DETECTION OF TEXT STRING FROM NATURAL SCENES BY IMAGE PARTITION AND GROUPING

Mrs. Prajakta P. Bastawade  
Department of Information Technology  
MIT College of Engineering, Pune

Mrs. Bharati Dixit  
Department of Information Technology  
MIT College of Engineering, Pune

Abstract

Detecting text information in natural scene images is important and in great demand for applications such as assistive navigation, auxiliary reading, image retrieval, scene understanding, etc. Extracting text from natural scene images is a challenging problem because of complex backgrounds and large variations of text patterns such as font, color, scale, and orientation. In this paper, we explore a new framework to detect text strings with arbitrary orientations in complex natural scene images. The proposed framework of text string detection consists of two steps: A) Image partition to find text character candidates by using gradient magnitude and color uniformity of character components and B) Text character grouping to detect text strings by using structural analysis of text characters for example size of character, distance between neighboring characters, and character height and alignment. Also, two algorithms of text string detection are proposed: 1) adjacent character grouping method and 2) text line grouping method. The adjacent character grouping method finds the sibling groups of each character as a string and then merges them into text string. The text line grouping method performs Hough transform to find out fit text line of all the text candidates. The proposed methods outperform the state-of-the-art results on the public RRD which contains text only in horizontal orientation. Furthermore, the effectiveness of proposed method proves better to detect text strings with arbitrary orientations on the OSTD.

Keywords: Adjacent character grouping, Gradient Magnitude, Image Partition, Text string detection, Text line grouping.

I. INTRODUCTION

With the increasing popularity of practical vision systems and smart phones, text detection in natural scenes becomes a critical yet challenging task. Image-based text information serves as an important indicator in many applications such as scene understanding, content-based image retrieval, assistive navigation and automatic geocoding. To extract text information from camera-captured document images (i.e., most part of the captured image contains well organized text with clean background), many algorithms and commercial optical character recognition (OCR) systems have been developed [1], [2]. Also, different from document images, in which text characters are normalized into elegant poses and proper resolutions, natural scene images embed text in arbitrary shapes, sizes, and orientations into complex background, as shown in Fig. 1. [6]
It is impossible to recognize text in natural scene images directly because the off-the-shelf OCR software cannot handle complex background interferences and non-orienting text lines. Thus, we need to detect image regions containing text strings and their corresponding orientations. This is compatible with the detection and localization procedure described in the survey of text extraction algorithms [3], [4].

We want to detect texts of large variations in language, font, color, scale and orientation in complex scenes although text detection has been studied extensively in the past but the problem remains unsolved. The difficulties mainly come from two aspects: (1) the diversity of the texts and (2) the complexity of the backgrounds. On one hand, text is a high level concept but better defined than the generic objects; on the other hand, repeated patterns (such as windows and barriers) and random clutters (such as grasses and leaves) may be similar to texts, and thus lead to potential false positives. Detecting texts of arbitrary orientations in complex natural images has received much less attention and remains a challenge for most practical systems.

Section II describes overview of framework Section III describes proposed algorithms of image partition to extract text, Section IV introduces two grouping methods to extract text strings Section V shows implementation in multiple scales. Experiments and result analysis are described in Section VI. Conclusion of the paper is given in Section VII

II. OVERVIEW OF OUR FRAMEWORK

In this paper, we propose a new framework to extract text strings with multiple sizes and colors, and arbitrary orientations from scene images with a complex and cluttered background. Fig.2 [6] depicts the flowchart of our framework. The proposed framework consists of two main steps, given here.

**Step 1)** Image partition to find text character candidates based on gradient feature and color uniformity. In this step, we propose two methods to partition scene images into binary maps of non-overlapped connected components: gradient-based method and color-based method. A post processing is then performed to remove the connected components which are not text characters by size, aspect ratio, and the number of inner holes.

**Step 2)** Character candidate grouping to detect text strings based on joint structural features of text characters in each text string such as character sizes, distances between two neighboring characters, and character alignment. In this step, we propose two methods of structural analysis of text strings: adjacent character grouping method and text line grouping method.
III. IMAGE PARTITION

To extract text information from a complex background, image partition is first performed to group together pixels that belong to the same text character, obtaining a binary map of candidate character components. Two algorithms have been designed gradient-based partition algorithm and color-based partition algorithm, respectively.

A. GRADIENT-BASED PARTITION BY CONNECTING PATHS OF PIXEL COUPLES

The family of gradient-feature-based text localization methods assumes that text exhibits a strong edge against background, and therefore those pixels with high gradient values are regarded as good candidates for text regions. Although text characters and strings vary in font, size, color, and orientation, they are composed of strokes which are rectangle connected components with closed-width boundaries and uniform torso intensities as shown in Fig 3 [5]

Fig 3 (a) Image patch of a text character (b) Stroke is marked by a red character boundary and the red arrow denotes stroke orientations and blue arrow indicates the stroke width.

In our method, each pixel is mapped to the connecting path of a pixel couple, defined by two edge pixels and on an edge map with approximately equal gradient magnitudes and opposite directions, as shown in fig 4 [6] Each pixel couple is connected by a path. Then the distribution of gradient magnitudes at pixels of the connecting path is computed to extract candidate character component.
The character is modeled by distribution of gradient magnitudes and stroke size including width, height, and aspect ratio. The partitioned components are calculated from connecting path of pixel couple across the pixels with small gradient magnitudes.

On the gradient map, G map(p) and dp are used, respectively, to represent the gradient magnitude and direction at pixel p. We take an edge pixel from edge map as starting point and probe its partner along a path in gradient direction. If another edge pixel is reached where gradient magnitudes satisfy G map(p) - G map(q) < 20° and directions satisfy dq - (dp - (dq dp/dp)*π/6) < π/6 we obtain a pixel couple and its connecting path from p to q. This algorithm is applied to calculate connecting paths of all pixel couples. Fig. 3(b) marks all of the connecting paths shorter than 30 as white foreground. To perform the gradient-based partition, we employ gradient magnitude at each pixel on the connecting path and length of connecting path describing the size of connected component to be partitioned. The partition process is divided into two rounds. In the first round, the length range of connecting path is set as 0 < l < 30 to describe stroke width. For each pixel couple whose connecting path falls on this length range, we establish an exponential distribution of gradient magnitudes of the pixels on its connecting path, denoted by

\[ g(G_{mag}; \lambda) = \lambda \exp(-\lambda G_{mag}) \]  

(1)

Where the \( \lambda \) decay rate. A larger decay rate leads to faster falloff of gradient magnitudes on a connecting path. This means that the connecting path crosses a number of pixels with small gradient magnitudes on gradient map. This feature is consistent with the intensity uniformity inside the character strokes. Thus, the connecting path with greater decay rate is marked as white foreground representing candidate character component, as shown in Fig. 5[6]
To extract the complete stroke in rectangle shape, we start the second round to analyze the connecting paths along the stroke height (larger side). Since the aspect ratio of the rectangle stroke is no more than 6:1, we extend the length range of the connecting path to $0<\theta<180$. Then, we repeat the same analysis of gradient magnitudes for the connecting path not only falling on this length range but also passing through the regions of candidate character components obtained from the first round. At last, we perform morphological close and open as post processing to refine the extracted connected components, as shown in Fig. 6.

**Fig. 6.** Connecting path of a pixel couple The top row shows pixel couples in purple across the larger side of rectangle strokes. The bottom row presents the partitioned components obtained from the first round and the second round, respectively.

The refined connected components are taken as candidate character components. The gradient-based partition generates a binary map of candidate character components on black background. By the model of local gradient features of character stroke, we can filter out background outliers while preserving the structure of text characters.

**B. COLOR-BASED PARTITION BY COLOR REDUCTION**

In most scene images, text strings are usually composed of characters with similar colors. Thus, we can locate text information by extracting pixels with similar colors. To label a region of connected pixels with similar colors as a connected component, we develop color-based partition method. Inspired by [4], we perform color reduction by using color histogram and weighted K-means clustering through the following steps.

First, a canny edge detector is employed to obtain edge image. Second, we calculate color histograms of the original input image. To capture the dominant colors and avoid drastic color variations around edge pixels, only non edge pixels are sampled for color histogram calculation to obtain a set of sampled pixels $P$. Third, after mapping all of the pixels from spatial domain to RGB color space, as shown in Fig. 7(b), weighted K-means clustering is performed to group together the pixels with similar colors.
Fig. 7. (a) Scene image with multiple colors. (b) Color distribution in RGB space. (c) Four of the initial cube color clusters with radius $h$.

By using the initial mean point $P_i$ which is randomly selected from the sampled pixels and an initial radius $h$, color clusters in RGB color space is established [cf. Fig. 7(c)], covering any pixel whose color is close to $P_i$.

$$\text{Cover}(q|P_i) = \begin{cases} 1, & \text{IF } K1 \text{ is satisfied} \\ 0, & \text{otherwise} \end{cases}$$

$$\text{Cluster}(P_i) = \{q|\text{Cover}(q|P_i) = 1\}.$$ \hspace{1cm} (2)

$$P = \bigcup_i \text{Cluster}(P_i)$$ \hspace{1cm} (3)

Fig. 8. Some examples of color-based partition, where the left column contains original images and other columns contain the corresponding dominant color layers.
Each input image is partitioned to several color layers. A color layer that consists of only one foreground color on white background is a binary map of candidate character components. Then, connected component analysis is performed to label foreground regions of connected pixels. [6]

IV. CONNECTED COMPONENTS GROUPING

The image partition creates a set of connected components from an input image, including both text characters and unwanted noises. Observing that text information appears as one or more text strings in most natural scene images, we perform heuristic grouping and structural analysis of text strings to distinguish connected components representing text characters from those representing noises. Assuming that a text string has at least three characters in alignment, we develop two methods to locate regions containing text strings: adjacent character grouping and text line grouping, respectively. In both algorithms, a connected component C is described by four metrics: height (.), width (.), centroid (.), area (.) In addition, we use D(.) to represent the distance between the centroids of two neighbouring characters.

A. ADJACENT CHARACTER GROUPING

Text strings in natural scene images usually appear in alignment, namely, each text character in a text string must possess character siblings at adjacent positions. The structure features among sibling characters can be used to determine whether the connected components belong to text characters or unexpected noises. Here, five constraints are defined to decide whether two connected components are siblings of each other.

1) Considering the capital and lowercase characters, the height ratio falls between and $1/T_1$ and $T_1$
2) Two adjacent characters should not be too far from each other despite the variations of width, so the distance between two connected components should not be greater than $T_2$ times the width of the wider one.
3) For text strings aligned approximately horizontally, the difference between -coordinates of the connected component centroids should not be greater than $T_3$ times the height of the higher one.
4) Two adjacent characters usually appear in the same font size, thus their area ratio should be greater than $1/T_4$ and less than $T_4$.
5) If the connected components are obtained from gradient based partition as described in Section III-A, the color difference between them should be lower than a predefined threshold $T_5$ because the characters in the same string have similar colors.

According to the five constraints, a left/right sibling set $FL/FR$ is defined for each connected component C as the set of sibling components located on the left/right of C. For two connected components C and C’, they can be grouped together as sibling components if the above five constraints are satisfied. When C and C’ are grouped together, their sibling sets will be updated according to their relative locations, that is, when C is located on the left of C’, C’will be added into the right-sibling set of C, which is simultaneously added into the left-sibling set of C’. The reverse operation will be applied when C is located on the right of C’. When a connected component corresponds to a text character, the five constraints ensure that its sibling set contains sibling characters rather than the foliage, pane or irregular grain.
Fig. 9. (a) Sibling group of the connected component “r” where “B” comes from the left sibling set and “o” comes from the right sibling set. (b) Merge the sibling groups into an adjacent character group corresponding to the text string “Brolly?” (c) Two detected adjacent character groups marked in red and green, respectively.

For a connected component C, if both sibling sets are not empty and their size difference does not exceed 3, a sibling group is defined as the union of the two sibling sets and the connected component itself. At this point, each sibling group can be considered as a fragment of a text string. To create sibling groups corresponding to complete text strings, we merge together any two sibling groups and when the intersection contains no less than two connected components. Repeat the merge process until no sibling groups can be merged together. As shown in Fig. 9[6] the resulting union of connected components is defined as adjacent character group denoted by AG, which is a subset of the set of connected components.

**B. TEXT LINE GROUPING**

In order to locate text strings with arbitrary orientations, we develop text line grouping method. To group together the connected components which correspond to text characters in the same string which is probably nonhorizontal, we use centroid as the descriptor of each connected component. Given a set of connected component centroids, groups of collinear character centroids are computed, as shown below [6]

\[
M = \{ m \in S \text{ and } m = \text{centroid}(C) \} \\
L = \{ G \subseteq M, |G| \geq 3, \forall m_i, m_j, m_k \in G \}, \quad (4)
\]

They are character centroids and they are collinear where M denotes the set of centroids of all of the connected components obtained from image partition, and L denotes the set of text lines which are composed of text character centroids in alignment. Hough transform to describe the fitted line by \( <r_x, \theta_y> \), resulting in, where \( \theta_y \) is the equation of the fitted line in the Hough space. Thus, other collinear centroids along can be added into the end positions to form a complete text string increasingly. Fig. 10[6] illustrates the processing of fitted line refinement.
Fig. 10. (a) Centroids of connected components b) $D(m_A,m_B)$ approximately equals to $D(m_B,m_C)$ in text region while $D(m_A,m_P)$ is much larger than $D(m_P,m_Q)$ in background, where $D(.)$ represents Euclidean distance. (c) Three neighboring connected components in red share similar areas while those in green have very different areas. (d) Resulting fitted lines from centroids cascading. Red line corresponds to text region while cyan lines are false positives to be removed.

V. EXPERIMENTAL RESULTS

A.DATASETS

Two datasets are employed to evaluate the proposed algorithms. The first is the Robust Reading Dataset1 from ICDAR 2003. In this dataset, there are 509 images in total, in which 258 images are prepared for training and 251 images for testing. All of the text strings in this dataset are in horizontal. In our testing, we selected 420 images which are compatible with the assumption that a text string contains at least three characters with relatively uniform color.

Furthermore, to verify that text line grouping can detect text strings with arbitrary orientations, we collect 89 scene images with nonhorizontal text strings to construct the OSTD. The resolutions of most images are from 600 ×450 to 1280 ×960. This OSTD dataset contains colorful logos, indoor scenes, and street views.

B.PERFORMANCE EVALUATION

Fig. 12. Performance evaluation of the four combinations of partition and grouping on the Robust Reading Dataset, where the box presents the average time of text string detection in each scene image.
VI. RESULTS

The experimental results on the Robust Reading dataset are illustrated in Fig. 12[6] where blue bars denote results of GA, cyan bars denote results of CA, yellow bars denote results of GT, and red bars denote results of CT. The average time of text string detection is presented in the upper boxes. By comparison with the algorithms presented in the text locating competition in ICDAR, the precision of our algorithm achieves the first rank while the recall and -measure is comparable with the algorithms with the high performance, as shown in Table III 6]

TABLE III
COMPARISON BETWEEN OUR ALGORITHM AND THE TEXT DETECTION
ALGORITHMS PRESENTED IN [21] AND [22] ON THE ROBUST READING
DATASET

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>( f )-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>0.71</td>
<td>0.62</td>
<td>0.62</td>
</tr>
<tr>
<td>H. Becker</td>
<td>0.62</td>
<td>0.67</td>
<td>0.62</td>
</tr>
<tr>
<td>A. Chen</td>
<td>0.60</td>
<td>0.60</td>
<td>0.58</td>
</tr>
<tr>
<td>Ashida</td>
<td>0.55</td>
<td>0.46</td>
<td>0.50</td>
</tr>
<tr>
<td>HWDavid</td>
<td>0.44</td>
<td>0.46</td>
<td>0.45</td>
</tr>
<tr>
<td>Q. Zhu</td>
<td>0.33</td>
<td>0.40</td>
<td>0.33</td>
</tr>
<tr>
<td>Wolf</td>
<td>0.30</td>
<td>0.44</td>
<td>0.35</td>
</tr>
<tr>
<td>J. Kim</td>
<td>0.22</td>
<td>0.28</td>
<td>0.22</td>
</tr>
<tr>
<td>Todoran</td>
<td>0.19</td>
<td>0.18</td>
<td>0.18</td>
</tr>
<tr>
<td>N. Ezaki</td>
<td>0.18</td>
<td>0.36</td>
<td>0.22</td>
</tr>
</tbody>
</table>
Some example results of text string detection on the Robust Reading Dataset are presented in Fig. 14[6]. Instead of using the rectangle line to denote the borders of text regions.

![Text String Detection Examples](image)

The experiment on the OSTD demonstrates that the text line grouping in our framework is able to detect the text strings with arbitrary orientations, as shown in Fig. 15[6]. Without the orientation of the text line, the multiple-line text leads to text line grouping among the characters of different text strings.

![Text Line Grouping Examples](image)

Fig. 18[6] depicts some examples that our method cannot handle to locate the text information because of very small size, overexposure, characters with nonuniform colors or fade, strings with less than three character members, and occlusions caused by other objects such as wire mesh.

![Text Locating Issues](image)
VII. CONCLUSION

Due to the unpredictable text appearances and complex backgrounds, text detection in natural scene images is still an unsolved problem to locate text regions embedded in those images, we propose a new framework based on image partition and connected components grouping. Structural analysis is performed from text characters to text strings. Experiments show that color-based partition performs better than gradient-based partition, but it takes more time to detect text strings on each color layer. The text line grouping is able to extract text strings with arbitrary orientations. The combination of color-based partition and adjacent character grouping (CA) gives the best performance, which outperforms the algorithms presented in ICDAR.

REFERENCES

[5] Localizing Text in Scene Images by Boundary Clustering, Stroke Segmentation, and String Fragment Classification by Chucai Yi, Student Member, IEEE, and Yingli Tian, Senior Member, IEEE