Multi-script and multi-oriented text localization from scene images

Thotreingam Kasar and A G Ramakrishnan

Medical Intelligence and Language Engineering Laboratory,
Department of Electrical Engineering, Indian Institute of Science
Bangalore, INDIA 560012.
{tkasar,ramkiag}@ee.iisc.ernet.in

Abstract. *This paper describes a new method of color text localization from generic scene images containing text of different scripts and with arbitrary orientations. A representative set of colors is first identified using the edge information to initiate an unsupervised clustering algorithm. Text components are identified from each color layer using a combination of a support vector machine and a neural network classifier trained on a set of low-level features derived from the geometric, boundary, stroke and gradient information. Experiments on camera-captured images that contain variable fonts, size, color, irregular layout, non-uniform illumination and multiple scripts illustrate the robustness of the method. The proposed method yields precision and recall of 0.8 and 0.86 respectively on a database of 100 images. The method is also compared with others in the literature using the ICDAR 2003 robust reading competition dataset.*

Keywords: Text detection, scene text, multi-script documents, multi-oriented text, camera-based document analysis

1 Introduction

Text provides useful semantic information that may be used to describe the content of a document image. While it is relatively easy to segment characters from clean scanned documents, text extraction from natural scenes is difficult since scene text can appear on any surface, not necessarily on a plane. They are often characterized by arbitrary text layouts, multiple scripts, artistic fonts, colors and complex backgrounds.

The Robust Reading Competition was held at the 7^{th} international conference on document analysis and recognition 2003 to find the system that best reads complete words in camera-captured images. The dataset contains various kinds of degradations such as uneven lighting conditions, complex backgrounds, variable font styles, low resolution and text appearing on shiny surfaces. However, there are no samples of inclined or multi-oriented text. It is also limited to English. In a multi-lingual country like India, many documents, forms and signboards are generally bi-lingual or multi-lingual in nature. Every script has certain distinctive characteristics and may require script-specific processing methods. Therefore, the presence of multiple scripts require a special treatment.

There is a felt need for methods to extract and recognize text in scenes since such text is the primal target of camera-based document analysis systems. Unlike the problem of classical object-driven image segmentation, such as separating sky from mountains, pixel-accurate segmentation is required for character recognition. Robust extraction of text is a critical requirement since it affects the whole recognition process that follows.

2 Review of text detection

The existing methods for text detection fall under the two broad categories: texture based methods and connected component based methods. Texture based methods exploit the fact that text has a distinctive texture. They use methods of texture analysis such as Gabor filtering, wavelets and spatial variance. Zhong et al. [1] use local spatial variance of gray-scale image and locate text with high variance regions. Li and Doermann [2] use a sliding window to scan the image and classify each window as text or non-text using a neural network. Sabari et al. [3] uses multi-channel filtering with a Gabor filter bank on the grayscale image. The responses of the Gabor filters and color-based CC analysis are merged and text regions are obtained using geometrical and statistical information of the individual components. Wu et al. [4] employ 9 derivative of Gaussian filters to extract local texture features and apply k-means clustering to group pixels that have similar filter outputs. Assuming that text is horizontally aligned, text strings are obtained by aggregating the filtered outputs using spatial cohesion constraints. Clark and Mirmehdi [5] apply a set of five texture features to a neural network classifier to label image regions as text and non-text. Chen and Yuille [6] extract features based on mean intensity, standard deviation of intensity, histogram and edge-linking and classify using an AdaBoost trained classifier. Shivakumara et al. [7] compute median moments of the average sub bands of wavelet decomposition and use k-means clustering (k=2) to classify text pixels. The cluster with the higher mean is chosen as the text cluster. Boundary growing and nearest neighbor concepts are used to handle skewed text.

In CC based methods, the image is segmented into regions of contiguous pixels having similar characteristics like color, intensity or edges. These CCs are then classified using a set of features distinguishing textual and non-textual characteristics followed by grouping of the textual CCs. Robust segmentation of text CCs from the image is the most critical part in the process. Gatos et al. [8] segment the grayscale and inverted grayscale images by rough estimation of foreground and background pixels to form CCs which are further filtered by using height, width and other properties of the CCs. Zhu et al. [9] use Niblack method to segment the grayscale image and each CC is described by a set of low level features. Text components are classified using a cascade of classifiers trained with Adaboost algorithm. Pan et al. [10] propose a method to detect text using sparse representation. CCs are labeled as text or non-text by a two-level labeling process using an over-complete dictionary, which is learned from edge segments of isolated character images. Layout analysis is further applied to verify these text candidates.

CC-based approaches are suitable for camera-based images since they can deal with arbitrary font styles, sizes, color and complex layouts. However, their performance significantly degrades in the presence of complex backgrounds which interfere in the accurate identification of CCs. In this paper, we introduce a novel color clustering technique for robust extraction of CCs from complex images. A set of 'intrinsic' text features are designed from the geometric, boundary, stroke and gradient properties for classifying text CCs. Finally, adjacent text CCs are grouped together making use of the spatial coherence property of text to form words.

3 Color text segmentation

Since the boundaries of characters are always closed, potential text candidates are identified from the regions bounded by closed edges. Canny edge detection is performed individually on each channel of the color image and the overall edge map E is obtained by combining the three edge images as follows:

$$E = E_R \vee E_G \vee E_B \tag{1}$$

Here, E_R, E_G and E_B are the edge images corresponding to the three color channels and $\vee$ denotes the logical OR operation. This simple method yields the boundaries of all the characters present in the document image irrespective of its color, size or orientation. However, there are several instances of broken edges in practice. We perform an edge-linking procedure to bridge narrow gaps in the resulting edge image. The co-ordinates of the end points of all the open edges are then determined by counting the number of edge pixels in a 3×3 neighborhood and the center pixels when the count is 2. These pair of edge pixels indicate the direction of the edge and are used to determine the direction for a subsequent edge-following and linking process. Depending on the arrangement of the two edge pixels, there can be 8 possible search directions namely North, North-East, East, South-East, South, South-West, West and North-West. Fig. 1(a) shows two local edge patterns for which the edge-following step is done in the North and North-East directions. The other 6 search directions are given by multiples of 90^o rotated versions of these edge patterns. The edge-following process in continued for a distance λ in the estimated search direction. If an edge pixel is encountered, the traversed path is made 'ON' and is included in the edge map. If on the other hand, no edge pixel is found at the end of λ pixel traversal, the path is ignored.

To close gaps between parallel edge segments, a simple bridging operation is also performed by dilating the pixels at the end points of all the open edges with a structuring element shown in Fig. 1(b). These post-processing steps result in closing all the small gaps that may have arisen during the thresholding step of edge detection.

$$\begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&1&0\\\hline 0&1&0\\\hline\end{array}\quad\begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&1&0\\\hline 1&0&0\\\hline\end{array}\qquad\qquad\begin{array}{|c|c|c|}\hline 0&1&0\\\hline 1&1&1\\\hline 0&1&0\\\hline\end{array}$$

(a) (b)

Fig. 1. (a) Edge patterns for which direction of search for edge-linking process is in the North and North-East directions. The 6 other search directions are given by multiples of 90^o rotated versions of these edge patterns (b) Structuring element for bridging parallel gaps.

The color values of the region bounded by each of the closed edges are then considered for the subsequent color clustering step using COCOCLUST algorithm [11]. Each of these blobs is described by its mean $L^*a^*b^*$ color. A one-pass Leader clustering process is applied on the mean colors of all the segmented components. The obtained color clusters initialize a k-means clustering process. The clusters obtained at the end of convergence of the k-means clustering represent the final segmented output. The pseudo-code for the clustering algorithm is given below.

```
Input:Color Samples,CS = {$C_1$,$C_2$,...,$C_M$}
      Color similarity threshold, $T_s$
Output:Color clusters, CL
1. Assign CL[1] = $C_1$ and Count = 1
2. For i = 2 to M, do
3.    For j = 1 to Count, do
4.       If Dist(CL[j],$C_i$)$\leq$ $T_s$
5.          CL[j] = Mean({CL[j],$C_i$})
6.          Break
7.       Else
8.          Count = Count + 1
9.          CL[Count] = $C_i$
10.      EndIf
11.   EndFor
12. EndFor
13. Perform k-means clustering initialized with the obtained
    color clusters CL
```

where $\mathrm{Dist}(\mathrm{C}_1, \mathrm{C}_2)$ denotes the distance between the colors $\mathrm{C}_1 = (L_1^*,\, a_1^*,\, b_1^*)^{\mathrm{T}}$ and $\mathrm{C}_2 = (L_2^*,\, a_2^*,\, b_2^*)^{\mathrm{T}}$ given by:

$$\mathrm{Dist}(\mathrm{C}_1,\, \mathrm{C}_2) = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2} \qquad (2)$$

The threshold parameter T_s decides the similarity between the two colors and hence the number of clusters. Antonacopoulos and Karatzas [12] perform grouping of color pixels based on the criterion that only colors that cannot be differentiated by humans should be grouped together. The threshold below which two colors are considered similar was experimentally determined and set to 20. We use a slightly higher threshold to account for the small color variations that may appear within the text strokes. In our implementation, the threshold parameter T_s is empirically fixed at 35, after experimentation.

It may be observed that unlike that of COCOCLUST, where clustering is performed on each pixel, the clustering is carried out at the component-level in this case. Each CC, described by its mean color, is checked for similarity with that of other CCs and they are grouped if the color distance is within a threshold value. Since each CC is represented by a single color (mean color), it cannot be further split and the above clustering process only allows merging of ‘similar’ CCs that were pre-segmented by the edge detection process. The result of color segmentation is reasonably robust to non-uniform illumination and shadows since edge detection is largely unaffected by these photometric distortions.

By making some sensible assumptions about the document image, obvious non-text CCs are removed by an area-based filtering. CCs whose area is less than 15 pixels or greater than one-fifth of the entire image area are not considered for further processing. Large CCs whose heights or widths are greater than one-third of the image dimension are also filtered out. This heuristic filtering removes a significant amount of noise without affecting any text CC, thereby reducing the computational load. Fig. 2 illustrates the robustness of the proposed method on an example color image where there is a gradual variation in the text color.

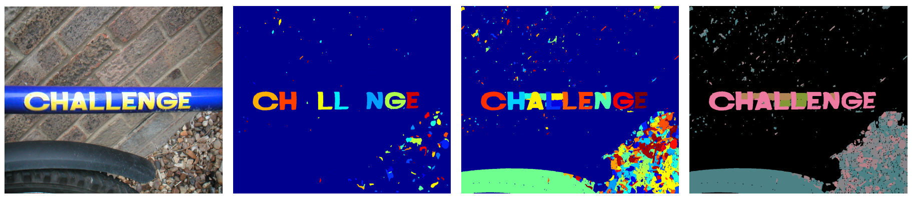


Fig. 2. Robustness of the proposed edge-guided color segmentation method (a) Input image (b) Segmented regions obtained using Canny edge detection (c) Segmented regions obtained after edge-linking (d) Segmented image obtained after color clustering. Note that the outputs shown in (b)-(d) are obtained after the area-based heuristic filtering.

4 Feature extraction for text classification

Each individual color layer is analyzed and text-like CCs are identified employing a set of 12 low-level features derived from the boundary, stroke and gradient-based features.

Geometric features: Text CCs have geometric properties that can be used to distinguish them from non-text CCs. Their aspect ratios and occupancy ratios tend to cluster around a small range of values. Text CCs are normally much smaller than most of the background clutter. They are also characterized by a small number of convex deficiency regions since text CCs are composed of a few number of strokes. For every CC, we compute the following features:

$$Aspect\,Ratio = \min\left(\frac{W}{H}, \frac{H}{W}\right) \tag{3}$$

$$Area\,Ratio = \frac{Area(CC)}{Area(Input\,Image)} \tag{4}$$

$$Occupancy = \frac{|CC|}{Area(CC)} \tag{5}$$

$$Convex\,Deficiency = \min\left(1, \frac{\#\,Convex\,deficiency}{\alpha}\right) \tag{6}$$

where W and H denote the width and height of the CC being processed and $|\,.\,|$ the number of ON pixels. The parameter α is used to normalize the feature value so that it lies in the range [0,1].

Boundary features: Text CCs generally have smooth and well-defined boundaries and hence have a higher degree of overlap with the edge image than the non-text components [9]. These characteristics are captured by the following features.

$$Boundary\,Smoothness = \frac{|CC - (CC \circ S_2)|}{|CC|} \tag{7}$$

$$Boundary\,Stability = \frac{|E_{CC} \bigcap \text{Boundary}(CC)|}{|\text{Boundary}(CC)|} \tag{8}$$

Here, E_{CC} denotes the set of edge pixels of the component CC, $\circ$ refers to the morphological opening operation and S_2 a square structuring element of size 2×2.

Stroke features: Text CCs are also characterized by a uniform stroke width all along the stroke, which is also normally much smaller than its height.

$$SW\,Deviation = \frac{StdDev[StrokeWidth(CC)]}{Mean[StrokeWidth(CC)]} \tag{9}$$

$$SW\,CCHeight\,Ratio = \frac{StrokeWidth(CC)}{H} \tag{10}$$

Assuming that each character is of a uniform color, the original CC and its corresponding binary image B should have a high degree of 'similarity' and a small degree of 'dissimilarity'. Here B is obtained by binarizing the region corresponding to the CC from the gray counterpart of the original color image. The converse holds for non-text regions since they are generally non-homogeneous in

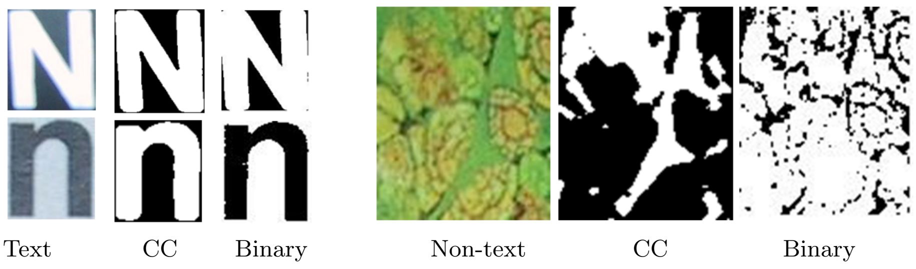


Fig. 3. Illustration of the stroke homogeneity feature. For any text CC, the corresponding binary output is 'similar' to the CC obtained from color segmentation. But, non-text CCs do not exhibit such a property owing to the inhomogeneity.

nature. This characteristic of text is named stroke homogeneity and is computed as follows:

$$
\begin{aligned}
B &= Binarize(ImagePatch) \quad (11)\\
\text{If } |CC \cap B| &\geq |CC \cap \text{NOT}(B)|\\
Sim &= |CC \cap B|\\
Dissim &= |\text{XOR}(CC, B)|\\
&\text{Else}\\
Sim &= |CC \cap \text{NOT}(B)|\\
Dissim &= |\text{XOR}(CC, \text{NOT}(B))|
\end{aligned}
$$

$$
Stroke\,Homogeneity = min\left(1, \frac{Dissim}{Sim}\right) \quad (12)
$$

Owing to its simplicity, we choose the block-based Otsu thresholding technique to binarize the gray-scale image patch described by each CC where the image patch is subdivided into 3×3 blocks. Due to the presence of inverse text, the binarized outputs may be inverted in some cases. This is illustrated in figure 3. The relation $|CC \cap B| \geq |CC \cap \text{NOT}(B)|$ holds for text brighter than the background. Thus, this test is used in obtaining the 'similarity' and 'dissimilarity' measures appropriately.

Gradient features: Text regions are characterized by a high density of edges. As pointed out by Clark and Mirmehdi [5], the gradient in text regions exhibit an anti-parallel property. Based on these observations, the following features are computed.

$$
Gradient\,Density = \frac{\sum_{(x,y)\in E_{CC}} G(x,y)}{|\,CC\,|} \quad (13)
$$

where $G(x, y)$ denotes the gradient magnitude obtained from the Gaussian derivative of the gray scale image.

$$
Gradient\,Symmetry = \frac{\sum_{i=1}^{8}[A(\theta_i) - A(\theta_{i+8})]^2}{\sum_{i=1}^{8} A(\theta_i)^2} \quad (14)
$$

$$Angle\,distribution = \frac{\sum_{i=1}^{8}[A(\theta_i) - \bar{A}]^2}{\sum_{i=1}^{8} A(\theta_i)^2} \qquad (15)$$

where $\theta(x, y)$ is the gradient orientation quantized into 16 bins i.e. $\theta_i \in \left[(i-1)\frac{\pi}{8}, \frac{i\pi}{8}\right)$; $i = 1, 2, \cdots, 16$, $A(\theta_i)$ is the magnitude of edges in direction θ_i and $\bar{A}$ is the mean gradient magnitude over all θ_i.

In order to classify the segmented CCs into text and non-text classes, we employ two classifiers namely an SVM and a neural network (NN) classifier trained on the above set of features. A total of 4551 English character CCs and 39599 non-character CCs are extracted from the training images from the ICDAR 2003 robust reading competition dataset [13]. The LIBSVM toolkit [14] is used for implementing the SVM classifier. Radial basis kernel function is used and the optimum parameters for the SVM are determined through a 5-fold validation process. In addition, we use the MATLAB NN Toolbox to implement a two-layer feed-forward neural network with one sigmoidal hidden layer and output neurons. During the testing stage, the input image is first segmented into its constituent CCs which are classified as text or non-text using the two trained classifiers. We declare a test CC as text if it is classified as text by either of the two classifiers.

5 Experiments and results

The proposed method is tested on our own database of camera-captured images with complex text layouts and multi-script content. Obviously, it is not appropriate to use rectangular word bounding boxes for quantifying the performance of the method as in the ICDAR 2003 text locating competitions. The ICDAR metric is strict and heavily penalizes errors in estimation of word boundaries that drastically affect the detection rate. Pixel-based evaluation of the performance of text detection is preferred since there is no need to consider the number of detected words. Hence, we develop a semi-automatic toolkit [15] for annotating generic scene images, which is available for free download. Using the toolkit, we obtain pixel-accurate ground truth of 100 scenic images containing text in various layout styles and multiple scripts. Figure 4 shows one such ground truth for a multi-script image where the text is laid out in an arc form. The availability of such a groundtruthed data enables us to evaluate the performance using pixel-based precision and recall measures which is also used in the Document Image Binarization Contest 2009.

1. A pixel is classified as true positive (TP) if it is ON in both ground truth and output of text detection.
2. A pixel is classified as false positive (FP) if it is ON only in the output of text detection.
3. A pixel is classified as false negative (FN) if it is ON only in the ground truth image.

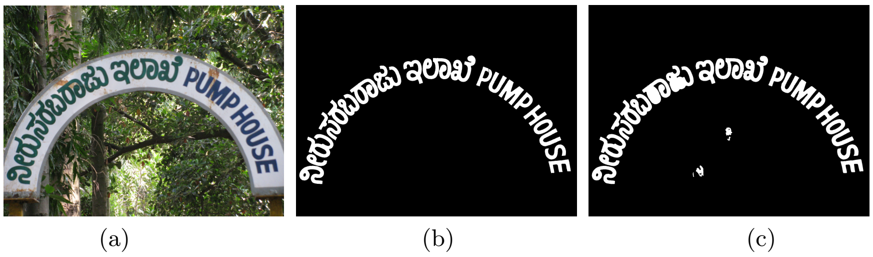


Fig. 4. (a) A sample multi-script image from our database that contains curved text. (b) The corresponding pixel-accurate ground truth (c) Text CCs identified by our method yielding precision and recall value of 0.88 and 0.99 respectively.

The precision and recall measures are then computed as,

$$p = \frac{Number\,of\,TP}{(Number\,of\,FP + Number\,of\,TP)} \tag{16}$$

$$r = \frac{Number\,of\,TP}{(Number\,of\,FN + Number\,of\,TP)} \tag{17}$$

Since the features used for identifying text CCs do not assume any characteristic of the script, the same method can detect text irrespective of the script and text orientation. Figure 5 shows some sample outputs of text detection on our database. Table 1 gives the overall performance of the proposed method, which yields $p = 0.80, r = 0.86$ and $f = 0.83$ respectively. The availability of pixel-accurate ground truth enables us to evaluate the performance of text detection directly without the need to group the text CCs into words.

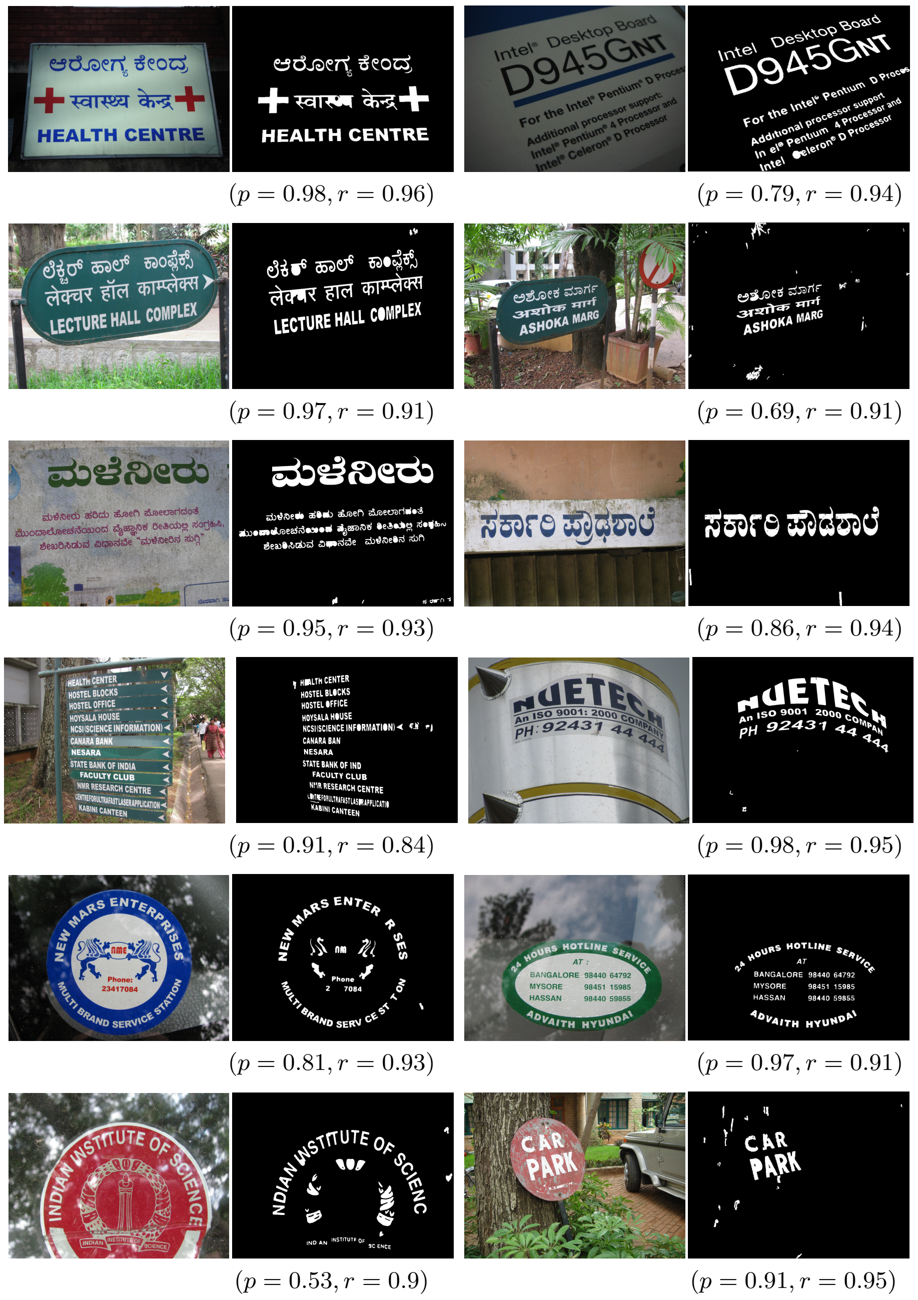
Table 1. Overall result of text localization on our dataset.

Precision	**Recall**	**f**
0.8	0.86	0.83

5.1 Estimation of word bounding boxes

In order to make a fair comparison with the results of other existing methods in the literature, we also use the ICDAR 2003 evaluation metrics computed from rectangle-based area matching score to compute the precision and recall. This requires grouping of the detected text CCs into words and obtaining its bounding rectangles for quantifying the result of text detection.

Since the ICDAR dataset contains only horizontally aligned text, we employ Delaunay triangulation to link adjacent CCs together and obtain those CCs that lie in a straight line using simple heuristics such as position, height and area. A link map, $V = \bigcup v_{ij}$ is created, where v_{ij} is an edge of a triangle linking CC_i



($p = 0.98, r = 0.96$) ($p = 0.79, r = 0.94$)

($p = 0.97, r = 0.91$) ($p = 0.69, r = 0.91$)

($p = 0.95, r = 0.93$) ($p = 0.86, r = 0.94$)

($p = 0.91, r = 0.84$) ($p = 0.98, r = 0.95$)

($p = 0.81, r = 0.93$) ($p = 0.97, r = 0.91$)

($p = 0.53, r = 0.9$) ($p = 0.91, r = 0.95$)

Fig. 5. Representative results of text localization on images with multi-script content and arbitrary orientations. The pixel-based precision and recall measures are indicated below each image.

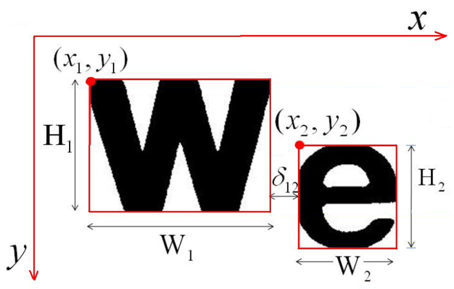


Fig. 6. The parameters that guide the process of grouping of adjacent CCs into text lines.

to CC_j obtained by applying Delaunay triangulation to the classified CCs. In the first step, the links are filtered by assigning labels based on height ratio and vertical overlap ratio:

$$\Phi(v_{ij}) = \begin{cases} +1, & [0.5 < (H_i/H_j) < 2] \wedge \\ & [(Y_i + H_i) - Y_j > 0] \wedge \\ & [(Y_j + H_j) - Y_i > 0] \\ -1, & \text{otherwise} \end{cases} \tag{18}$$

where (X_i, Y_i, W_i, H_i) are the attributes of bounding box (See Fig. 6).

The links labeled -1 are filtered out and the remaining links between the characters are used to generate text lines. Text lines containing less than 3 CCs are eliminated since a text line usually contains more than 2 characters. This criterion helps in eliminating false positives. To handle isolated characters, we mark all components with high posteriors for text during the classification step and accept only those CCs whose likelihood exceeds 0.9.

We make use of the spatial regularity in the occurrence of characters in a text line to recover false negatives. A straight line $F(x) = ax + b$ is fitted to the coordinates of the centroids $\{C_{xi}, C_{yi}\}$ of the bounding boxes of all CCs in a text group. A component CC'_k, whose bounding box center (C'_{xk}, C'_{yk}) lies inside the text line, is re-classified as text component if the following criteria are satisfied:

$$\Lambda' < mean(\Lambda) + \beta H_{line} \tag{19}$$

$$(\bar{A}_{CC}/4) < Area(CC_k) < 2\bar{A}_{CC} \tag{20}$$

where $\Lambda = abs(F(X_i) - Y_i)$, $\Lambda' = abs(F(X'_k) - Y'_k)$, β is a control parameter empirically set to 0.2, H_{line} is the height of bounding rectangle of text line and $\bar{A}_{CC}$ is the mean area of all CCs in the string. As illustrated in Fig. 7, this grouping procedure enables the recovery of CCs that were misclassified by the classifiers thereby increasing the performance of the method.

The inter bounding box distance is used to cut each text line into words. The distance between adjacent components CC_j and CC_i is calculated as follows:

$$\delta_{ji} = abs(X^i - (X^j + W^j)) \tag{21}$$

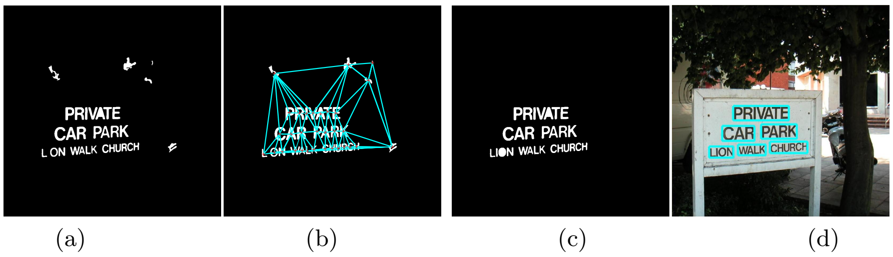


Fig. 7. Grouping of verified CCs into words and recovery of false negatives using the spatial coherence property of text. Notice that the character ‘I’ in the last line, which was initially misclassified as non-text, is re-classified as text during the grouping stage since its size is comparable to that of the adjacent text CCs.

The text line is cut wherever the following condition occurs:

$$\delta_{ji} > \gamma \times mean(\{\delta_{ji}\}) \tag{22}$$

where γ is a control parameter empirically set to 2.65. Each segment of the text line is considered as a word and its bounding rectangle is computed.

Table 2 lists the performance of our method and compares it with other existing methods. The overall precision, recall and f-measures obtained on the whole test set are 0.63, 0.59 and 0.61 respectively. The performance of our method is comparable to that of the best-performing methods in the ICDAR 2005 text locating competition [16]. It performs marginally worse than a recent method proposed by Epshtein et al. [19], which is designed for locating only horizontal text. Our method, however, works well for generic scene images having arbitrary text orientations.

Table 2. Comparison of the proposed method with performance of other techniques on ICDAR dataset.

Method	Precision	Recall	f
ICDAR 2005			
Hinnerk Becker	0.62	0.67	0.62
Chen and Yuille	0.6	0.6	0.58
Recent methods			
Nuemann and Matas [17]	0.59	0.55	0.57
Minetto et al. [18]	0.63	0.61	0.61
Epshtein et al. [19]	0.73	0.6	0.66
Proposed method	**0.63**	**0.59**	**0.61**

6 Conclusions

This paper describes an important preprocessing for scene text analysis and recognition. CC-based approaches for text detection are known for their robustness to large variations in font styles, sizes, color, layout and multi-script content and therefore suitable for processing camera-based images. However, extraction of CCs from complex background is not a trivial task. To this end, we introduce a edge-guided color clustering technique suitable for extracting text CCs from complex backgrounds. We also design a set of 'intrinsic' features of text for classifying each of the identified CC as text or non-text to achieve layout and script independence. The system's performance is enhanced by invoking the spatial regularity property of text that effectively filters out inconsistent CCs which also aid in the recovery of missed text CCs. Experiments on camera-captured images that contain variable font, size, color, irregular layout, non-uniform illumination and multiple scripts illustrate the robustness of the method.

References

1. Y. Zhong, K. Karu and A. K. Jain, Locating text in complex color images, Patt. recog., 28(10), 1523 - 1535 (1995)
2. H. Li, D. Doermann and O. Kia, Automatic Text Detection and Tracking in Digital Video, IEEE Trans. Image proc., 9(1), 147 - 156 (2000)
3. S. S. Raju, P. B. Pati and A. G. Ramakrishnan, Gabor Filter Based Block Energy Analysis for Text Extraction from Digital Document Images, Proc. Intl. workshop DIAL, 233 - 243 (2004)
4. V. Wu, R. Manmatha and E. M. Riseman, TextFinder: an automatic system to detect and recognize text in images, IEEE Trans. PAMI, 21(11), 1124 - 1129 (1999)
5. P. Clark and M. Mirmehdi, Finding text using localised measures, Proc. British Machine Vision Conf., 675 - 684 (2000)
6. X. Chen and A. L. Yuille, Detecting and Reading Text in Natural Scenes, Proc. IEEE Intl. Conf. CVPR, 2, 366 - 373 (2004)
7. P. Shivakumara, A. Dutta, C. L. Tan and U. Pal, A New Wavelet-Median-Moment based Method for Multi-Oriented Video Text Detection, Proc. Intl. Workshop on Document Analysis and Systems, 279 - 286 (2010)
8. B. Gatos, I. Pratikakis, K. Kepene and S. J. Perantonis, Text detection in indoor/outdoor scene images, Proc. Intl. Workshop CBDAR, 127 - 132 (2005)
9. K. Zhu, F. Qi, R.Jiang, L. Xu, M. Kimachi, Y. Wu and T. Aizawa, Using Adaboost to Detect and Segment Characters from Natural Scenes, Proc. Intl. Workshop CBDAR, 52-59 (2005)
10. W. Pan, T.D. Brui and C.Y. Suen, Text Detection from Scene Images Using Sparse Representation, Proc. ICPR, 1 - 5 (2008)
11. T. Kasar and A. G. Ramakrishnan, COCOCLUST: Contour-based Color Clustering for Robust Binarization of Colored Text, Proc. Intl. Workshop CBDAR, 11–17 (2009)
12. A. Antonacopoulos and D. Karatzas, Fuzzy segmentation of characters in web images based on human colour perception, Proc. Workshop Document Analysis Systems, 295 -306 (2002)

13. ICDAR 2003 Robust reading competition data set http://algoval.essex.ac.uk/icdar/Competitions.html.
14. C. C. Chang and C.J Lin, LIBSVM: a library for support vector machines, http://www.csie.ntu.edu.tw/ cjlin/libsvm.
15. T. Kasar, D. Kumar, A. N. Prasad, D. Girish and A. G. Ramakrishnan, MAST: Multi-scipt Annotation Toolkit for Scenic Text, Joint workshop on MOCR and AND, 113-120 (2011), Software available at http://mile.ee.iisc.ernet.in/mast.
16. S. M. Lucas, ICDAR 2005 Text Locating Competition Results, Proc. ICDAR, 80 - 84 (2005)
17. L. Nuemann and J. Matas, A method for text localization and recognition in real-world images, Proc. ACCV, 3, 770 -783 (2010)
18. R. Minetto, N. Thome, M. Cord, J. Fabrizio and B. Marcotegui, SNOOPERTEXT: A multiresolution system for text detection in complex visual scenes, Proc. IEEE ICIP, 3861 - 3864 (2010)
19. B. Epshtein, E. Ofek and Y. Wexler, Detecting text in natural scenes with stroke width transform, Proc. IEEE Conf. CVPR, 2963 -2970 (2010)