Text Information Extraction in Images and Video: A Survey

Keechul Jung, Kwang In Kim, Anil K. Jain

Abstract

Text data present in images and video contain useful information for automatic annotation, indexing, and structuring of images. Extraction of this information involves detection, localization, tracking, extraction, enhancement, and recognition of the text from a given image. However, variations of text due to differences in size, style, orientation, and alignment, as well as low image contrast and complex background make the problem of automatic text extraction extremely challenging. While comprehensive surveys of related problems such as face detection, document analysis, and image & video indexing can be found, the problem of text information extraction is not well surveyed. A large number of techniques have been proposed to address this problem, and the purpose of this paper is to classify and review these algorithms, discuss benchmark data and performance evaluation, and to point out promising directions for future research.

Keywords: Text information extraction, text detection, text localization, text tracking, text enhancement, OCR

1 Introduction

Content-based image indexing refers to the process of attaching labels to images based on their content. Image content can be divided into two main categories: *perceptual content* and *semantic content* [1]. Perceptual content includes attributes such as color, intensity, shape, texture, and their temporal changes, whereas semantic content means objects, events, and their relations. A number of studies on the use of relatively low-level perceptual content [2-6] for image and video indexing have already been reported. Studies on semantic image content in the form of text, face, vehicle, and human action have also attracted some recent interest [7-16]. Among them, text within an image is of particular interest as *(i)* it is very useful for describing the contents of an image; *(ii)* it can be easily extracted compared to other semantic contents, and *(iii)* it enables applications such as keyword-based image search, automatic video logging, and text-based image indexing.

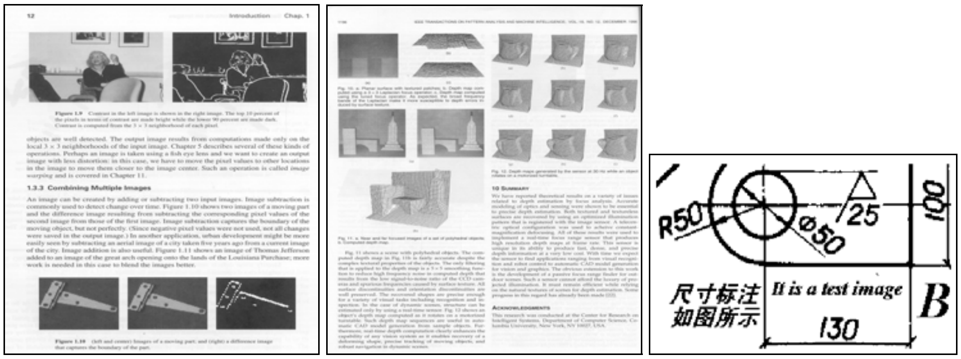
1.1 Text in images

A variety of approaches to text information extraction (TIE) from images and video have been proposed for specific applications including page segmentation [17, 18], address block location [19], license plate location [9, 20], and content-based image/video indexing [5, 21]. In spite of such extensive studies, it is still not easy to design a general-purpose TIE system. This is because there are so many possible sources of variation when extracting text from a shaded or textured background, from low-contrast or complex images, or from images having variations in font size, style, color, orientation, and alignment. These variations make the problem of automatic TIE extremely difficult.

Figures 1-4 show some examples of text in images. Page layout analysis usually deals with document images[1] (Fig. 1). Readers may refer to papers on document segmentation/analysis [17, 18] for more examples of document images. Although images acquired by scanning book covers, CD covers, or other multi-colored documents have similar characteristics as the document images (Fig. 2), they can not be directly dealt with using a conventional document image analysis technique. Accordingly, this survey distinguishes this category of images as multi-color document images from other document images. Text in video images can be further classified into caption text (Fig. 3), which is artificially overlaid on the image, or scene text (Fig. 4), which exists naturally in the image. Some researchers like to use the term ‘*graphics text*’ for scene text, and ‘*superimposed text*’ or ‘*artificial text*’ for caption text [22, 23]. It is well known that scene text is more difficult to detect and very little work has been done in this area. In contrast to caption text, scene text can have any orientation and may be distorted by the perspective projection. Moreover, it is often affected by variations in scene and camera parameters such as

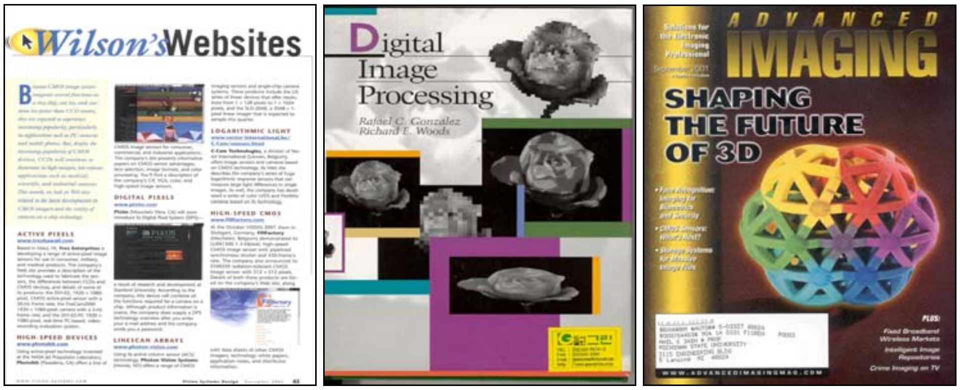
[1] The distinction between document images and other scanned images is not very clear. In this paper, we refer to images

illumination, focus, motion, etc.



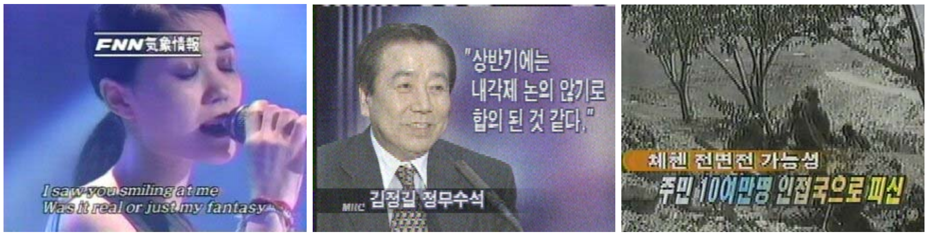
(a) (b) (c)

Fig. 1. Grayscale document images: (a) single-column text from a book, (b) two-column page from a journal (*IEEE Transactions on PAMI)*, and (c) an electrical drawing (courtesy of Lu [24]).



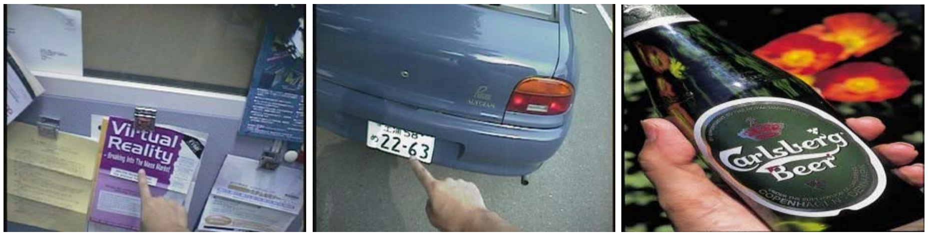
(a) (b) (c)

Fig. 2. Multi-color document images: each text line may or may not be of the same color.



(a) (b) (c)

Fig. 3. Images with caption text: (a) shows captions overlaid directly on the background. (b) and (c) contain text in frames for better contrast. (c) contains a text string that is polychrome.



with text contained in a homogeneous background as *document images*.

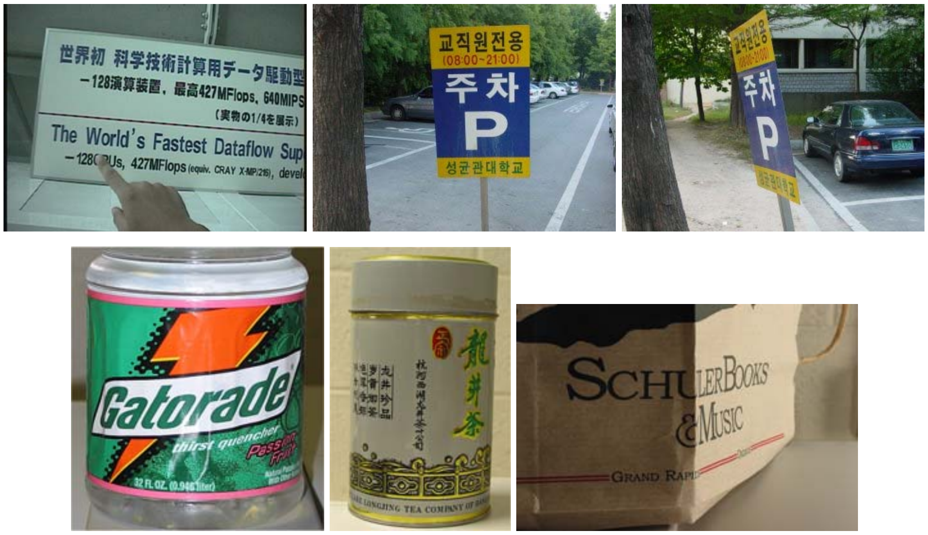


Fig. 4. Scene text images: Images with variations in skew, perspective, blur, illumination, and alignment.

Before we attempt to classify the various techniques used in TIE, it is important to define the commonly used terms and summarize the characteristics[2] of text that can be used for TIE algorithms. Table 1 shows a list of properties that have been utilized in recently published algorithms [25-30]. Text in images can exhibit many variations with respect to the following properties:

1. Geometry:
   - Size: Although the text size can vary a lot, assumptions can be made depending on the application domain.
   - Alignment: The characters in the caption text appear in clusters and usually lie horizontally, although sometimes they can appear as non-planar texts as a result of special effects. This does not apply to scene text, which can have various perspective distortions. Scene text can be aligned in any direction and can have geometric distortions (Fig. 4).
   - Inter-character distance: characters in a text line have a uniform distance between them.

2. Color: The characters in a text line tend to have the same or similar colors. This property makes it possible to use a connected component-based approach for text detection. Most of the research reported till date has concentrated on finding ‘text strings of a single color (monochrome)’. However, video images and other complex color documents can contain ‘text strings with more than two colors (polychrome)’ for effective visualization, i.e., different colors within one word.

3. Motion: The same characters usually exist in consecutive frames in a video with or without movement. This property is

[2] All these properties may not be important to every application.

used in text tracking and enhancement. Caption text usually moves in a uniform way: horizontally or vertically. Scene text can have arbitrary motion due to camera or object movement.

4. Edge: Most caption and scene text are designed to be easily read, thereby resulting in strong edges at the boundaries of text and background.

5. Compression: Many digital images are recorded, transferred, and processed in a compressed format. Thus, a faster TIE system can be achieved if one can extract text without decompression.

Table 1. Properties of text in images

Property		Variants or sub-classes
Geometry	Size	Regularity in size of text
	Alignment	Horizontal/vertical
		Straight line with skew (implies vertical direction)
		Curves
		3D perspective distortion
	Inter-character distance	Aggregation of characters with uniform distance
Color		Gray
		Color (monochrome, polychrome)
Motion		Static
		Linear movement
		2D rigid constrained movement
		3D rigid constrained movement
		Free movement
Edge		Strong edges (contrast) at text boundaries
Compression		Un-compressed image
		JPEG, MPEG-compressed image

1.2 What is Text Information Extraction (TIE)?

The problem of Text Information Extraction needs to be defined more precisely before proceeding further. A TIE system receives an input in the form of a still image or a sequence of images. The images can be in gray scale or color, compressed or un-compressed, and the text in the images may or may not move. The TIE problem can be divided into the following sub-problems: (i) detection, (ii) localization, (iii) tracking, (iv) extraction and enhancement, and (v) recognition (OCR) (Fig. 5).

Text detection, localization, and extraction are often used interchangeably in the literature. However, in this paper, we differentiate between these terms. The terminology used in this paper is mainly based on the definitions given by Antani et al. [28]. Text detection refers to the determination of the presence of text in a given frame (normally text detection is used for a sequence of images). Text localization is the process of determining the location of text in the image and generating bounding boxes around the text. Text tracking is performed to reduce the processing time for text localization and to maintain the integrity of position across adjacent frames. Although the precise location of text in an image can be indicated by bounding boxes, the text still needs to be segmented from the background to facilitate its recognition. This means that the extracted text image has to be converted to a binary image and enhanced before it is fed into an OCR engine. Text extraction is the stage where the text components are segmented from the background. Enhancement of the extracted text components is required because the text region usually has low-resolution and is prone to noise. Thereafter, the extracted text images can be transformed into plain text using OCR technology.

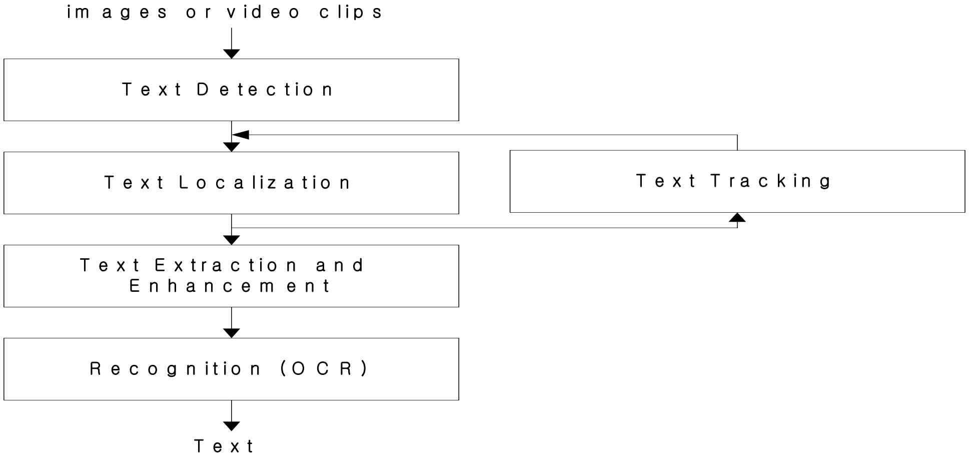


Fig. 5. Architecture of a TIE system.

1.3 Scope and organization

This paper presents a comprehensive survey of TIE from images and videos. Page layout analysis is similar to text localization in images. However, most page layout analysis methods assume the characters to be black with a high contrast on a homogeneous background. Tang et al. [18] presented a survey of page layout analysis, and Jain and Yu [31] provided a brief survey of page decomposition techniques. In practice, text in images can have any color and be superimposed on a complex background. Although a few TIE surveys have already been published, they lack details on individual approaches and are not clearly organized [22, 26]. We organize the TIE algorithms into several categories according to their main idea and discuss their pros and cons.

Section 2 reviews the various sub-stages of TIE and introduces approaches for text detection, localization, tracking, extraction, and enhancement. We also point out the ability of the individual techniques to deal with color, scene text, compressed images, etc. The important issue of performance evaluation is discussed in Section 3, along with sample public test data sets and a review of evaluation methods. Section 4 gives an overview of the application domains for TIE in image processing and computer vision. The final conclusions are presented in Section 5.

2 Text Information Extraction

As described in the previous section, TIE can be divided into five sub-stages: detection, localization, tracking, extraction and enhancement, and recognition. Each sub-stage will be reviewed in this section, except for recognition. A chronological listing of some of the published work on TIE is presented in Table 2.

2.1 Text Detection

In this stage, since there is no prior information on whether or not the input image contains any text, the existence or non-existence of text in the image must be determined. Several approaches assume that certain types of video frame or image contain text. This is a common assumption for scanned images (e.g., compact disk cases or book covers). However, in the case of video, the number of frames containing text is much smaller than the number of frames without text.

The text detection stage seeks to detect the *presence* of text in a given image. Kim [1] selected a frame from shots detected by a scene-change detection method as a candidate containing text. Although Kim's scene-change detection method is not described in detail in his paper, he does mention that very low threshold values are needed for scene-change detection because the portion occupied by a text region relative to the whole image is usually small. This approach is very sensitive to scene-change detection. Text-frame selection is performed at an interval of 2 seconds for caption text in the detected scene frames. This can be a simple and efficient solution for video indexing applications that only need key words from video clips, rather than the entire text.

Smith and Kanade [7] defined a scene-change based on the difference between two consecutive frames and then used this scene-change information for text detection. They achieved an accuracy of 90% in scene-change detection. Gargi et al. [29, 32] performed text detection using the assumption that the number of intracoded blocks in P- and B- frames of an MPEG compressed video increases, when a text caption appears. Lim et al. [33] made a simple assumption that text usually has a higher intensity than the background. They counted the number of pixels that are lighter than a predefined threshold value and exhibited a significant color difference relative to their neighborhood, and regarded a frame with a large number of such pixels as a text frame. This method is extremely simple and fast. However, problems can occur with color-reversed text.

Table 2. A brief survey of TIE

Author	Year	Approach	Features
Ohya et al. [34]	1994	Adaptive thresholding and relaxation operations	Color, scene text (train, signboard, skew and curved), localization and recognition
Lee and Kankanhalli [35]	1995	Coarse search using edge information, followed by connected component (CC) generation	Scene text (cargo container), localization and recognition
Smith and Kanade [7]	1995	3×3 filter seeking vertical edges	Caption text, localization
Zhong et al. [36]	1995	CC-based method after color reduction, local spatial variance-based method, and hybrid method	Scene text (CD covers), localization
Yeo and Liu [56]	1996	Localization based on large inter-frame difference in MPEG compressed image	Caption text, localization
Shim et al. [21]	1998	Gray level difference between pairs of pixels	Caption text, localization
Jain and Yu [38]	1998	CC-based method after multi-valued color image decomposition	Color (book cover, Web image, video frame), localization
Sato et al. [11]	1998	Smith and Kanade's localization method and recognition-based character extraction	Recognition
Chun et al. [54]	1999	Filtering using neural network after FFT	Caption text, localization
Antani et al. [28]	1999	Multiple algorithms in functional parallelism	Scene text, recognition
Messelodi and Modena [39]	1999	CC generation, followed by text line selection using divisive hierarchical clustering procedure	Scene images (book covers, slanted) localization
Wu et al. [45]	1999	Localization based on multi-scale texture segmentation	Video and scene images (newspaper, advertisement) recognition
Hasan and Karam [42]	2000	Morphological approach	Scene text, localization
Li et al. [51]	2000	Wavelet-based feature extraction and neural network for texture analysis	Scene text (slanted), localization, enhancement, and tracking
Lim et al. [33]	2000	Text detection and localization using DCT coefficient and macroblock type information	Caption text, MPEG compressed video, localization
Zhong et al. [27]	2000	Texture analysis in DCT compressed domain	Caption text, JPEG and I-frames of MPEG, localization
Jung [49]	2001	Gabor filter-like multi-layer perceptron for texture analysis	Color, caption text, localization
Chen et al. [43]	2001	Text detection in edge-enhanced image	Caption text, localization and recognition
Strouthopoulos et al. [16]	2002	Page layout analysis after adaptive color reduction	Color document image, localization

In our opinion, researchers have not given much attention to the text detection stage, mainly because most applications of TIE are related to scanned images, such as book covers, compact disk cases, postal envelopes, etc. which are supposed to include text. However, when dealing with video data, the text detection stage is indispensable to the overall system for reducing the time complexity.

If a text localization module (see Section 2.2) can operate in real-time; it can also be used for detecting the presence of text. Zhong et al. [27] and Antani et al. [28] performed text localization on compressed images, which resulted in a faster performance. Therefore, their text localizers could also be used for text detection. The text detection stage is closely related to the text localization and text tracking stages, which will be discussed in Sections 2.2 and 2.3, respectively.

2.2 Text Localization

According to the features utilized, text localization methods can be categorized into two types: region-based and texture-based. Section 2.2.3 deals with text localization in compressed domain. Methods that overlap the two approaches or are difficult to categorize are described in section 2.2.4. For reference, the performance measures (computation time and localization rate) are presented for each approach based on experimental results whenever available.

2.2.1 Region-based methods

Region-based methods use the properties of the color or gray-scale in a text region or their differences with the corresponding properties of the background. These methods can be further divided into two sub-approaches: connected component (CC)-based and edge-based. These two approaches work in a bottom-up fashion; by identifying sub-structures, such as CCs or edges, and then merging these sub-structures to mark bounding boxes for text. Note that some approaches use a combination of both CC-based and edge-based methods.

2.2.1.1 CC-based Methods

CC-based methods use a bottom-up approach by grouping small components into successively larger components until all regions are identified in the image. A geometrical analysis is needed to merge the text components using the spatial arrangement of the components so as to filter out non-text components and mark the boundaries of the text regions.

Ohya et al. [34] presented a four-stage method: (i) binarization based on local thresholding, (ii) tentative character component detection using gray-level difference, (iii) character recognition for calculating the similarities between the character candidates and the standard patterns stored in a database, and (iv) relaxation operation to update the similarities. They were able to extract and recognize characters, including multi-segment characters, under varying illuminating conditions, sizes, positions, and fonts when dealing with scene text images, such as freight train, signboard, etc. However, binary segmentation is inappropriate for video documents, which can have several objects with different gray levels and high levels of noise and variations in illumination. Furthermore, this approach places several restrictions related to text alignment, such as upright and not connected, as well as the color of the text (monochrome). Based on experiments involving 100 images, their recall rate of text localization was 85.4% and the character recognition rate was 66.5%.

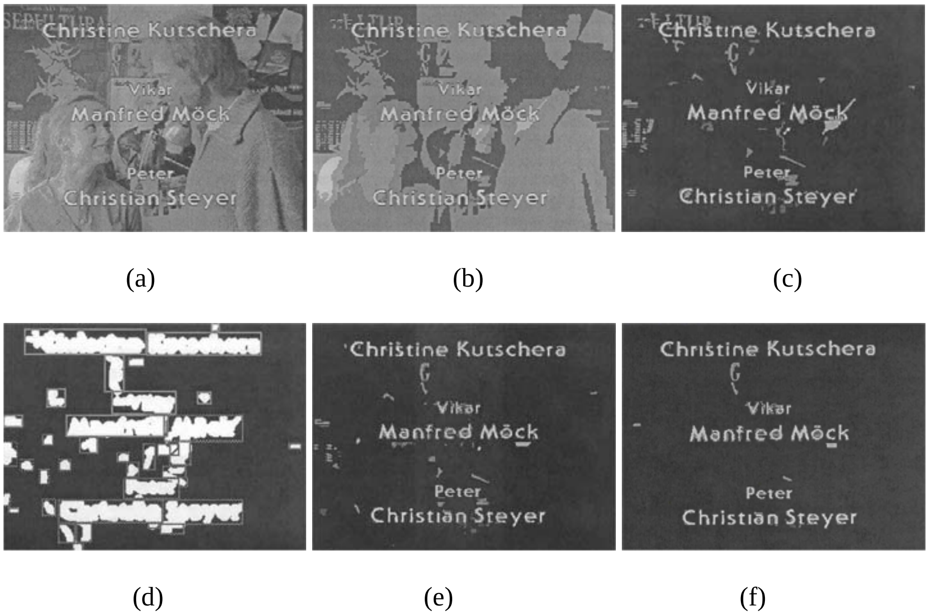
Lee and Kankanhalli [35] applied a CC-based method to the detection and recognition of text on cargo containers, which can have uneven lighting conditions and characters with different sizes and shapes. Edge information is used for a coarse search prior to the CC generation. The difference between adjacent pixels is used to determine the boundaries of potential characters after quantizing an input image. Local threshold values are then selected for each text candidate, based on the pixels on the boundaries. These potential characters are used to generate CCs with the same gray-level. Thereafter, several heuristics are used to filter out non-text components based on aspect ratio, contrast histogram, and run-length measurement. Despite their claims that the method could be effectively used in other domains, experimental results were only presented for cargo container images.

Zhong et al. [36] used a CC-based method, which uses color reduction. They quantize the color space using the peaks in a color histogram in the RGB color space. This is based on the assumption that the text regions cluster together in this color space and occupy a significant portion of an image. Each text component goes through a filtering stage using a number of heuristics, such as area, diameter, and spatial alignment. The performance of this system was evaluated using CD images and book cover images.

Kim [1] segments an image using color clustering in a color histogram in the RGB space. Non-text components, such as long horizontal lines and image boundaries, are eliminated. Then, horizontal text lines and text segments are extracted based on an iterative projection profile analysis. In the post-processing stage, these text segments are merged based on heuristics. Since several threshold values need to be determined empirically, this approach is not suitable as a general-purpose text localizer. Experiments were performed with 50 video images, including various character sizes and styles, and a localization rate of 87% was reported.

Shim et al. [21] used the homogeneity of intensity of text regions in images. Pixels with similar gray levels are merged into a group. After removing significantly large regions by regarding them as background, text regions are sharpened by performing a region boundary analysis based on the gray level contrast. The candidate regions are then subjected to verification using size, area, fill factor, and contrast. Neighboring text regions are examined to extract any text strings. The average processing time was 1.7 seconds per frame on a 133 MHz Pentium processor and the miss rate ranged from 0.29% to 2.68% depending on the video stream.

Lienhart et al. [30, 37] regard text regions as CCs with the same or similar color and size, and apply motion analysis to enhance the text extraction results for a video sequence. The input image is segmented based on the monochromatic nature of the text components using a split-and-merge algorithm. Segments that are too small and too large are filtered out. After dilation, motion information and contrast analysis are used to enhance the extracted results. A block-matching algorithm using the mean absolute difference criterion is employed to estimate the motion. Blocks missed during tracking are discarded. Their primary focus is on caption text, such as pre-title sequences, credit titles, and closing sequences, which exhibit a higher contrast with the background. This makes it easy to use the contrast difference between the boundary of the detected components and their background in the filtering stage. Finally, a geometric analysis, including the width, height, and aspect ratio, is used to filter out any non-text components. Based on experiments using 2247 frames, their algorithm extracted 86% to 100% of all the caption text. Fig. 6 shows an example of their text extraction process.



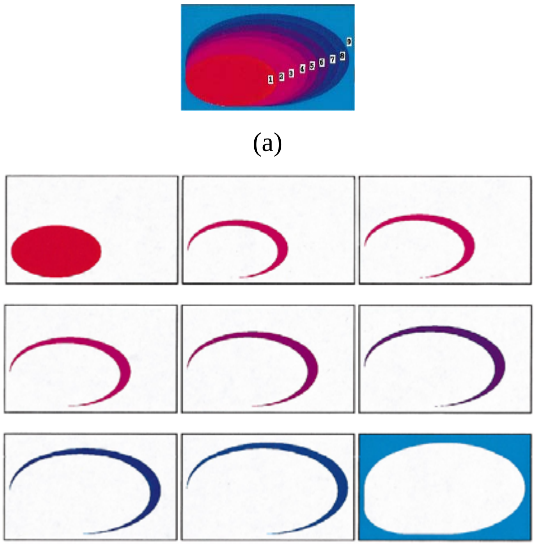
(a) (b) (c)

(d) (e) (f)

Fig. 6. Intermediate stages of processing in the method by Lienhart et al. (a) original video frame; (b) image segmentation using split-and-merge algorithm; (c) after size restriction; (d) after binarization and dilation; (e) after motion analysis; and (f) after contrast analysis and aspect ratio restriction (courtesy of Lienhart et al. [30, 37]).

Jain and Yu [38] apply a CC-based method after preprocessing, which includes bit dropping, color clustering, multi-valued image decomposition, and foreground image generation. A 24-bit color image is bit-dropped to a 6-bit image, and then quantized by a color-clustering algorithm. After the input image is decomposed into multiple foreground images, each foreground image goes through the same text localization stage. Figure 7 shows an example of the multi-valued image decomposition. CCs are generated in parallel for all the foreground images using a block adjacency graph. The localized text components in the individual foreground images are then merged into one output image. The algorithm was tested with various types of images such as binary document images, web images, scanned color images, and video images. The processing time reported for a Sun UltraSPARC I system (167MHz) with a 64MB memory was less than 0.4 seconds for

769×537 color images. The algorithm extracts only horizontal and vertical text, and not skewed text. The authors point out that their algorithm may not work well when the color histogram is sparse.



(a)

(b)

Fig. 7. A multi-colored image and its element images: (a) color input image; (b) nine element images (courtesy of Jain and Yu [38]).

Messelodi and Modena’s method [39] consists of three sequential stages: (i) extraction of elementary objects, (ii) filtering of objects, and (iii) text line selections. Preprocessing, such as noise reduction, deblurring, contrast enhancement, quantization, etc., is also performed. After the preprocessing, intensity normalization, image binarization, and CC generation are performed. Various filters based on several *internal features*, including area, relative size, aspect ratio, density, and contrast are then applied to eliminate non-text components. As noted by the authors, the filter selection is heavily dependent on the application and threshold values. Finally, the text line selection stage starts from a single region and recursively expands, until a termination criterion is satisfied, using *external features* such as closeness, alignment, and comparable height. This method can deal with various book cover images containing different sizes, fonts, and styles of text. In addition, it can be used for text lines with different amounts of skew in the same image. Based on experiments using 100 book cover images, a 91.2% localization rate was achieved.

Kim et al. [40] used cluster-based templates for filtering out non-character components for multi-segment characters to alleviate the difficulty in defining heuristics for filtering out non-text components. A similar approach was also reported by Ohya et al. [34]. Cluster-based templates are used along with geometrical information, such as size, area, and alignment. They are constructed using a K-means clustering algorithm from actual text images.

Hase et al. [84] proposed a CC-based method for color documents. They assume that every character is printed in a single color. This is a common assumption in text localization algorithms. Pixel values are translated into L*a*b* color space and representative colors are selected based on the color histogram. The image is then divided into several binary images and string extraction by multi-stage relaxation, which was presented by the same authors previously [85], is performed on each binary image. After merging all results from the individual binary images, character strings are selected by their likelihoods, using conflict resolution rules. The likelihood of a character string is defined using the alignment of characters in a string, mean area ratio of black pixels in elements to its rectangle, and the standard deviation of the line widths of initial elements. Two types of conflicts (inclusion and overlap) between character strings are filtered using tree representation. Contrary to other text localization techniques, they pay more attention to the filtering stage. As a result, they can deal with shadowed and curved strings. However, as the authors described, the likelihood of a character string is not easy to define accurately, which results in missed text or false alarms.

Due to their relatively simple implementation, CC-based methods are widely used. Nearly all CC-based methods have four processing stages: (i) preprocessing, such as color clustering and noise reduction, (ii) CC generation, (iii) filtering out non-text components, and (iv) component grouping. A CC-based method could segment a character into multiple CCs, especially in the cases of polychrome text strings and low-resolution and noisy video images [1]. Further, the performance of a CC-based method is severely affected by *component grouping*, such as a projection profile analysis or text line selection. In addition, several threshold values are needed to filter out the non-text components, and these threshold values are dependent on the image/video database.

2.2.1.2 **Edge-based methods**

Among the several textual properties in an image, edge-based methods focus on the 'high contrast between the text and the background'. The edges of the text boundary are identified and merged, and then several heuristics are used to filter out the non-text regions. Usually, an edge filter (e.g., a Canny operator) is used for the edge detection, and a smoothing operation or a morphological operator is used for the merging stage.

Smith and Kanade [7] apply a 3×3 horizontal differential filter to an input image and perform thresholding to find vertical edges. After a smoothing operation, that is used to eliminate small edges, adjacent edges are connected and a bounding box is computed. Then heuristics, including the aspect ratio, fill factor, and size of each bounding box are applied to filter out non-text regions. Finally, the intensity histogram of each cluster is examined to filter out clusters that have similar texture and shape characteristics.

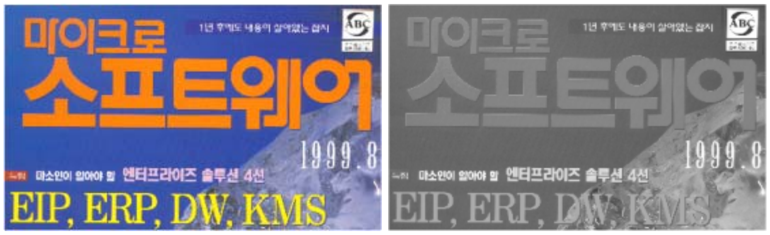
The main difference between the method by Sato et al. [11] and other edge-based methods is their use of recognition-based character segmentation. They use character recognition results to make decisions on the segmentation and positions of individual characters, thereby improving the accuracy of character segmentation [41]. When combined with Smith and

Kanade’s text localization method, the processing time from detection to recognition was less than 0.8 seconds for a 352×242 image. A more detailed description is given in Section 2.3.

Hasan and Karam [42] presented a morphological approach for text extraction. The RGB components of a color input image are combined to give an intensity image Y as follows:

$$Y = 0.299\,R + 0.587\,G + 0.114\,B\,,$$

where R, G, and B are the red, green, and blue components, respectively. Although this approach is simple and many researchers have adopted it to deal with color images, it has difficulties dealing with objects that have similar gray-scale values, yet different colors in a color space. Fig. 8(a) shows a color image and 8(b) shows the corresponding gray-scale image. Some text regions that are prominent in the color image are difficult to detect in the gray-level image. After the color conversion, the edges are identified using a morphological gradient operator. The resulting edge image is then thresholded to obtain a binary edge image. Adaptive thresholding is performed for each candidate region in the intensity image, which is less sensitive to illumination conditions and reflections. Edges that are spatially close are grouped by dilation to form candidate regions, while small components are removed by erosion. Non-text components are filtered out using size, thickness, aspect ratio, and gray-level homogeneity. This method seems to be robust to noise, as shown by experiments with noisy images. The method is insensitive to skew and text orientation, and curved text strings can also be extracted. However, the authors compare their method with other methods on only three images.



(a) (b)

Fig. 8. Difference between color image and its gray-scale image: (a) color image, (b) corresponding gray scale image.

Chen et al. [43] used the Canny operator to detect edges in an image. Only one edge point in a small window is used in the estimation of scale and orientation to reduce the computational complexity. The edges of the text are then enhanced using this scale information. Morphological dilation is performed to connect the edges into clusters. Some heuristic knowledge, such as the horizontal-vertical aspect ratio and height, is used to filter out non-text clusters. Two groups of Gabor-type asymmetric filters are used for generating input features for scale estimation: an edge-form filter and a stripe-form filter and a neural network are used to estimate the scale of the edge pixels based on these filters’ outputs. The edge information is then enhanced at an appropriate scale. As such, this results in the elimination or blurring of structures that do not have the specified scales. The text localization is applied to the enhanced image. The authors used a commercial OCR package (TypeReader OCR package [http://www.expervision.com]) after size normalization of individual characters into 128 pixels using bilinear interpolation.

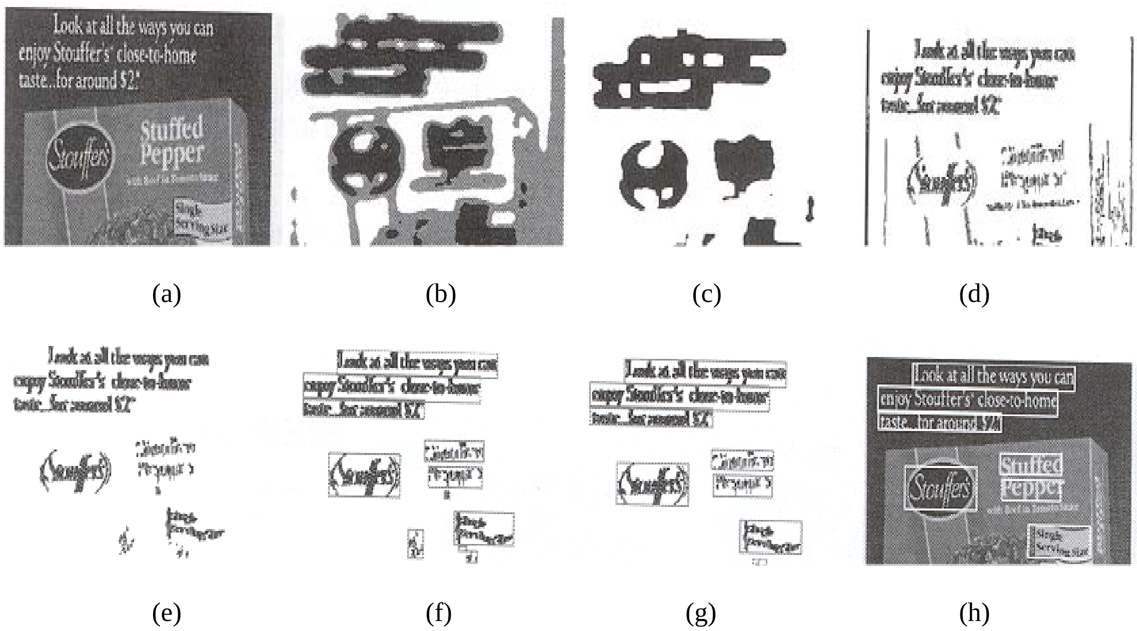
2.2.2 Texture-based methods

Texture-based methods use the observation that text in images have distinct textural properties that distinguish them from the background. The techniques based on Gabor filters, Wavelet, FFT, spatial variance, etc. can be used to detect the textural properties of a text region in an image.

Zhong et al. [36] use the local spatial variations in a gray-scale image to locate text regions with a high variance. They utilize a horizontal window of size 1×21 to compute the spatial variance for pixels in a local neighborhood. Then the horizontal edges in an image are identified using a Canny edge detector, and the small edge components are merged into longer lines. From this edge image, edges with opposite directions are paired into the lower and upper boundaries of a text line. However, this approach can only detect horizontal components with a large variation compared to the background. A 6.6 second processing time was achieved with a 256×256 image on a SPARC station 20. In another paper, Zhong [27] presents a similar texture-based method for JPEG images and I-frames of MPEG compressed videos. This is discussed in detail in Section 2.2.3.

A similar method has been applied to vehicle license plate localization by Park et al. [44]. In this case, the horizontal variance of the text is also used for license plate localization. The only difference lies in the use of a time delay neural network (TDNN) as a texture discriminator in the HSI color space. Two TDNNs are used as horizontal and vertical filters. Each neural network receives HSI color values for a small window of an image as input and decides whether or not the window contains a license plate number. After combining the two filtered images, bounding boxes for license plates are located based on projection profile analysis.

In contrast, Wu et al. [45, 46] segment an input image using a multi-scale texture segmentation scheme. Potential text regions are detected based on nine second-order Gaussian derivatives. A non-linear transformation is applied to each filtered image. The local energy estimates, computed at each pixel using the output of the nonlinear transformation, are then clustered using the K-means algorithm. This process is referred to as *texture segmentation*. Next, the *chip generation* stage is initiated, which consists of 5 steps: (i) stroke generation, (ii) stroke filtering, (iii) stroke aggregation, (iv) chip filtering, and (v) chip extension. These texture segmentation and chip generation stages are performed at multiple scales to detect text with a wide range of sizes, and then mapped back onto the original image. The test set consisted of 48 different images, including video frames, newspapers, magazines, envelopes, etc. The process took 10 seconds for a 320×240 sized image on a 200 MHz Pentium Pro PC with 128 Mbytes of memory. Although insensitive to the image resolution due to its multi-scale texture discrimination approach, this method tends to miss very small text. The system was evaluated at two levels: character-level and OCR-level detection. The localization rate for characters larger than 10 pixels was higher than 90%, while the false alarm rate was 5.6 %. A recognition rate of 94% was achieved for extracted characters with the OmniPage Pro 8.0 recognizer, and 83.8% for all the characters. Fig. 9 shows examples of the output at the intermediate stages.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 9. Example of texture segmentation: (a) sub-image of input image, (b) clustering, (c) text region after morphological operation (shown in black), (d) stroke generation, (e) stroke filtering, (f) stroke aggregation, (g) chip filtering and extension, and (h) text localization (courtesy of Wu, Manmatha, and Riseman [45, 46]).

Sin et al. [81] use frequency features such as the number of edge pixels in horizontal and vertical directions and Fourier spectrum to detect text regions in real scene images. Based on the assumption that many text regions are on a rectangular background, rectangle search is then performed by detecting edges, followed by the Hough transform. However, it is not clear how these three stages are merged to generate the final result.

Mao et al. [82] propose a texture-based text localization method using Wavelet transform. Harr Wavelet decomposition is used to define local energy variations in the image at several scales. Binary image, which is acquired after thresholding the local energy variation, is analyzed by connected component-based filtering using geometric attributes such as size and aspect ratio. All the text regions, which are detected at several scales, are merged to give the final result.

Since the utilization of texture information for text localization is also sensitive to the character font size and style, it is difficult to manually generate a texture filter set for each possible situation. Therefore, to alleviate the burden of manually designing texture filters, several learning-based methods have been proposed for the automatic generation of a filter set. Jain and Zhong [17] and others [47, 48] use a learning-based texture discrimination method to separate text, graphics, and halftone image regions in documents. Jung [49] and Jeong et al. [50] also use a similar approach for TIE in complex color images, where a neural network is employed to train a set of texture discrimination masks that minimize the classification error for the two texture classes: text regions and non-text regions. The textural properties of the text regions are analyzed using the R, G, and B color bands. The input image is scanned by the neural network, which receives the color values from the neighborhood of a given pixel as input. Li's method [51] has no explicit feature extraction stage, unlike other approaches such as wavelets, FFT, and Gabor-based feature extraction. An arbitration neural network can combine the outputs from the three color bands into a single decision about the presence of text. A box generation algorithm, based on projection profile analysis is then applied to the final output of the arbitration neural network. In experiments using 950 320×240 sized images, a 92.2% localization rate was achieved with an 11.3 second processing time. Jung et al. [83] applied a simple extension of their text localization method for wide text strings such as banners. They used a low-resolution camera as the input device and the text localization stage was performed after generating a mosaic image using their real-time mosaicing algorithm.

Kim et al. [52] used Support Vector Machines (SVMs) for analyzing the textural properties of text in images. The input configuration is the same as that in Jain and Zhong [47]. SVMs work well even in this high-dimensional space and can incorporate a feature extractor within their architecture. After texture classification using an SVM, a profile analysis is performed to extract text lines.

A learning-based method was also employed by Li et al. [51, 53] for localizing and tracking text in video, where a small window (typically 16×16) is applied to scan an image. Each window is then classified as either text or non-text using a neural network after extracting feature vectors. The mean value and the second and third order central moments of the decomposed sub-band images, computed using wavelets, are used as the features. The neural network is operated on a pyramid of the input image with multiple resolutions. A tracking process is performed to reduce the overall processing time and stabilize the localization results based on a pure translational model and the sum of the squared difference is used as a matching metric. After tracking, contour-based text stabilization is used to deal with more complex motions. Various kinds of video images were used for experiments, including movie credits, news, sports commercials, and videoconferences. They included caption and scene texts with multiple font sizes and styles. The localization procedure required about 1 second to process a 352×240 frame on a Sun Ultra Workstation. The tracking time for one text block was about 0.2 seconds per frame. A precision rate of 62% and recall rate of 88% was achieved for text detection and a 91% precision rate and 92.8% recall rate for text localization. This method was subsequently enhanced for skewed text [53] as follows. The classification result from the neural network is a binary image, with an output of 1 indicating text and 0 indicating non-text. Connected components are generated for the binary image and skewed text is identified using the 2-D moments of the components. The moment-based method for estimating skewed text is simple and works well for elongated text lines.

Chun et al. [54] used a combination of FFT and neural network. The FFT is computed for overlapped line segments of 1×64 pixels to reduce the processing time. The output of each line segment consists of 32 features. A feed-forward neural network with one hidden-layer is used for pixel discrimination, using the 32-dimensional feature vector. Noise elimination and labeling operations are performed on the neural network's output image. Although the authors claim that their system can be used in real-time, the actual processing times were not reported.

The problem with traditional texture-based methods is their computational complexity in the texture classification stage, which accounts for most of the processing time. In particular, texture-based filtering methods require an exhaustive scan of the input image to detect and localize text regions. This makes the convolution operation computationally expensive. To tackle this problem, Li et al. [51] classified pixels at regular intervals and interpolated the pixels located between the classified pixels. However, this still does not eliminate the unnecessary texture analysis of non-text regions and merely trades precision for speed. Jung et al. [55] adopt a mean shift algorithm as a mechanism for automatically selecting regions of interest (ROIs), thereby avoiding a time-consuming texture analysis of the entire image. By embedding a texture analyzer into the mean shift, ROIs related to possible text regions are first selected based on a coarse level of classification (sub-sampled classification of image pixels). Only those pixels within the ROIs are then classified at a finer level, which significantly reduces the processing time when the text size does not dominate the image size.

2.2.3 Text Extraction in Compressed Domain

Based on the idea that most digital images and videos are usually stored, processed, and transmitted in a compressed form, TIE methods that directly operate on images in MPEG or JPEG compressed formats have recently been presented. These methods only require a small amount of decoding, thereby resulting in a faster algorithm. Moreover, the DCT coefficients and motion vectors in an MPEG video are also useful in text detection [27].

Yeo and Liu [56] proposed a method for caption text localization in a reduced resolution image that can be easily reconstructed from compressed video. The reduced resolution images are reconstructed using the DC and AC components from MPEG videos. However, the image resolution is reduced by a factor of 16 or 64, which results in a lower localization rate. Since the text regions are localized based on a large interframe difference, only abrupt frame sequence changes can be detected. This results in missed scrolling titles. The method is also limited by the assumption that text only appears in pre-defined regions of the frame.

Gargi et al. [32] used a four-stage approach for TIE – detection, localization, segmentation, and recognition. They perform text detection in a compressed domain, yet only use the number of intracoded blocks in P- and B-frames, without the I-frames of MPEG video sequences. This is based on the assumption that when captions appear or disappear, corresponding blocks are usually intracoded. However, this method is also vulnerable to abrupt scene changes or motion.

Lim et al. [33] described a method using DCT coefficients and macroblock information in an MPEG compressed video. Their method consists of three stages: text frame detection, text region extraction, and character extraction. First, a DC image is constructed from an I-frame. When the number of pixels satisfying the intensity and neighboring pixel difference conditions is larger than a given threshold; the corresponding frame is regarded as a text frame. Candidate text blocks are extracted using the sum of the edge strengths based on the AC coefficients. The background regions are filtered out using a histogram analysis and the macroblock information. Finally, the extracted text regions are decompressed for character extraction.

Zhong et al. [27] presented a method for localizing captions in JPEG images and I-frames of MPEG compressed videos. They use DCT coefficients to capture textural properties, such as the directionality and periodicity of local image blocks. The results are then refined using morphological operations and connected component analysis. Their algorithm, as described by the authors, is very fast (0.006 seconds to process a 240 × 350 image) and has a recall rate of 99.17% and false alarm rate of 1.87%. However as each *unit block* is determined as text or non-text, precise localization results could not be generated.

2.2.4 Other Approaches

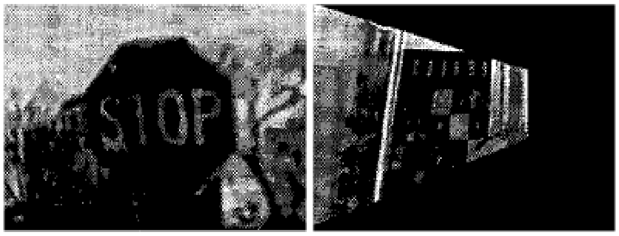
Binarization techniques, which use global, local, or adaptive thresholding, are the simplest methods for text localization. These methods are widely used for document image segmentation, as these images usually include black characters on a white background, thereby enabling successful segmentation based on thresholding. This approach has been adopted for many specific applications such as address location on postal mail, courtesy amount on checks, etc., due to its simplicity in implementation [26].

Due to the large number of possible variations in text in different types of applications and the limitations of any single approach to deal with such variations, some researchers have developed hybrid approaches. Zhong et al. [36] fused the connected component (CC)-based approach with the texture-based approach. Their CC-based method does not perform well when characters are merged or not well separated from the background. Also the drawback of their texture-based method is that characters, which extend below the baseline or above other characters, are segmented into two components. In the hybrid scheme, the CC-based method is applied after the bounding boxes have been localized using the texture-based method, and characters extending beyond the bounding boxes are filled in. However, the authors do not provide any quantitative analysis regarding the performance enhancement when using this hybrid approach.

Antani et al. [23, 28] proposed a multi-pronged approach for text detection, localization, tracking, extraction, and recognition. They used several different methods to minimize the risk of failure. The result of each detection and localization method is a set of bounding boxes surrounding the text regions. The results from the individual methods are then merged to produce the final bounding boxes. They utilize algorithms by Gargi et al. [29, 32] (See section 2.2.3), Chaddha et al. [57], Schaar-Mitrea and de With [58], and LeBourgeois [59]. Chaddha et al. use texture energy to classify 8×8 pixel blocks into text or non-text blocks. The sum of the absolute values of a subset of DCT coefficients from MPEG data is thresholded to categorize a block as text or non-text. Schaar-Mitrea and de With's algorithm [58] was originally developed for the classification of graphics and video. Antani et al. modified this to classify 4×4 blocks as text or non-text. The number of pixels in a block with a similar color is compared with threshold values to classify the block. LeBourgeois [59] used a threshold value based on the sum of the gradients of each pixel's neighbors. Antani's effort to utilize and merge several traditional approaches would seem to produce a reasonable performance for general image documents. However, the limitation of this approach is in the merging strategy, which merges the bounding boxes from each algorithm, spatially, and merges the boxes from consecutive frames, temporally. Later, in his thesis, Antani [23] gave more consideration to the merging strategy. He tried to merge several text localizers' output with intersection, union, majority vote, weighted vote, and supervised classifier fusion.

Gandhi et al. [60] use a planar motion model for scene text localization. Motion model parameters are estimated using a gradient-based method, followed by multiple-motion segmentation, based on the assumption that scene text lies on a planar surface. The multiple-motion segmentation is performed using the split and merge technique, interactively. The motion model parameters are then used to compute the structure and motion parameters. Perspective correction is performed using the estimated plane normals, resulting in a better OCR performance (See Figure 10). However, since only motion information is used for text segmentation, the algorithm requires more processing time. The use of additional information, such as texture or tonal information, could enhance the results.

(a)



(b)

Fig. 10. Results of perspective correction: (a) input images, (b) corrected images (courtesy of Gandhi et al. [60]).

A simplified version of perspective correction by Gandhi et al. [60] has been used for license plate localization by Kim et al. [20]. After the corners of a license plate are detected, the perspective distortion is corrected using image warping, which maps an arbitrary quadrilateral onto a rectangle.

L'assainato et al. [61] used the vanishing point information to recover the 3D information. The vanishing point is used in many computer vision applications to extract meaningful information from a scene image. In this case, it is assumed that the characters are usually aligned vertically or horizontally. From the detected line segments, those segments near or equal to vertical or horizontal directions pointing to a vanishing point are maintained. The final text regions are extracted using region compactness of the initial regions, which are detected using vanishing points.

Strouthopoulos et al. [16] presented a method using page layout analysis (PLA) after adaptive color reduction. Adaptive tree clustering procedure using principal component analysis and self-organized feature map is used to achieve color reduction. The PLA algorithm is then applied on the individual color planes. Two stages of non-text component filtering are performed: (1) filtering using heuristics such as area, number of text pixels, number of edge pixels, etc., and (2) a neural network block classifier. After the application of the PLA for each color plane, all localized text regions are merged and the final text regions are determined.

2.3 Tracking, Extraction, and Enhancement

This section discusses text tracking, extraction, and enhancement methods. In spite of its usefulness and importance for verification, enhancement, speedup, etc., tracking of text in video has not been studied extensively. There has not been much research devoted to the problems of text extraction and enhancement either. However, owing to inherent problems in locating text in images, such as low resolution and complex backgrounds, these topics need more investigation.

To enhance the system performance, it is necessary to consider temporal changes in a frame sequence. The text tracking stage can serve to verify the text localization results. In addition, if text tracking could be performed in a shorter time than text detection and localization, this would speedup the overall system. In cases where text is occluded in different frames, text tracking can help recover the original image.

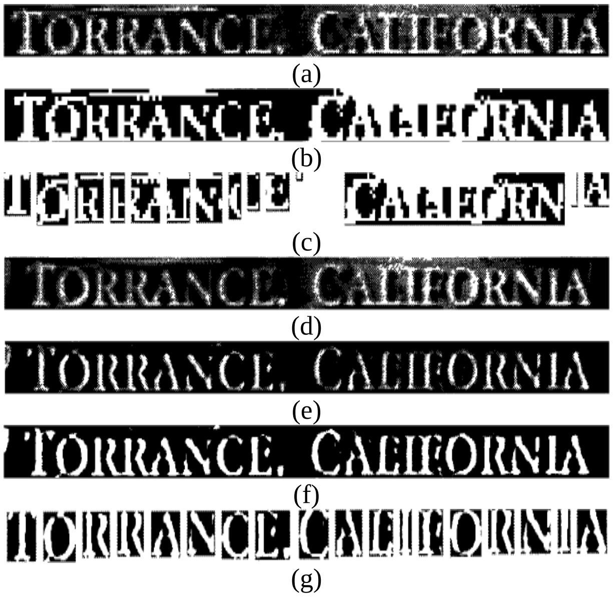
Lienhart [30, 37] described a block-matching algorithm, which is an international standard for video compression such as H.261 and MPEG, and used temporal text motion information to refine extracted text regions. The minimum mean absolute difference is used as the matching criterion. Every localized block is checked as to whether its fill factor is above a given threshold value. For each block that meets the required fill factor, a block-matching algorithm is performed. When a block has an equivalent in a subsequent frame and the gray scale difference between the blocks is less than a threshold value, the block is considered as a text component.

Antani et al. [28] and Gargi et al. [32] utilize motion vectors in an MPEG-1 bit stream in the compressed domain for text tracking, which is based on the methods of Nakajama et al. [62] and Pilu [63]. This method is implemented on the P and I frames in MPEG-1 video streams. Some preprocessing, such as checking any spatial-inconsistency and checking the number of significant edges in the motion vector, is first performed. The original bounding box is then moved by the sum of the motion vectors of all the macroblocks that correspond to the current bounding box. The matching results are refined using a correlation operation over a small neighborhood of the predicted text box region.

Li et al. [64] presented a text tracking approach that is suitable for several circumstances, including scrolling, captions, text printed on an athlete’s jersey, etc. They used the sum of the squared difference for a pure translational motion model, based on multi-resolution matching, to reduce the computational complexity. For more complex motions, text contours are used to stabilize the tracking process. Edge maps are generated using the Canny operator for a slightly larger text block. After a horizontal smearing process to group the text blocks, the new text position is extracted. However, since a pure translational model is used, this method is not appropriate to handle scale, rotation and perspective variations.

OCR systems have been available for a number of years and the current commercial systems can produce an extremely high recognition rate for machine-printed documents on a simple background. However, it is not easy to use commercial OCR software for recognizing text extracted from images or video frames. New OCR systems need to be developed to handle large amount of noise and distortion in TIE applications. Sawaki et al. [65] proposed a method for adaptively acquiring templates of degraded characters in scene images involving the automatic creation of ‘context-based image templates’ from text line images. Zhou et al. [66-68] use their own OCR algorithm based on a surface fitting classifier and an n-tuple classifier.

We now address the issue of text extraction and enhancement for generating the input to an OCR algorithm. Although most text with simple background and high contrast can be correctly localized and extracted, poor quality text can be difficult to extract. Text enhancement techniques can be divided into two categories: single frame-based or multiple frame-based. Many thresholding techniques have been developed for still images. However, these methods do not work well for video sequences. Based on the fact that text usually spans several frames, various approaches that use a tracking operation in consecutive frames have been proposed to enhance the text quality. Such enhancement methods can be effective when the background movement is different from the text movement.



(a)

(b)

(c)

(d)

(e)

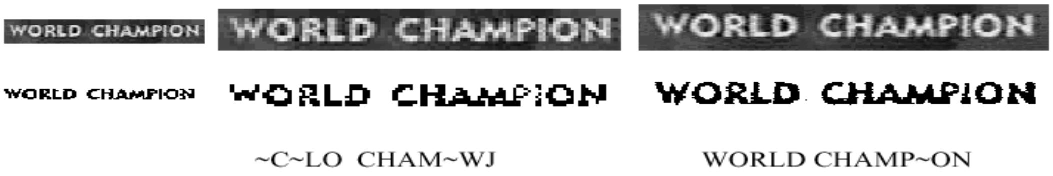
(f)

(g)

Fig. 11. Examples of Sato et al. [11]'s approach: (a) original image (204×14 pixels), (b) binary image, (c) character extraction result using the conventional technique, (d) result of sub-pixel interpolation and multi-frame integration (813×56 pixels), (e) integration of four character-extraction filters, (f) binary image, (g) result of character segmentation (courtesy of Sato et al. [11]).

Sato et al. [11] used a linear interpolation technique to magnify small text at a higher resolution for commercial OCR software. Text regions are detected and localized using Smith and Kanade's method [7], and sub-pixel linear interpolation is applied to obtain higher resolution images. Based on the fact that complex backgrounds usually include some movement, and the video captions are relatively stable across frames, the image quality is improved by multi-frame integration using resolution-enhanced frames. This method is unable to clean the background when both the text and the background are moving at the same time. After the image enhancement stages, four specialized character-extraction filters are applied based on correlation, and a recognition-based segmentation method is used for character segmentation. This means that the intermediate character recognition results are used to enhance the character segmentation results. This method takes about 120 seconds to process a 352×242 frame on MIPS R4400 200 MHz processor, and almost doubles the recognition rate of a conventional OCR technique that performs binarization of an image based on a simple thresholding, character extraction using a projection analysis, and matching by correlation. Figure 11 shows intermediate results from the enhancement procedure.

Li et al. [51, 53] presented several approaches for text enhancement. For example, they use the Shannon interpolation technique to enhance the image resolution of video images. The image resolution is increased using an extension of the Nyquist sampling theorem and it is determined whether the text is normal or inverse by comparing it with a global threshold and background color. Niblack's [70] adaptive thresholding method is used to filter out non-text regions. They investigated the relationship between the OCR accuracy and the resolution and found that the best OCR results were obtained when using a factor of 4 in image interpolation. Figure 12 shows examples of OCR with no enhancement, zeroth order interpolation, and Shannon interpolation.



(a) (b) (c)

Fig. 12. Comparison of OCR results: with (a) no enhancement, (b) zeroth order interpolation, (c) Shannon interpolation (reproduced from Li et al. [51, 53]).

Li and Doermann [64] also used a multiple frame-based text image enhancement technique, where consecutive text blocks are registered using a pure translational motion model. To ensure that the text blocks are correctly tracked, the mean square errors of two consecutive text blocks and motion trail information are used. A sub-pixel accuracy registration is also performed using bi-linear interpolation to reduce the noise sensitivity of the two low-resolution text blocks. However, this can reduce non-text regions only when text and background have different movements. Lastly, a super-resolution-based text enhancement scheme is also presented for de-blurring scene text, involving a projection onto convex sets (POCS)-based method [69].

Antani et al. [23] used a number of algorithms for extracting text. They developed a binarization algorithm similar to Messelodi and Modena's approach [39]. To refine the binarization result, a filtering stage is employed based on color, size, spatial location, topography, and shape. It is not easy to determine whether text is lighter or darker than the background. Therefore, many approaches assume that the text is brighter than the background. However, this is not always true. Therefore, these extraction algorithms operate on both the original image and its inverse image to detect text regions.

Chen et al. [87] proposed a text enhancement method which uses a multi-hypotheses approach. Text regions are located using their former work [86], where text-like regions are detected using horizontal and vertical edges. Text candidates are localized using their base line locations that are filtered by a Support Vector Machine. Localized text regions, which are gray scale images, are transferred to an EM-based segmentation stage, followed by geometric filtering using connected component analysis. By varying the number of Gaussians in the segmentation stage, multiple hypotheses are provided to an OCR system and the final result is selected from a set of outputs. In experiments using 9,562 extracted characters, as described by the authors, 92.5% character recognition rate is acquired by Open OCR toolkit (OpenRTK) from Expervision.

3 Performance Evaluation

There are several difficulties related to performance evaluation in nearly all research areas in computer vision and pattern recognition (CVPR). The empirical evaluation of CVPR algorithms is a major endeavor as a means of measuring the ability of algorithms to meet a given set of requirements. Although various studies in CVPR have investigated the issue of objective performance evaluation, there has been very little focus on the problem of TIE in images and video. This section reviews the current evaluation methods used for TIE and highlights several issues in these evaluation methods.

The performance measure used for text *detection*, which is easier to define than for *localization* and *extraction*, is the *detection rate*, defined as the ratio between the number of detected text frames and all the given frames containing text. Measuring the performance of text extraction is extremely difficult and until now there has been no comparison of the different extraction methods. Instead, the performance is merely inferred from the OCR results, as the text extraction performance is closely related to the OCR output.

Performance evaluation of text localization is not simple. Some of the issues related to the evaluation of text localization methods have been summarized by Antani et al. [23]

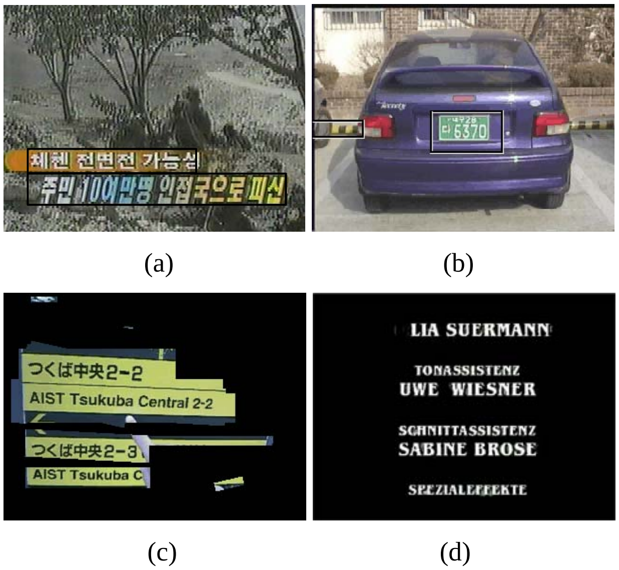
(i) *Ground truth data*: Unlike evaluating the automatic detection of other video events, such as video shot changes, vehicle detection, or face detection, the degree of preciseness of TIE is difficult to define. This problem is related to the construction of the ground truth data. The ground truth data for text localization is usually marked by bounded rectangles that include gaps between characters, words, and text lines. However, if an algorithm is very accurate and detects text at the character level, it will not include the above gaps and thus will not have a good recall rate [22].

(ii) *Performance measure*: After determining the ground truth data, a decision has to be made on which measures to use in the matching process between localized results and ground truth data. Normally, the recall and precision rates are used. Additionally, a method is also needed for comparing the ground truth data and the algorithm output: pixel-by-pixel, character-by-character, or rectangle-by-rectangle comparison.

(iii) *Application dependence*: The aim of each text localization system can differ. Some applications require that all the text in the input image must be located, while others only focus on extracting important text. In addition, the performance also depends on the weights assigned to *false alarm* or *false dismissal.*

(iv) *Public database*: Although many researchers seek to compare their methods with others, there are no domain-specific or general comprehensive databases of images or videos containing text. Therefore, researchers use their own databases for evaluating the performance of the algorithms. Further, since many algorithms include specific assumptions and are usually optimized on a particular database, it is hard to conduct a comprehensive objective comparison.

(v) *Output format*: The results from different localization algorithms may be different, which also makes it difficult to compare their performances. Various examples are shown in Fig. 13. Some papers only present a rectangular region containing text and are unconcerned about the text skew (Fig. 13(b)). Other localization algorithms just present text blocks. A localized text region has to be segmented into a set of text lines before being fed into an OCR algorithm (Fig. 13(a)). Some algorithms localize text regions while considering the skew (Fig. 13(c)), and others perform more processing to extract text pixels from the background (Fig. 13(d)).



(a) (b)

(c) (d)

Fig. 13. Different localization and extraction results: (a) text line extraction, (b) boundary rectangles without considering skew, (c) rectangles considering skew, and (d) text pixel extraction (from LAMP[2] Web site).

3.1 Public databases

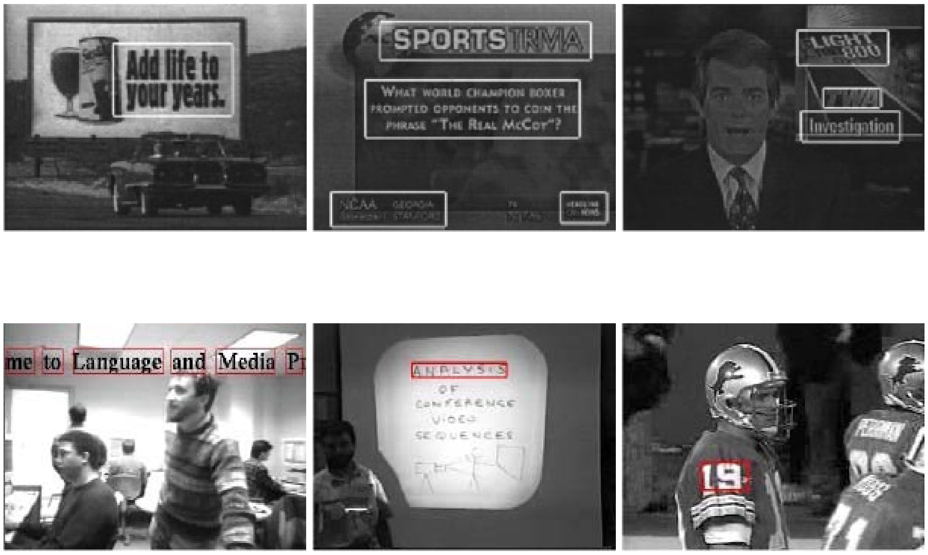
We summarize TIE database sources that can be down loaded from the Internet, although some have no ground truth data. There are also several research institutes that are currently working on this problem (see Table 3). Readers can find detailed information and test data on the web sites shown in Table 3. Fig. 14 shows a collection of images gathered from several

[3] http://documents.cfar.umd.edu/LAMP/

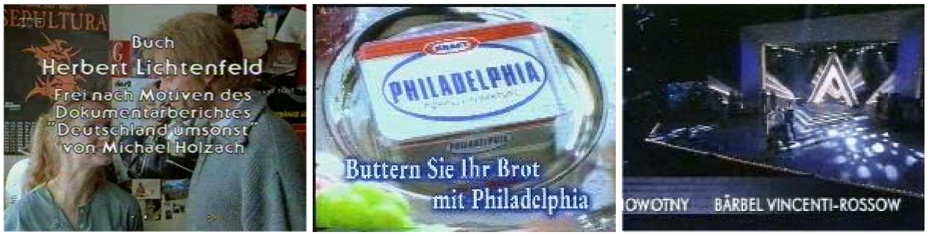
research institutes. Some images contain bounding boxes as a result of a text localization algorithm.

Table 3. Research groups and test data for TIE

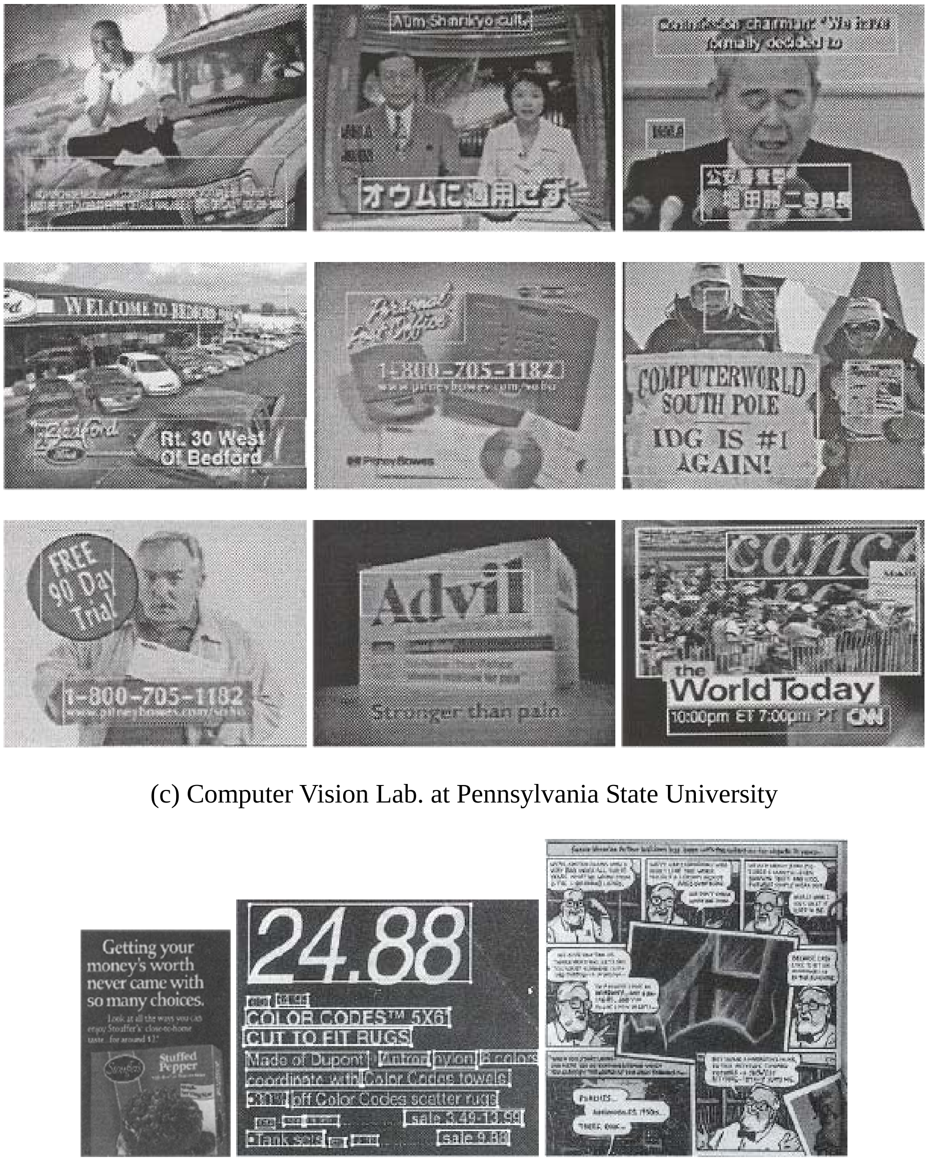
Data Set	Location	Features
Laboratory for Language And Media Processing (LAMP)	http://documents.cfar.umd.edu/LAMP/ http://www.cfar.umd.edu/~doermann/UMDTextDetectionData.tar.gz	Demonstrations, database and evaluation program
Automatic Movie Content Analysis (MoCA) Project	http://www.informatik.uni-mannheim.de/informatik/pi4/projects/MoCA/	Source code and test data
Computer Vision Lab., Pennsylvania State University	http://vision.cse.psu.edu/data/textvid/-	Lots of data, including caption and scene text
The Center for Intelligent Information Retrieval, University of Massachusetts	http://ciir.cs.umass.edu	Well organized reference papers
UW-II, UW-III Databases, ISL, University of Washington and Queens College, New York.	http://documents.cfar.umd.edu/resources/database/UWII.html	English, Japanese text; Mathematical and Chemical formulae.


Add life to your years.
SPORTS TRIVIA
WHAT WORLD CHAMPION BOXER PROMPTED OPPONENTS TO COIN THE PHRASE "THE REAL McCOY"?
NCAA
Investigation
me to Language and Media Pr
ANALYSIS OF CONFERENCE VIDEO SEQUENCES
19

(a) LAMP


SEPULTURA
Buch
Herbert Lichtenfeld
Frei nach Motiven des Dokumentarberichtes "Deutschland umsonst" von Michael Holzach
PHILADELPHIA
Buttern Sie Ihr Brot mit Philadelphia
OWOTNY BÄRBEL VINCENTI-ROSSOW

(b) MoCA



(c) Computer Vision Lab. at Pennsylvania State University

(d) Informedia project team at CMU

Fig. 14. Sample images.

3.2 Review

There have been very few attempts that quantitatively evaluate TIE algorithms, reported in the literature. Antani et al. and Hua et al. recently presented their comparison methodologies and results. Antani et al. [22] used a formal evaluation method for TIE, and selected five algorithms [29, 57, 58, 59, 71] as promising. Each algorithm was slightly modified for the sake of comparison and the ground truth data was generated, frame-by-frame, manually to include text box size, position, and orientation angle. The evaluation was performed frame-by-frame and pixel-by-pixel. After classifying all the pixels as *Correct Detection*, *False Alarm*, or *Missed Detection*, the recall and precision rates were calculated for all the algorithms, along with approximate processing times. Thereafter, the authors concluded that “*No one text detection and localization* *method is robust for detecting all kinds of text, and more than one algorithm is necessary to be able to detect all kinds of text"*. All the five algorithms exhibited lower than 35% recall rates and lower than 50 % precision rates. However, the five algorithms tested are not frequently referred to in TIE literature, so the rationale for their choice is not clear.

Hua et al. [72] proposed a performance evaluation protocol for text localization algorithms based on Liu and Dori's method [73] of text segmentation from engineering drawings. The ground truth data is constructed from 45 video frames, and the *localization difficulty* (LD) and *localization importance* (LI) are defined based on the ground truth data. The LD depends on the following factors: text box location, height, width, character height variance, skew angle, color and texture, background complexity, string density, contrast, and recognition importance level. LI is defined as the multiplication of the text length and the recognition importance level. The localization qualities (LQ) are checked using the size of overlapping area between ground truth data and localization result. The overall localization rate is the weighted value relative to the LI of the LQ. Although the ground truth data was generated carefully, the LI and LD still depend on the application and TIE algorithm. Nonetheless, as argued by the authors, their method can be used to determine the optimal parameters for yielding the best decision results for a given algorithm.

4 Applications

There are numerous applications of a text information extraction system, including document analysis, vehicle license plate extraction, technical paper analysis, and object-oriented data compression. In the following, we briefly describe some of these applications.

- *Wearable or portable computers*: with the rapid development of computer hardware technology, wearable computers are now a reality. A TIE system involving a hand-held device and camera was presented as an application of a wearable vision system. Watanabe's [74] translation camera can detect text in a scene image and translate Japanese text into English after performing character recognition. Haritaoglu [75] also demonstrated his TIE system on a hand-held device.
- *Content-based video coding or document coding*: The MPEG-4 standard supports object-based encoding. When text regions are segmented from other regions in an image, this can provide higher compression rates and better image quality. Feng et al. [76] and Cheng et al. [77] apply adaptive dithering after segmenting a document into several different classes. As a result, they can achieve a higher quality rendering of documents containing text, pictures, and graphics.
- *License/container plate recognition*: There has already been a lot of work done on vehicle license plate and container plate recognition. Although container and vehicle license plates share many characteristics with scene text, many assumptions have been made regarding the image acquisition process (camera and vehicle position and direction,

illumination, character types, and color) and geometric attributes of the text. Cui and Huang [9] model the extraction of characters in license plates using Markov random field. Meanwhile, Park et al. [44] use a learning-based approach for license plate extraction, which is similar to a texture-based text detection method [47, 49]. Kim et al. [88] use gradient information to extract license plates. Lee and Kankanhalli [34] apply a connected component-based method for cargo container verification.

- *Text-based image indexing*: This involves automatic text-based video structuring methods using caption data [11, 78].
- *Texts in WWW images*: The extraction of text from WWW images can provide relevant information on the Internet. Zhou and Lopresti [67, 68] use a CC-based method after color quantization.
- *Video content analysis*: Extracted text regions or the output of character recognition can be useful in genre recognition [1, 79]. The size, position, frequency, text alignment, and OCR-ed results can all be used for this.
- *Industrial automation*: Part identification can be accomplished by using the text information on each part [80].

5 Discussion

We have provided a comprehensive survey of text information extraction in images and video. Even though a large number of algorithms have been proposed in the literature, no single method can provide satisfactory performance in all the applications due to the large variations in character font, size, texture, color, etc.

There are several information sources for text information extraction in images (e.g., color, texture, motion, shape, geometry, etc). It is advantageous to merge various information sources to enhance the performance of a text information extraction system. It is, however, not clear as to how to integrate the outputs of several approaches. There is a clear need for a public domain and representative test database for objective benchmarking. The lack of a public test set makes it difficult to compare the performances of competing algorithms, and creates difficulties when merging several approaches.

For caption text, significant progress has been made and several applications, such as an automatic video indexing system, have already been presented. However, their text extraction results are inappropriate for general OCR software: text enhancement is needed for low quality video images and more adaptability is required for general cases (e.g., inverse characters, 2D or 3D deformed characters, polychrome characters, and so on). Very little work has been done on scene text. Scene text can have different characteristics from caption text. For example, part of a scene text can be occluded or it can have complex movement, vary in size, font, color, orientation, style, alignment, lighting, and transformation.

Although many researchers have already investigated text localization, text detection and tracking for video images is required for utilization in real applications (e.g., mobile handheld devices with a camera and real-time indexing systems). A text-image-structure-analysis, analogous to a document structure analysis, is needed to enable a text information extraction system to be used for any type of image, including both scanned document images and real scene images through a video camera. Despite the many difficulties in using TIE systems in real world applications, the importance and usefulness of this field continues to attract much attention. A text localization competition will be held at the International Conference on Document Analysis and Recogntion'2003. We hope that this competition will lead to active discussion on the problem of performance evaluation.

References

1. H. K. Kim, Efficient Automatic Text Location Method and Content-Based Indexing and Structuring of Video Database, Journal of Visual Communication and Image Representation 7 (4) (1996) 336-344.
2. Yu Zhong and Anil K. Jain, Object Localization using Color, Texture, and Shape, Pattern Recognition 33 (2000) 671-684.
3. S. Antani, R. Kasturi, and R. Jain, A Survey on the Use of Pattern Recognition Methods for Abstraction, Indexing, and Retrieval of Images and Video, Pattern Recognition 35 (2002) 945-965.
4. M. Flickner, H. Sawney et al., Query by Image and Video Content: The QBIC System, IEEE Computer 28 (9) (1995) 23-32.
5. H. J. Zhang, Y. Gong, S. W. Smoliar, and S. Y. Tan, Automatic Parsing of News Video, Proc. of IEEE Conference on Multimedia Computing and Systems, 1994, pp. 45-54.
6. Arnold W.M. Smeulders, Simone Santini, Amarnath Gupta, and Ramesh Jain, Content-Based Image Retrieval at the End of the Early Years, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (12) (2000) 1349-1380.
7. M.A. Smith and T. Kanade, Video Skimming for Quick Browsing Based on Audio and Image Characterization, Technical Report CMU-CS-95-186, Carnegie Mellon University, July 1995.
8. M. H. Yang, D. J. Kriegman, and N. Ahuja, Detecting faces in Images: A Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence, 24 (1) (2002) 34-58.
9. Y. Cui and Q. Huang, Character Extraction of License Plates from Video, Proc. of IEEE Conference on Computer Vision and Pattern Recognition, 1997, pp. 502 –507.
10. C. Colombo, A. D. Bimbo, and P. Pala, Semantics in Visual Information Retrieval, IEEE Multimedia, 6 (3) (1999) 38-53.
11. T. Sato, T. Kanade, E. K. Hughes, and M. A. Smith, Video OCR for Digital News Archive, Proc. of IEEE Workshop on Content based Access of Image and Video Databases, 1998, pp. 52-60.
12. Atsuo Yoshitaka and Tadao Ichikawa, A Survey on Content-based Retrieval for Multimedia Databases, IEEE Transactions on Knowledge and Data Engineering, 11 (1) (1999) 81-93.
13. W. Qi, L. Gu, H. Jiang, X. Chen, and H. Zhang, Integrating Visual, Audio, and Text Analysis for News Video, Proc. of IEEE International Conference on Image Processing, 2000, pp. 10-13.
14. H. D.Wactlar, T. Kanade, M. A. Smith, and S. M. Stevens, Intelligent Access to Digital Video: The Informedia Project, IEEE Computer, 29 (5) (1996) 46-52.
15. H. Rein-Lien, M. Abdel-Mottaleb, A. K. Jain, Face Detection in Color Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, 24 (5) (2002) 696-706.

16. C. Strouthpoulos, N. Papamarkos, and A.E. Atsalakis, Text Extraction in Complex Color Document, Pattern Recognition, 35 (8) (2002) 1743-1758.

17. A. K. Jain, and Y. Zhong, Page Segmentation using Texture Analysis, Pattern Recognition, 29 (5) (1996) 743-770.

18. Y. Y. Tang, S. W. Lee, and C. Y. Suen, Automatic Document Processing: A Survey, Pattern Recognition, 29 (12) (1996) 1931-1952.

19. B. Yu, A. K. Jain, and M. Mohiuddin, Address Block Location on Complex Mail Pieces, Proc. of International Conference on Document Analysis and Recognition, 1997, pp. 897-901.

20. D. S. Kim and S. I. Chien, Automatic Car License Plate Extraction using Modified Generalized Symmetry Transform and Image Warping, Proc. of International Symposium on Industrial Electronics, 2001, Vol. 3, pp. 2022-2027.

21. J. C. Shim, C. Dorai, and R. Bolle, Automatic Text Extraction from Video for Content-based Annotation and Retrieval, Proc. of International Conference on Pattern Recognition, Vol. 1, 1998, pp. 618-620.

22. S. Antani, D. Crandall, A. Narasimhamurthy, V. Y. Mariano, and R. Kasturi, Evaluation of Methods for Detection and Localization of Text in Video, Proc. of the IAPR workshop on Document Analysis Systems, Rio de Janeiro, December 2000, pp. 506-514.

23. S. Antani, Reliable Extraction of Text From Video, PhD thesis, Pennsylvania State University, August 2001.

24. Z. Lu, Detection of Text Region from Digital Engineering Drawings, IEEE Transactions on Pattern Analysis and Machine Intelligence, 20 (1998) 431-439.

25. D. Crandall, S. Antani, and R. Kasturi, Robust Detection of Stylized Text Events in Digital Video, Proceedings of International Conference on Document Analysis and Recognition, 2001, pp. 865-869.

26. D. Chen, J. Luettin, and K. Shearer, A Survey of Text Detection and Recognition in Images and Videos, Institut Dalle Molled'Intelligence Artificielle Perceptive (IDIAP) Research Report, IDIAP-RR 00-38, August 2000.

27. Yu Zhong, Hongjiang Zhang, and Anil K. Jain, Automatic Caption Localization in Compressed Video, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22, (4) (2000) 385-392.

28. S. Antani, U. Gargi, D. Crandall, T. Gandhi, and R. Kasturi, Extraction of Text in Video, Technical Report of Department of Computer Science and Engineering, Penn. State University, CSE-99-016, August 30, 1999.

29. U. Gargi, S. Antani, and R. Kasturi, Indexing Text Events in Digital Video Database, Proc. of International Conference on Pattern Recognition, 1998, vol. 1. pp. 1481-1483.

30. R. Lienhart and F. Stuber, Automatic Text Recognition In Digital Videos, Proc. of SPIE, 1996, pp. 180-188.

31. A. K. Jain, and B. Yu, Document Representation and Its Application to Page Decomposition, IEEE Transactions on Pattern Analysis and Machine Intelligence, 20 (3) (1998) 294-308.

32. U. Gargi, D. Crandall, S. Antani, T. Gandhi, R. Keener, and R. Kasturi, A System for Automatic Text Detection in Video, Proc. of International Conference on Document Analysis and Recognition, 1999, pp. 29 – 32.

33. Y. K. Lim, S. H. Choi, and S.W. Lee, Text Extraction in MPEG Compressed Video for Content-based Indexing, Proc. of International Conference on Pattern Recognition, 2000 pp. 409-412.

34. J. Ohya, A. Shio, and S. Akamatsu, Recognizing Characters in Scene Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, 16 (2) (1994) 214-224.

35. C.M. Lee, and A. Kankanhalli, Automatic Extraction of Characters in Complex Images, International Journal

of Pattern Recognition Artificial Intelligence, 9 (1) (1995) 67-82.

36. Yu Zhong, Kalle Karu, and Anil K. Jain, Locating Text In Complex Color Images, Pattern Recognition, 28 (10) (1995) 1523-1535.

37. R. Lienhart and W. Effelsberg, Automatic Text Segmentation and Text Recognition for Video Indexing, Technical Report TR-98-009, Praktische Informatik IV, University of Mannheim, 1998.

38. A. K. Jain, and B. Yu, Automatic Text Location in Images and Video Frames, Pattern Recognition, 31 (12) (1998) 2055-2076.

39. S. Messelodi and C.M. Modena, Automatic Identification and Skew Estimation of Text Lines in Real Scene Images, Pattern Recognition, 32 (1992) 791-810.

40. E. Y. Kim, K. Jung, K. Y. Jeong, and H. J. Kim, Automatic Text Region Extraction Using Cluster-based Templates, Proc. of International Conference on Advances in Pattern Recognition and Digital Techniques, 2000, pp. 418-421.

41. S.-W. Lee, D.-J. Lee, and H.-S. Park, A New Methodology for Gray-scale Character Segmentation and Recognition, IEEE Transactions on Pattern Recognition and Machine Intelligence, 18 (10) (1996) 1045-1050.

42. Yassin M. Y. Hasan and Lina J. Karam, Morphological Text Extraction from Images, IEEE Transactions on Image Processing, 9 (11) (2000) 1978-1983.

43. D. Chen, K. Shearer, and H. Bourlard, Text Enhancement with Asymmetric Filter for Video OCR, Proc. of International Conference on Image Analysis and Processing, 2001, pp. 192-197.

44. S. H. Park, K. I. Kim, K. Jung, and H. J. Kim, Locating Car License Plates using Neural Networks, IEE Electronics Letters, 35 (17) (1999) 1475-1477.

45. V. Wu, R. Manmatha, and E. M. Riseman, TextFinder: An Automatic System to Detect and Recognize Text in Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, 21 (11) (1999) 1224-1229.

46. V. Wu, R. Manmatha, and E. R. Riseman, Finding Text in Images, Proc. of ACM International Conference on Digital Libraries, 1997, pp. 1-10.

47. A. K. Jain, and K. Karu, Learning Texture Discrimination Masks, IEEE Transactions on Pattern Analysis and Machine Intelligence, 18 (2) (1996) 195-205.

48. A. K. Jain, and S. Bhattacharjee, Text Segmentation using Gabor Filters for Automatic Document Processing, Machine Vision and Application, 1992, Vol. 5, pp. 169-184.

49. K. Jung, Neural network-based Text Location in Color Images, Pattern Recognition Letters, 22 (14) December (2001) 1503-1515.

50. K. Y. Jeong, K. Jung, E. Y. Kim, and H. J. Kim, Neural Network-based Text Location for News Video Indexing, Proc. of IEEE International Conference on Image Processing, 1999, Vol. 3, pp. 319-323.

51. H. Li, D. Doerman, and O. Kia, Automatic Text Detection and Tracking in Digital Video, IEEE Transactions on Image Processing, 9 (1) January (2000) 147-156.

52. K. I. Kim, K. Jung, S. H. Park, and H. J. Kim, Support Vector Machine-based Text Detection in Digital Video, Pattern Recognition, 34 (2) (2001) 527-529.

53. H. Li and D. Doermann, A Video Text Detection System based on Automated Training, Proc. of IEEE International Conference on Pattern Recognition, 2000, pp. 223-226.

54. B. T. Chun, Y. Bae, and T. Y. Kim Automatic Text Extraction in Digital Videos using FFT and Neural Network, Proc. of IEEE International Fuzzy Systems Conference, 1999, Vol. 2, pp. 1112 –1115.

55. K. Jung, K. I. Kim, and J. Han, Text Extraction in Real Scene Images on Planar Planes, Proc. of International

Conference on Pattern Recognition, 2002, Vol. 3, pp. 469-472.

56. B.L. Yeo, and B. Liu, Visual Content Highlighting via Automatic Extraction of Embedded Captions on MPEG Compressed Video, Proc. of SPIE, 1996, pp.142-149.

57. N. Chaddha, R. Sharma, A. Agrawal, and A. Gupta, Text Segmentation in Mixed-mode Images, Proc. of Asilomar Conference on Signals, Systems and Computers, 1994, pp. 1356-1361.

58. M.v.d.Schaar-Mitrea and P. de With, Compression of Mixed Video and Graphics Images for TV Systems, Proc. of SPIE Visual Communication and Image Processing, 1998, pp. 213-221.

59. F. LeBourgeois, Robust Multifont OCR System from Gray Level Images, Proc. of International Conference on Document Analysis and Recognition, 1997, Vol. 1, pp. 1-5.

60. T. Gandhi, R. Kasuturi and S. Antani, Application of Planar Motion Segmentation for Scene Text Extraction, Proc. of International Conference on Pattern Recognition, 2000, Vol. 1, pp. 445-449.

61. P. L'assainato, P. Gamba, and A. Mecocci, Character Recognition in External Scenes by Means of Vanishing Point Grouping, Proc. of the 13th International Conference on Digital Signal Processing, 1997, pp. 691-694.

62. Y. Nakajima, A. Yoneyama, H. Yanagihara, and M. Sugano, Moving Object Detection from MPEG Coded Data, Proc. of SPIE, 1998, Vol. 3309, pp.988-996.

63. M. Pilu, On Using Raw MPEG Motion Vectors to Determine Global Camera Motion, Proc. of SPIE, 1998, Vol. 3309, pp. 448-459.

64. H. Li, O. Kia, and D. Doermann, Text Enhancement in Digital Video, Proc. of SPIE, Document Recognition IV, 1999, pp. 1-8.

65. M. Sawaki, H. Murase, and N. Hagita, Automatic Acquisition of Context-based Image Templates for Degraded Character Recognition in Scene Images, Proc. of International Conference on Pattern Recognition, 2000, Vol. 4, pp. 15-18.

66. J. Zhou and D. Lopresti, Extracting text from WWW Images, Proc. of International Conference on Document Analysis and Recognition, 1997, Vol. 1, pp. 248 –252.

67. J. Zhou, D. Lopresti, and Z. Lei, OCR for World Wide Web Images, Proc. of SPIE on Document Recognition IV, Vol. 3027, 1997, pp. 58-66.

68. J. Zhou, D. Lopresti, and T. Tasdizen, Finding Text in Color Images, Proc. of SPIE on Document Recognition V, 1998, pp. 130-140.

69. H. Li and D. Doermann, Superresolution-based Enhancement of Text in Digital Video, Proc. of International Conference of Pattern Recognition, 2000, Vol. 1, pp. 847-850.

70. YL. Niblack, An Introduction to Image Processing, Englewood Cliffs, N. J.:Prentice Hall, 1986.

71. V. Y. Mariano and R. Kasturi, Locating Uniform-Colored Text in Digital Videos, Proc. of International Conference on Pattern Recognition, 2000, Vol. 1, pp. 539-542.

72. X. S. Hua, L. Wenyin, and H. J. Zhang, Automatic Performance Evaluation for Video Text Detection, Proc. of International Conference on Document Analysis and Recognition, 2001, pp. 545 –550.

73. W. Y. Liu, and D. Dori, A Proposed Scheme for Performance Evaluation of Graphics/Text Separation Algorithm, Graphics Recognition – Algorithms and Systems, K. Tombre and A. Chhabra (eds.), Lecture Notes in Computer Science, 1998, Vol. 1389, pp. 359-371.

74. Y. Watanabe, Y. Okada, Y. B. Kim, and T. Takeda, Translation Camera, Proc. of International Conference on Pattern Recognition, 1998, Vol. 1, pp. 613-617.

75. I. Haritaoglu, Scene Text Extraction and Translation for Handheld Devices, Proc. of IEEE Conference on

Computer Vision and Pattern Recognition, 2001, Vol. 2, pp.408-413.

76. G. Feng, H. Cheng, and C. Bouman, High Quality MRC Document Coding," Proc. of International Conference on Image Processing, Image Quality, Image Capture Systems Conference (PICS), Montreal Canada, 2001.
77. H. Cheng, C. A. Bouman, and J. P. Allebach, Multiscale Document Segmentation, Proc. of IS&T 50th Annual Conference, 1997, pp. 417-425, Cambridge, MA.
78. B. Shahraray and D. C. Gibbon, Automatic Generation of Pictorial Transcripts of Video Programs, Proc. of SPIE, 1995, Vol. 2417.
79. S. Fisher, R. Lienhart, and W. Effelsberg, Automatic Recognition of Film Genres, Proc. of ACM Multimedia'95, 1995, pp. 295-304, San Francisco.
80. Y. K. Ham, M. S. Kang, H. K. Chung, and R. H. Park, Recognition of Raised Characters for Automatic Classification of Rubber Tires, Opt. Eng., 1995, Vol. 34, pp.102-108.
81. B. Sin, S. Kim, and B. Cho, Locating Characters in Scene Images using Frequency Features, Proc. of International Conference on Pattern Recognition, 2002, Vol. 3, pp. 489-492.
82. W. Mao, F. Chung, K. Lanm, and W. Siu, Hybrid Chinese/English Text Detection in Images and Video Frames, Proc. of International Conference on Pattern Recognition, 2002, Vol. 3, pp. 1015-1018.
83. K. Jung, K. Kim, T. Kurata, M. Kourogi, and J. Han, Text Scanner with Text Detection Technology on Image Sequence, Proc. of International Conference on Pattern Recognition, 2002, Vol. 3, pp. 473-476.
84. H. Hase, T. Shinokawa, M. Yoneda, and C. Y. Suen, Character String Extraction from Color Documents, Pattern Recognition, 34 (7) (2001) 1349-1365.
85. H. Hase, T. Shinokawa, M. Yoneda, M. Sakai, and H. Maruyama, Character String Extraction by Multi-stage Relaxation, Proc. of ICDAR'97, 1997, pp. 298-302.
86. D. Chen, H. Bourlard, and J. -P. Thiran, Text Identification in Complex Background using SVM, Proc. of IEEE Conference on Computer Vision and Pattern Recognition, 2001, Vol. 2, pp. 621-626.
87. D. Chen, J. Odobez, and H. Bourlard, Text Segmentation and Recognition in Complex Background Based on Markov Random Field, Proc. of International Conference on Pattern Recognition, 2002, Vol. 4, pp. 227-230.
88. S. Kim, D. Kim, Y. Ryu, and G. Kim, A Robust License-Plate Extraction Method under Complex Image Conditions, Proc. of International Conference on Pattern Recognition, 2002, Vol. 3, pp. 216-219.