A Study on Automatic Text Detection and Extraction in Complex Scene Images

T. Beula Bell and Dr. M.K. Jeya kumar

Abstract

Text extraction and recognition have quite a lot of relevant application for automatic indexing or information retrieval such as document indexing, content-based image retrieval, and the famous car plate recognition which further opens up the possibility for more improved and advanced systems. Existing texts play an important role in understanding a scene image. Scene images differ from document images, which are composed of text characters of various size, shape, direction and situation along with complicated backgrounds, such as map, Picture or painting, etc. Hence the extraction of texts in scene images is a difficult as well as demanding task. Recently, numerical based algorithm finds applications to extract texts from scene images. In this paper a method for the segmentation of the text regions from compound images is projected.

INTRODUCTION

A wide variety of methods to text information extraction from images and video have been proposed for particular applications including segmentation, block location, license plate location, and content-based Image/video indexing (Keechul Junga, 2003). Text detection in an image or video border contains two stages. In the first stage, the locations of text blocks are determined and in the second stage, the precise areas of text characters are determined. In today’s complex digital world, taking out of text information present in natural scene images finds varied applications, such as automatic explanation, indexing, and structuring of images, manuscript analysis, vehicle license plate extraction, procedural paper analysis, and object oriented data firmness. However, text prints data firmness. However, text printed on magazine covers, signs, indicators, book covers, etc. always mixes with photos and designs (Ninnala Shivananda and Nagabhushan, 2003). This kind of texts in scene images may contain much information and thus separation of text strings is an important issue.

Basically, the idea behind enhancement techniques is to bring out detail that is hidden or simply to highlight certain skin texture of interest in an image. The detection and mining of text regions in an image is a well known problem in the computer vision research area. Compressing an image is significantly different than compressing raw binary data. Algorithms of automatic text localization are necessary to filter out background outliers and localize the regions containing text lettering or strings in images for further processes of text segmentation and detection. Natural scene images with text information are divided into two categories according to the complexity of the background. In the first category, text characters and strings are in high resolution with a relatively simple background. We observe that characters and strings in natural scene images are printed in same color for almost all cases.

Related Works:

Many previous text localization algorithms applied color-based clustering to group the pixels in like colors into own color layers. Thus, text characters and strings could be alienated from the background objects in different colors. Nikolaou et al. (2009) proposed an algorithm of color reduction based on color histogram and mean-shift algorithm. It initialized color centers arbitrarily and projected each pixel to the nearest color center. Then mean-shift algorithm was applied to fix the color centers into the mean positions as last color layers. Chen et al. (2004) established Gaussian mixture model (GMM) in five color channels (red, green, blue, hue, and intensity) to analyze the distributions of text pixels and background pixels. The parameters of these characteristic distributions were then used to label candidate text regions. Cosine similarity (Mancas-Thillou, 2006) and K-means clustering (Song, 2008) were, respectively, applied to RGB channels to segment text.
characters in uniform color. In these algorithms, color clustering is a single-variable function that maps each pixel to the adjacent color center. From the high-level viewpoint, it maps each text string in uniform color to the most well-suited color layer. However, these algorithms ignore that text string is attached to the neighboring surface in uniform color in most cases. Based on this feature, we design the algorithm of boundary clustering (BC) which is another significant property of text is stroke width consistency. As the basic element of text, stroke is defined as a connected region in the form of a band of approximately constant width (Epshtein, 2010).

Text features, such as edge distribution, gradient variation, closed component boundary, and edge-based filter response, were obtained from boundary maps and gradient maps of scene images to detect and verify text regions. They were connected to the geometrical structure of text. In the above algorithms, multiple pixel-based features were employed to distinguish text characters and strings from background outliers. But most of the processes were based on the individual selection of features and the hard assignment of parameters.

**Proposed Methodology:**

In this paper, we propose a novel framework of automatic text localization to calculate text regions in natural scene images by using features at 3 levels. At the pixel level, assuming that text characters and strings in scene images mostly appear in uniform color, the edge pixels are clustered into several layers to separate the boundaries of text strokes from those of background outliers with different color-pair. At the character level, assuming that each text character is composed of a single stroke, the pixels inside the body of strokes are segmented from each boundary layer to extract Candidate characters in the form of connected components. At the string level, assuming that scene text is mostly in the form of approximately horizontal strings, layout analysis is first performed to group the horizontally aligned connected components into candidate fragments of text strings, and then a text classifier is learned from training set to predict whether an image patch of candidate string fragment contains text or not.

The Color-Edge combined algorithm consists of two stages initially text background removal. In this stage, text area is detected by the Canny Edge detection algorithm. Secondly the text line extraction. In the process of color chain segmentation some difficulties may be encountered because of smooth transitions at the edge of character components e.g. due to image blurring. To overcome this, were used preprocessing operations on the input image to obtain sharp transitions at the edges.

![Fig. 1: Input Image](image1)

Popular edge enhancement techniques like the sobel filter enhance the edge using the intensity information of its eight neighbor pixel values.

**A. Edge detection:**

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 filters is used for the edge detection, and a smoothing operation.

![Fig. 2: Edge detected output image (canny)](image2)

The Canny method finds edges by looking for local maxima of the gradient. The gradient is calculated using the derivative of a Gaussian filter. The method uses two thresholds, to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges (Bharatratna, 2001; Manza, 2012).

This method is therefore less likely than the others to be fooled by noise, and more likely to detect true weak edges (Canny, 1989).
B. Boundary clustering:

Depending on the assumption that text pixels are in consistent color. As long as the difference is not very large, text pixels tend to be clustered together because the difference between text and non-text pixels is usually larger. K-means is adopted in our method because of its simplicity and efficiency. The k-means algorithm partitions a set of elements into k clusters so that the intra cluster similarity is high but the inter cluster similarity is low. In text segmentation, researches about k-means clustering for grayscale text segmentation have been done in (Chen, 2004). They take k as 2, 3 and 4 and choose the best OCR result. But it may be more complicated in practical considering the contours around the text and background complexity. Thus we have done several experiments to find out the most appropriate value for k. In this paper, we consider situations that when k is less than 5, greater than 5, and k is equal to 5.

In our experiments on scene images, the number of Gaussian mixtures is $K = 5$, which generates the best results of BC.

![Fig. 3: Clustered output images.](image)

At first, K, means clustering is applied to calculate $K$ centers of observation points, which are used as initial means of the Gaussian mixture distributions. Then the corresponding $K$ variances are calculated from the means of observation points. Thus, we can initialize a group of Gaussian distributions. EM algorithm is applied to obtain the maximum likelihood estimate of the GMM parameters, including weights, means, and variances of the $K$ Gaussian distribution. In the EM process, the GMM parameters are iteratively updated from their initial values derived by K-means clustering. This iterative update is performed until the log likelihood log is convergent. Then the boundary layer is built from each of the $K$ Gaussian distributions under the parameters derived by EM. The expectation of GMM is represented by

$$P(x|\mu, \sigma) = \sum_{i}^{K} w_i \frac{1}{\sigma_i} e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}}$$

where $x$ represents the observation points, $w_i$ represents the weights of the $i$th Gaussian distribution in the mixture set, and $\mu_i$ and $\sigma_i$ represent the mean and variance of the $i$th Gaussian distribution, respectively. Next, over the observation points of edge pixels, EM algorithm is applied to obtain the maximum likelihood estimate of the GMM parameters, including weights, means, and variances of the $K$ Gaussian distributions.

Text detection: Find the regions that contain text. Text segmentation: Segment text in the detected text regions. The result is usually a binary image for text recognition. Then extract texture features of the image and then locating text regions. And also separate the image into small regions and then grouping character regions into text regions. The character image from the detected string is selected. Each character is automatically selected by using segmentation methods. The techniques which are used are Gabor Filter – Color Bigram Features (Color text with size and exact size). The edge detector is applied on each sub-band to get the candidate text edges. If any template is unmatched template we have to make import the simulated (not recognition template) into the templates.m, So unmatched text can be recognized.

Experimental Results:

Results of color reduction based on the clustering of absolute color values, where white region represents background, and color region consists of the pixels that are grouped to the layer. In this paper, we only consider normal text (text is brighter and background is darker) for simplicity.
Fig. 4: Segmented text strokes.

We tested our method on a database of 100 images of scene images. Most of the scene images contained outdoor signs written in English alphabets along with graphics. All the text area was correctly segmented in 96(96%) of these 100 images by using MATLAB.

The comparison results of analyzed characters and extracted texts are calculated with their edge detection based groupings. The performance comparison efficiency table was given. If the text samples mismatched during the computations they are again trained as a templates with their particular character.

Table 1: Comparison of Edge Detection Algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Efficiency (%)</th>
<th>Time Elapsed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harr wavelet</td>
<td>90.2%</td>
<td>12 sec</td>
</tr>
<tr>
<td>Prewitt edge</td>
<td>85.4%</td>
<td>15 sec</td>
</tr>
<tr>
<td>Sobel edge</td>
<td>92.3%</td>
<td>11.5 sec</td>
</tr>
<tr>
<td>Canny edge</td>
<td>95.6%</td>
<td>9.2 sec</td>
</tr>
</tbody>
</table>

Among the actual evaluation measures of the text segmentation accuracy, the classical measures of precision, the ratio of the true boundaries among the false boundaries, recall of referent boundaries are retrieved.

Conclusion:
The developed algorithm with the MATLAB platform, and the study comparison of the all other algorithms which are all based on the character recognition. The Character recognized was improved comparatively with other existing algorithms. The extracted text can be further processed in different kinds of applications in this paper; we designed a novel framework to extract the text patterns in the scene images. This new text extraction algorithm automatically detects and extracts text from complex background images.

REFERENCES


