21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

Text Detection in Natural Scenes Using Gradient Vector Flow-Guided Symmetry

Trung Quy Phan, Palaiahnakote Shivakumara and Chew Lim Tan
School of Computing, National University of Singapore
{phanquyt, shiva, tancl}@comp.nus.edu.sg

Abstract

In this paper, we propose a novel method for text detection in natural scenes. Gradient Vector Flow is first used to extract both intra-character and inter-character symmetries. In the second step, we group horizontally aligned symmetry components into text lines based on several constraints on sizes, positions and colors. Finally, to remove false positives, we employ a learning-based approach which makes use of Histogram of Oriented Gradients feature. The main advantage of the proposed method lies in the use of both the text features and the gap (i.e., inter-character) features. Existing techniques typically extract only the former and ignore the latter. Experiments on the benchmark ICDAR 2003 dataset show the good detection performance of our method on natural scene text.

1. Introduction

Detecting text in natural scene images refers to the problem of identifying the positions of texts that appear on, e.g., bill boards and road signs. It has a wide range of applications, e.g., translating signs into different languages, aiding the visually-impaired and helping robots navigate the environment.

Although many methods have been proposed over the past years, scene text detection is still a challenging problem due to the unconstrained appearance of text, in terms of sizes, colors, backgrounds and alignments.

There have been previous works on text detection in natural scenes, as surveyed in [3]. We review some of the recent works, which belong to two main approaches, below. The gradient-based approach typically performs edge detection to locate potential text regions. Epshtein et al. [1] proposed the stroke width transform, based on the observation that character strokes in the same text line often have almost constant thickness. A similar idea was used in [4] to identify “pixel couples”, pairs of pixels that had similar gradient magnitudes and opposite directions. Although these methods are fast, they produce many false positives for images with complex backgrounds.

To overcome this problem, the texture-based approach considers text as a special texture. These methods extract features such as intensity and gradient [2], and Histogram of Oriented Gradients (HOG) [5]. Classifiers such as Support Vector Machines (SVM) are then used for text/non-text classification. Although this approach is more robust against complex backgrounds, it requires a large amount of training data and is computationally expensive.

Most methods in the literature only extract text features, e.g., stroke width. There has been very little attention to gap (i.e., inter-character) features. In this paper, we introduce a framework to capture both types of features for more effective text detection.

2. Proposed approach

The proposed method consists of three steps: text symmetry identification, text line grouping and false positive elimination.

2.1. Text symmetry identification

Within a text line, there are both *intra-character symmetry* and *inter-character symmetry*. The first type of symmetry comes from the fact that most characters have an inner contour and an outer contour, and there is a correspondence between the two. The second type of symmetry refers to the property that within a gap between consecutive characters, there is symmetry between the two outer contours of the two characters.

We propose to use Gradient Vector Flow (GVF) [6]

978-4-9906441-1-6 ©2012 IAPR

in a novel way to detect both intra-character and inter-character symmetries. The main idea of GVF is to propagate the gradients into homogenous regions. The propagation is done by minimizing the following energy functional:

$$\mathcal{E} = \iint \mu\left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right) + |\nabla f|^2 |g - \nabla f^2| \, dxdy \quad (1)$$

where $f(x,y)$ is the edge map of the input image and $g(x,y) = \left(u(x,y), v(x,y)\right)$ is the GVF field (Figure 1a and b).

It is observed that the GVF "arrows" are attracted to the edges. Hence, we define the symmetry points as those where the arrows point away from each other. This condition ensures that the symmetry points are at the middle of two edges (Figure 1c).

For example, pixel (x,y) is classified as a *vertical* symmetry point if:

$$\begin{cases} u(x,y) < 0 \\ u(x+1,y) > 0 \\ angle\left(g(x,y), g(x+1,y)\right) > \theta_{min} \end{cases} \quad (2)$$

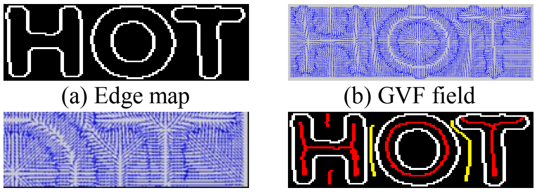
where $angle(.)$ returns the angle between two vectors. Intuitively, the GVF vector at pixel (x,y) should point to the left hand side, the GVF vector at pixel $(x+1,y)$ should point to the right hand side, and the angle between these two vectors should be sufficiently large, e.g., greater than $\pi/6$.

We also consider three other directions: horizontal, left-diagonal and right-diagonal. The symmetry conditions for these directions can be derived in a similar manner as in (2). The rationale for using these four directions is that they capture the major directions of character strokes.

Figure 1d shows the symmetry points of a sample text line. Both intra-character symmetry (red) and inter-character symmetry (yellow) are successfully detected. It is observed that the symmetry points belonging to the same character or the same gap are often connected to each other. Hereafter, we refer to them as *symmetry components*.

Figure 2 shows the result of applying the same symmetry detection process on a whole image. An important observation is that if we combine GVF with different edge operators, e.g., Sobel and Canny, the symmetry components of text regions are almost the same. This is because the character contours in the two cases are very similar. On the other hand, the symmetry components in the background are very different. The reason is that the Canny edge map contains finer details in these regions than the Sobel edge map (Figure 2b and c).

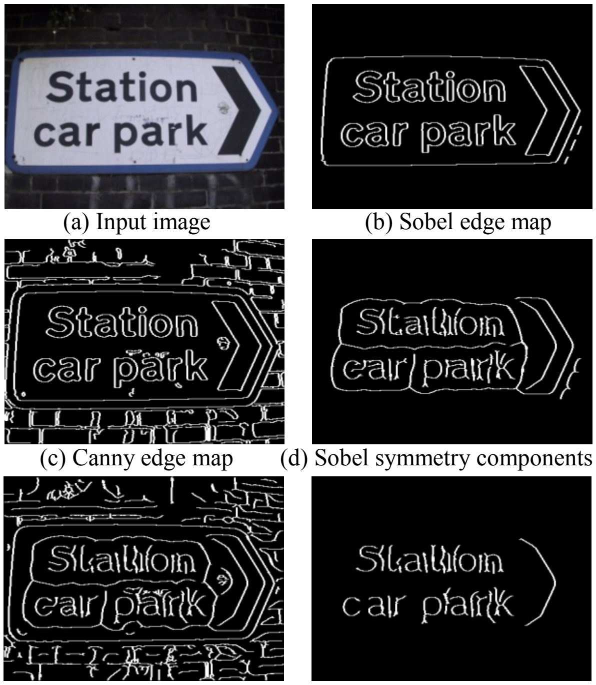
Hence, we use a relaxed intersection to suppress symmetry components in the background while preserving those in text regions: If a Canny symmetry component overlaps with a Sobel symmetry component for at least 50%, the former is retained; otherwise, it is discarded. Figure 2f illustrates that most symmetry components in the background are removed after the intersection. (For this sample image, it may seem that using Sobel GVF alone is sufficient. However, we have found that combining both Canny GVF and Sobel GVF gives more reliable text candidates, especially for texts of small font sizes.)



(a) Edge map (b) GVF field

(c) Zoomed-in GVF field (d) Symmetry components

Figure 1. GVF helps to detect both text symmetry components (red) and gap symmetry components (yellow).



(a) Input image (b) Sobel edge map

(c) Canny edge map (d) Sobel symmetry components

(e) Canny symmetry components (f) Text candidates

Figure 2. By applying GVF with two different edge maps, the proposed method is able to identify text candidates.

Note that in an earlier work for character segmentation [7], we used a similar GVF formulation. However, that work only utilized the inter-character symmetry, due to the nature of the segmentation task. In contrast, the present work exploits both intra-character and inter-character symmetries.

2.2. Text line grouping

At the end of the previous step, the text candidates have been identified. However, they are not grouped into text lines yet. Therefore, we will assign them to groups such that each group contains one or more symmetry components with similar properties. The proposed method scans the image (from top to bottom, left to right) for an unassigned component and assigns it to a new group. After that, we grow the group by looking for components in the neighborhood which have similar properties. If the group cannot be grown any further, we start another group and again look for an unassigned component. The algorithm ends when all components are each assigned to a group.

More specifically, let g and c be a group and a component, respectively. The following conditions are checked before we grow g to include c:

- **Height**. Characters in the same line are expected to have similar heights. It is required that:

$$T_1 \leq \frac{Height(c)}{MedianHeight(g)} \leq 1/T_1 \qquad (3)$$

- **Position**. We assume that most text lines are horizontally aligned. Therefore, the y-coordinates of the characters should not be too far from each other:

$$\begin{aligned}|Centroid(c).y - MedianCentroid(g).y| \\ \leq T_2 \times MedianHeight(g)\end{aligned} \qquad (4)$$

- **Color**. Many text lines are monochrome. This is especially true for those that are designed to be easy to read, e.g., sign boards. Thus:

$$RGBColorDistance(c, g) \leq T_3 \qquad (5)$$

Based on the ICDAR 2003 training set, the parameter values are set as follows: $T_1 = T_2 = 0.4$ and $T_3 = 70$.

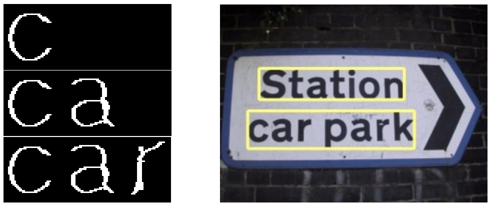
Figure 3a shows the first three iterations of the grouping process for the second text line in Figure 2a. In Figure 3b, groups with less than three symmetry components are discarded, as most text lines are expected to have at least this many characters [1].

2.3. False positive elimination

The above two steps sometimes detect non-text regions that happen to satisfy the symmetry and grouping constraints. To remove these false positives, we propose a learning-based approach where HOG [8] and SVM are used for feature extraction and classification, respectively.

SVM is trained on the ICDAR 2003 training set. By using a 48 × 48 window, we extract 11,600 positive patches (sampled from ground truth text regions) and 14,100 negative patches (randomly sampled from non-text regions).

To perform text verification, each detected box in the previous section is resized to a fixed height of 48 pixels. We then slide a window horizontally and compute the SVM confidence score for each position. To combine these scores, we use the weighted average proposed in [9] where windows nearer to the center of the box are given higher weights. A detected box is classified as text if its combined score is non-negative.



(a) Text line grouping (b) Output of Figure 2a

Figure 3. The grouping process for the second text line in Figure 2a (a) and the final output (b).

Figure 4. Sample false positives that are successfully removed by using HOG-SVM.

As noted in [1], the typical false positives in scene images are repeating patterns such as windows, bricks and leaves. By using HOG features, we are able to remove some of these patterns (Figure 4). Hence, texture analysis helps to improve the precision.

3. Experimental results

For performance evaluation, we used the ICDAR 2003 dataset [13]. The training and test sets contain 250 and 249 images, respectively.

3.1. Sample text detection results

Figure 5 shows the text detection results of the proposed method for a variety of images. By using the symmetry feature, it is able to locate the text lines accurately with tight bounding boxes, despite the reflection, complex backgrounds and strong highlights of the input images. It is also robust against repetitive, textured regions due to the use of HOG features.

3.2. Quantitative evaluation

To facilitate comparison with other methods, we used the same performance measures as those in the ICDAR 2005 competition: precision, recall and f-measure [10]. To conform to the ground truth of this dataset, we segment text lines into separate words by analyzing their vertical projection profiles.

Table 1 shows that the proposed method achieves the highest f-measure. Compared to the state-of-the-art method [1], our method achieves a better recall.

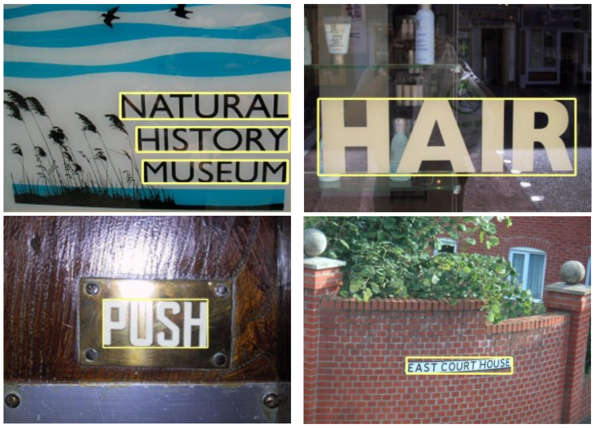


Figure 5. Sample detection results of the proposed method.

Table 1. Performance on the ICDAR 2003 test set

Method	Precision	Recall	f
Proposed method	**0.68**	**0.66**	**0.67**
Epshtein et al. [1]	0.73	0.60	0.66
Lee et al. [11]	0.69	0.60	0.64
Minetto et al. [12]	0.63	0.61	0.61
1st ICDAR 2005	0.62	0.67	0.62
2nd ICDAR 2005	0.60	0.60	0.58

Method [1] extracts only character feature, i.e., stroke width. However, for some images, the character edges are broken (especially at the joints of different strokes) and thus the stroke width estimate is not reliable.

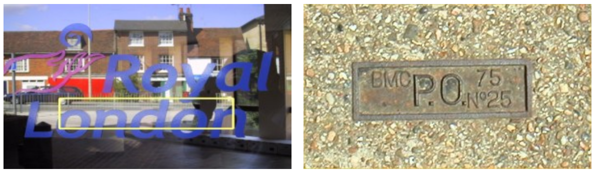
On the other hand, our method exploits both the text feature and the gap feature due to the use of GVF symmetry components. Even if the character edges are broken, the regular spaces between consecutive characters may still be visible, and are extracted by our method. Hence, in these cases, it is the inter-character symmetry that enables the proposed method to detect the difficult text lines.

The precision of our method is lower than that of [1] because the symmetry feature is sometimes detected for text-like patterns in the background, e.g., regular structures on buildings. We plan to address this problem in future research, e.g., by using a character recognition module to discard these false positives.

Some failure cases are shown in Figure 6. Due to low contrast (a), small font size, and similar foreground and background colors (b), our method is not able to fully detect the symmetry components. (Note that for (b), our method does pick up "PO" but it fails the constraint of having at least three characters.)

4. Conclusion and future work

We have proposed a method for detecting natural scene text, based on the intra-character and inter-character symmetries. GVF is used to extract the symmetry points in both cases. Grouping based on the similarity in sizes, positions and colors helps to merge horizontally aligned symmetry components into text lines. Finally, HOG features are used for false positive elimination. In the future, we plan to extend the current work to multi-oriented text lines.



(a) (b)

Figure 6. Failure cases of the proposed method.

Acknowledgment

This research is supported in part by A*STAR grant R252-000-402-305.

References

[1] B. Epshtein, E. Ofek and Y. Wexler. Detecting Text in Natural Scenes with Stroke Width Transform. In Proc. CVPR 2010.

[2] X. Chen and A. L. Yuille. Detecting and Reading Text in Natural Scenes. In Proc. CVPR 2004.

[3] J. Liang, D. Doermann and H. Li. Camera-based Analysis of Text and Documents: A Survey. *International Journal on Document Analysis and Recognition*, 7(2), 2005, pp. 84–104.

[4] C. Yi and Y. Tian. Text String Detection from Natural Scenes by Structure-based Partition and Grouping. *IEEE Transactions on Image Processing*, 20(9), 2011, pp. 2594–2605.

[5] K. Wang, B. Babenko and S. Belongie. End-to-End Scene Text Recognition. In Proc. ICCV 2011.

[6] C. Xu and J. L. Prince. Snakes, Shapes, and Gradient Vector Flow. *IEEE Transactions on Image Processing*, 7(3), 1998, pp. 359–369.

[7] T. Q. Phan, P. Shivakumara, B. Su and C. L. Tan. A Gradient Vector Flow-Based Method for Video Character Segmentation. In Proc. ICDAR 2011.

[8] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In Proc. CVPR 2005.

[9] D. Chen, J.-M. Odobez and H. Bourlard. Text detection and recognition in images and video frames. *Pattern Recognition*, 37(3), 2004, pp. 595– 608.

[10] S. M. Lucas. ICDAR 2005 Text Locating Competition Results. In Proc. ICDAR 2005.

[11] S. Lee, M. S. Cho, K. Jung and J. H. Kim. Scene Text Extraction with Edge Constraint and Text Collinearity. In Proc. ICPR 2010.

[12] R. Minetto, N. Thome, M. Cord, J. Fabrizio and B. Marcotegui. Snoopertext: A multiresolution system for text detection in complex visual scenes. In Proc. ICIP 2010.

[13] ICDAR 2003 Dataset. http://algoval.essex.ac.uk/icdar/Datasets.html