 illustrates a region's circumscribing boundaries. Those pixels in $\mathcal{R}$ whose gray

Figure 2: Broken lines constitute the circumscribing boundaries for a region containing two holes. Region interior and boundary are marked solid.

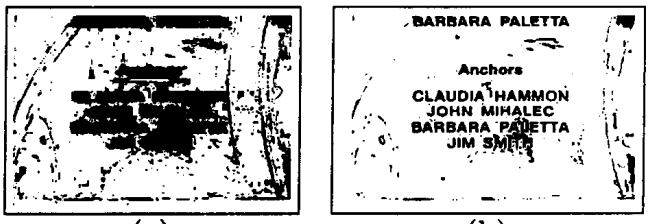

Figure 3: Intermediate results on an image shown in Figure 1: (a) binary image containing candidate text regions shown in black; (b) locally thresholded text regions.

level is less than $\mathcal{T}$ are marked as belonging to the background and discarded, while the others are retained in the region. Note that this condition is reversed for the negative image. This step is repeated computing a new $\mathcal{T}'$ until the value of $\mathcal{T}'$ does not change over two consecutive iterations. The character boundaries are sharpened and the regions are enhanced in terms of gray scale homogeneity at the end of this stage.

3.1.4 Text Characteristics Verification

The few candidate character regions that remain in the image are then subject to a verification step where they are tested for exhibiting typical text font characteristics. A candidate region is removed (i) if its area is less than 12 or its height is less than 4 pixels because small fonts are difficult to be recognized by OCR systems; (ii) if the ratio of the area of its MBR to the region area (fill factor) is greater than 4; (iii) if the gray level contrast with the background is low, i.e., if

$$|\sum_k \mathcal{I}_{cb_k} - \sum_l \mathcal{I}_{b_l}| < 20 \quad (2)$$

where $\mathcal{I}_{cb_k}$ is the gray level of the pixel k on the circumscribing boundaries of the region and $\mathcal{I}_{b_l}$ is the gray level of the pixel l on the boundaries of the region. Since region boundary information is easily available owing to our GRL algorithm, this new boundary-based test can be easily performed to handle the removal of noisy non-text regions.

3.1.5 Text Consistency Analysis

Consistency between neighboring text regions is verified to eliminate false positive regions. Unlike many systems, our system attempts to ensure that the adjacent regions in a line in the image exhibit the characteristics of a text string, thus verifying the global structure of a row of text in a local manner. This text consistency test includes (i) position analysis that checks intercharacter spacing. The width between the centroids of the MBRs of a pair of neighboring character regions that are retained is less than 50 pixels; (ii) horizontal alignment analysis of characters. The vertical centers of the MBRs of neighboring characters is within 6 pixels of one another; (iii) vertical proportions analysis of adjacent character regions. The height of the larger of the two regions is less than twice the height of the smaller region.

Given a candidate text string, we perform a final series of tests involving the MBRs of characters present in the string. The MBRs of the regions (characters) are first verified to be present along a line within a given tolerance of 2 pixels. Observe that characters present along a diagonal line can be easily identified as a string by our system. The intercharacter distance in the string is verified to be less than 16 pixel. We also ensure that the MBRs of adjacent characters do not overlap by more than 2 pixels. If all three conditions are satisfied, we retain the candidate word region as a text string. The final output is a binary image containing the text characters that can be directly used as input to an OCR system to get the text string in ASCII and also a text file containing information about features of the character regions. Figures 3 and 4 illustrate the results of our processing of a video frames shown in Figure 1.

3.2 Interframe Analysis for Text Refinement

Intraframe processing can be followed by an interframe verification as text in videos persists over multiple consecutive frames. Text regions determined from five consecutive frames are analyzed together to add missing characters in frames and to delete incorrect regions posing as text. This interframe analysis involves examination of the similarity of text regions in terms of their positions, intensities, and shape features and aids in omitting false positive regions.

4 Highlights of Our Algorithm

Some may argue that character text regions from the input image can be obtained using classical methods such as global or local thresholding. However, local thresholding techniques fail with

images containing complex background and global thresholding techniques fail with images containing numerous text strings, each in a different position and at different gray level contrast with the background. Segmentation techniques such as region growing and split and merge tend to provide better results with complex images but they are quite slow.

Our approach removes as many non-text regions as early as possible, so that only a few candidate regions can be verified in detail for the presence of text. Each stage of our algorithm is designed to handle commonly encountered problems that arise in extracting text from video. For example, a local thresholding method, by itself can result in the retention of some non-text regions, especially in low contrast images. However, a local thresholding mechanism followed by region boundary analysis results in removing many background regions and retains only those that can strictly exhibit text-like characteristics. Only in the third stage of our algorithm, we employ strong constraints that embody the characteristics of text such as the font size, fill factor, horizontal character alignment, etc. These constraints are used to examine the attributes of the text regions, thus verifying whether it is indeed a character string and whether it forms part of a string containing two or more characters. Thus, our algorithm is designed to be general enough to handle a lot of variations in font size, style, gray level contrast, complex image backgrounds. Many systems cannot handle the presence of normal and inverse video text in the same image. By processing both the positive and negative images and combining the results, our system handles text in reverse video easily.

5 Experimental Results and System Performance

Our algorithm has been implemented in Visual C++ on a personal computer with a 133 MHz Pentium processor and 32 MB memory, running *Windows 95*. We tested our text extraction system on 12 video streams whose play-durations varied from 10 seconds to 1 minute. Table 1 provides details of some of the experimental data for which we determined the ground truth meticulously. Figures 4(c) and 5 show some results obtained by our system. We also tested

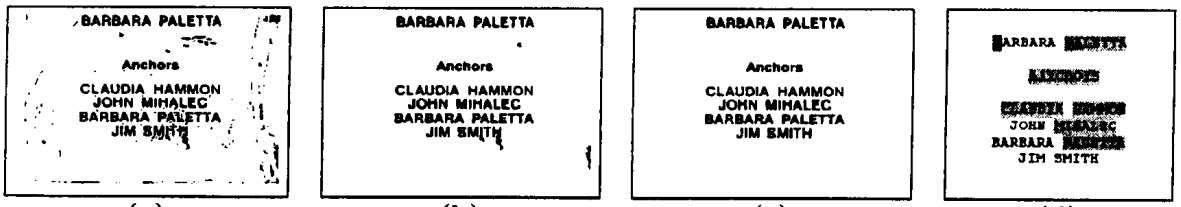

Figure 4: More results of our text extraction system: (a) boundary refined text regions; (b) non-text regions removed; (c) characters extracted using text consistency analysis; (d) output of Cunei3.0 OCR system.

Figure 5: Text extracted by our system on the frames shown in Figure 1.

Table 1: Experimental data used to assess our text extraction system.

Data source	Name of the video	Frame size	Type	Total Number of Frames
IBM	Bangkok	352 × 240	Color, Commercial	901
IBM	Arab	352 × 240	Color, Commercial	1000
IBM	News1	352 × 240	Color, News	42
WWW	SM	532 × 288	B/W, Credit Seq.	395
WWW	R1	384 × 288	Color, Commercial	259

our system on more video frames drawn from eight video streams of news stories, commercials, sitcoms, etc. The total number of frames used over 5,000. Table 2 gives a quantitative account of the performance of our system on five sequences with ground truth.

Table 2: System performance results.

Video	Frames containing text	Total number of characters in video	Number of characters. missed	% Miss rate
Bangkok	467	11800	34	0.29
Arab	441	8118	69	0.85
News1	41	862	20	2.32
SM	159	9116	244	2.68
R1	55	1705	0	0

Among two important criteria for evaluating the performance namely, *completeness* which signifies that the miss rate must be zero and *soundness* that implies that the false positive rate is zero, we are primarily interested in the former in light of our video retrieval application. False positive text can be easily corrected with human verification of the output transcripts. It can be observed from Table 2 that our system performs very well in terms of the miss rate in commercials, where the characters appear much brighter than the background. With the SM sequence, we observed that our system missed 244 characters mainly due to the dissolve nature of 12 frames containing text. This count reduces to 40 if these frames are not included

in the discussion. Our system extracted text from each frame in about 1.7 seconds on the average and this can be improved with code optimization. We conducted these experiments even without utilizing the interframe-based refinement. We anticipate that in conjunction with this refinement step, small noisy regions that remain as false positives can be eliminated.

Our current system does not allow detection of a single character present alone in a single line in an image. This is reasonable as text in video contain mainly captions and credits which are not usually single character strings. Strings such as "a" do not serve as effective keywords for video retrieval. The system does detect a single character in a line as long as it is preceded or followed by other character strings in the same line. Though we have not reported any result with images containing vertically oriented text, we believe that our algorithm can be easily extended to handle it by relaxing some of our requirements on horizontal alignment of text strings and also by processing a transformed version of the input image. Very large fonts can also be accommodated by modifying some of the system parameters. Our system handles scene text that is horizontally or diagonally oriented and whose font is within the allowed size. The system is stable as the parameters remain the same for all images.

6 Conclusions

We presented an effective and robust text extraction system for automatically obtaining text present in videos. Text extracted from video is very useful as keywords for content annotation and searching. Our scheme extracts text regions from gray scale video frames using a new and fast region segmentation algorithm and employs successive refinement of the candidate regions to retain only regions containing characters. The character outlines are enhanced using boundary analysis and text consistency criteria are used to eliminate false positive regions. The system outputs OCR-ready bitmaps and experimental results over 5,000 frames demonstrate good performance. The algorithm is general enough to handle text that appears as part of a scene as well as superimposed text, and robust enough to handle the noise and artifacts introduced by block-based MPEG encoding of digital video. We also presented a novel mechanism that exploits the temporal relation of the text appearing over multiple consecutive frames to improve the accuracy of text detection in video. Our approach can also be easily used for extracting text in still images from photographs, newspapers, and magazine advertisements. Our future research is directed towards handling multi-colored text and developing character recognition techniques specialized for video fonts.

References

[1] H. Zhang and S. W. Smoliar, "Developing power tools for video indexing and retrieval," in *Proc. SPIE Conference on storage and retrieval for image and video databases*, (San Jose,

CA), pp. 140–149, 1994.

[2] M. Christel *et al.*, "Informedia digital video library," *Communications of the Association for Computing Machinery*, vol. 38, no. 4, pp. 57–58, 1995.

[3] J. Smith and S.-F. Chang, "Visually searching the Web for content," *IEEE Multimedia*, vol. 4, pp. 12–20, 1997.

[4] Y. Tang, S. Lee, and C. Suen, "Automatic document processing: A survey," *Pattern Recognition*, vol. 29, pp. pp. 1931–1952, 1996.

[5] Y. Liu and S. Srihari, "Document image binarization on texture features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 540–544, 1997.

[6] Y. Cui and Q. Huang, "Character extraction of license plates from video," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, (Puerto Rico), pp. 502–507, June 1997.

[7] A. K. Jain and B. Yu, "Automatic text location in images and video frames," tech. rep., Michigan State University, Department of Computer Science, East Lansing, MI, 1996.

[8] V. Wu, R. Manmatha, and E. M. Riseman, "Finding text in images," in *2nd ACM Intl. Conf. on Digital Libraries*, 1997.

[9] J. Ohya, A. Shio, and S. Akamatsu, "Recognizing characters in scene images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 214–220, February 1994.

[10] S. Kannangara, E. Asbun, R. X. Browning, and E. J. Delp, "The use of nonlinear filtering in automatic video title capture," in *Proceedings of the 1997 IEEE/EURASIP Workshop on Nonlinear Signal and Image Processing*, (Mackinac Island, Michigan), September 8-10 1997.

[11] M. Smith and T. Kanade, "Video skimming and characterization through the combination of image and language understanding techniques," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, (Puerto Rico), pp. 775–781, June 1997.

[12] B.-L. Yeo and B. Liu, "Visual content highlighting via automatic extraction of embedded captions on mpeg compressed video," in *Proc. SPIE Digital Video Compression: Algorithms and Technologies*, vol. 2668, February 1996.

[13] R. Lienhart, "Automatic text recognition for video indexing," in *Proc. ACM Multimedia 96*, (Boston, MA), pp. 11–20, November 1996.

[14] J.-C. Shim and C. Dorai, "A fast and generalized region labeling algorithm," tech. rep., IBM Thomas J. Watson Research Center, Yorktown Heights, New York, 1998.

[15] T. Ridler and S. Calvard, "Picture thresholding using an interactive selection method," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, no. 8, pp. 630–632, 1978.