2011 International Conference on Document Analysis and Recognition

Dot Text Detection Based on FAST Points

Yuning Du, Haizhou Ai
Computer Science & Technology Department
Tsinghua University
Beijing, China
dyn10@mails.tsinghua.edu.cn, ahz@mail.tsinghua.edu.cn

Shihong Lao
Core Technology Center
Omron Corporation
Kyoto, Japan
lao@ari.ncl.omron.co.jp

***Abstract*—In this paper, we propose a method for dot text detection based on FAST points. This problem is different from general scene text detection because of discontinuous text stroke. Unlike many other methods which assume that text is horizontally oriented, our method is able to deal with slant dot text. We extract interesting patches from FAST points and define four features based on the stroke and gray value similarity of dot text to describe a patch. Then, we generate some candidate regions from these patches and utilize SVM to filter out non-dot text ones with the first and second order moments of FAST points in them. Experimental results show that the proposed method is effective and fast to detect dot text.**

***Keywords*-dot text detection; slant text; SVM**

I. Introduction

Texts in images carry very important information on the contents of images. If text can be extracted effectively, it can certainly help computer to better understand the contents of images. In this paper, we focus on a particular kind of texts which consist of dots (called dot text for short), see Fig. 1. These texts often appear in our daily life, such as the product data of foods and medicines, the product number of goods, advertisement lamp box with LED and so on. Extracting this kind of text information is meaningful for factory automation, image retrieval and so on.

In general, text detection is the first step for a text information extraction system which is of critical importance to the success of the entire system. Recently, many researchers pay attention to text detection in natural scene. In such situation, size, color and orientation of text are variable. Meanwhile, the background of text is complex due to illumination and texture. Therefore, text detection in natural scene is a very challenge problem. The methods of text detection in natural scene can be classified in two groups: region-based methods and texture-based methods [1]. Region-based methods use similarity criterions of text, such as color, size, stroke width, edge and gradient information. The pixels with similar properties are gathered. Then the resulting connected components (CCs) which are non-texts are filtered out by some heuristic geometric hypothesis. The methods in [2] [3] fall into this category. However, this kind of method couldn't

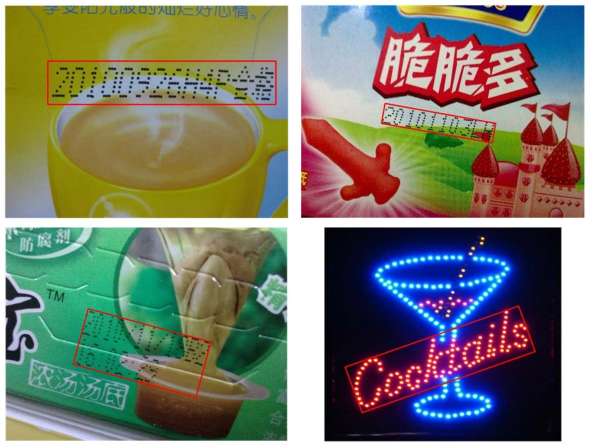

Figure 1. Some samples of dot text images and red rectangles represent the results of our method.

effectively deal with dot text detection, since very small candidate regions are considered as non-text regions in its processing. Texture-based methods are similar as the method of face detection in which a classifier between text and non-text is trained based on texture features, sliding window search is used to extract candidate regions and the results are merged for final outputs. The methods in [4] [5] fall into this category. Texture-based methods may still work in dot text detection, but they could be time consuming. Besides, most existing methods suppose that text is in a horizontal line form and they have difficulty in dealing with slant text detection.

In this paper, we consider the problem of dot text detection based on the observation that there are lots of FAST points around dot text and develop a new method based on their spatial distribution to detect dot text. FAST is short for "Features from Accelerated Segment Test"[6]. A point whose surround intensity has wide variations is likely to be a FAST point. These FAST points maybe form around dot text easily. Moreover, the time consumption of extracting FAST points is very low. For example, extracting approximately 500 features on a field of 768×288 by a 850 MHz Pentium III processor needs only about $5ms$ [6]. Therefore, we develop our dot text detection algorithm based on FAST points that combines region-based methods and texture-based methods that could deal with slant dot text detection as illustrated in Fig. 1.

The organization of this paper is as follows. In section II, we present our dot text detection method. The experimental results are given in section III. Section IV concludes the paper and talks about our future work.

II. Proposed Detection Method

The proposed method consists of 3 steps, Fig. 2 shows the flowchart of our method:

1) *Extract patches:* Patches are extracted from the neighborhood of FAST points. To describe a patch, four features are defined and some non-dot text patches are eliminated based on those features.
2) *Generate candidate regions:* The patches are merged based on their spatial relationship and features' similarity to generate some candidate regions.
3) *Filter out non-text regions:* Some candidate regions are non-dot text ones, so we utilize the spatial distribution of FAST points in candidate regions to filter out those ones by SVM classification.

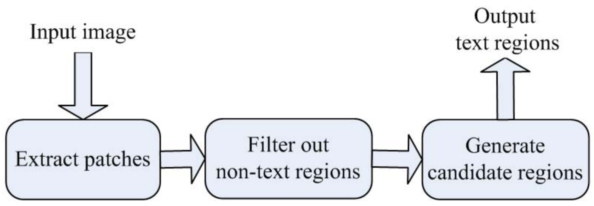

Figure 2. The flowchart of proposed method.

A. Extract Patches

We use a FAST point detector [7] to extract FAST points. A candidate point p as a FAST point if there exists a set of n contiguous pixels in its neighborhood circle which are all brighter than I_p+t, or all darker than I_p-t, where I_p is the intensity of the candidate point and t is a threshold. In this paper, all operations are on grey images. The two parameters of FAST point detector are set to $n = 9$, $t = 15$. Besides, in order to keep the spatial distribution of dot text in FAST point image, the points are extracted without non-maximal suppression. Most FAST points are around dot text and they keep the spatial distribution of dot text as seen in Fig. 3b, so the information of FAST points is enough for dot text detection.

In order to obtain more information about dot text, patches are extracted from the surrounding of FAST point. The procedure of patch extraction is as follows:

a) All FAST points are unlabelled.
b) Select an unlabelled point, the neighborhood of the point (size is $N \times N$) is regarded as a patch. The points which are unlabelled in the patch are labelled.
c) Repeat 2), until all points are labelled.

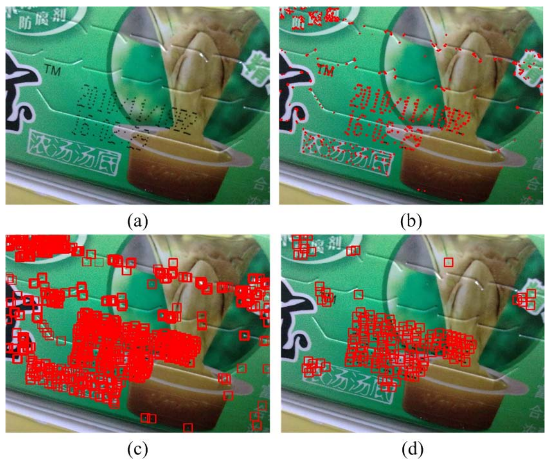

Figure 3. Extract patches. (a) is the original image. The red dots in (b) shows the FAST points of (a). Patches extracted from FAST points are represented as red rectangles in (c). (d) shows the patches after eliminating non-dot text ones.

Because the stroke width range of dot text is 3 to 15 pixels in our database, N is set to 31 so that one patch contains more than two dots. If the stroke width is larger than the limits, scale space method can be used.

Having extracted patches, four features which are inspired from the stroke width extraction in [2] and the character energy definition in [3] are defined to describe a patch: averaged difference of gradient directions (G_{angle}), the ratio between the mode and the mean of stroke width histogram (R_{sw}), the stroke width value (V_{sw}) and the gray value (V_{gray}). The definition of these features is based on the stroke and gray value similarity of dot text.

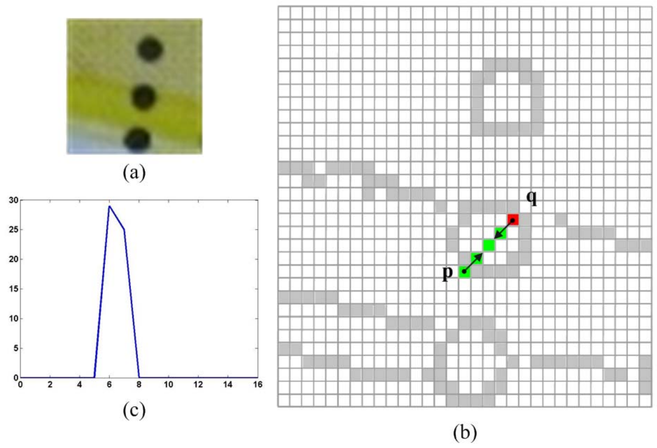

Figure 4. Features extraction of patch. (a) A patch of dot text. (b) p is an edge point, q is the corresponding point of p. The segment $[p, q]$ represents an approximate stroke of dot text, see green pixels. (c) The stroke width histogram of (a).

Fig. 4 illustrates features extraction of patch. In order to extract the features, the strokes of dot text are recovered based on edge information. We compute edges in the image by Canny edge operator [8] as shown in Fig. 4b. Given an edge point p in a patch, another edge point q in the patch is called as the corresponding point of p, if q satisfies following conditions (see Fig. 4b):

- q is on the gradient direction of p.
- Let θ_x represents the direction of the gradient at point x and $\theta_x \in [-\pi, \pi)$, the difference between θ_q and $-\theta_p$ is no greater than $\pi/2$, i.e. $|\theta_q - (-\theta_p)| \leq \pi/2$.

Obviously, if p is on the boundary of dot text, the direction of the gradient at p and q are opposite and the difference between θ_q and $-\theta_p$ should be very small. For a patch, not all the edge points in the patch have corresponding point as seen in Fig. 4b and the edge points which have corresponding points form a set E. The averaged difference of gradient directions is defined as:

$$G_{angle} = \frac{\sum\limits_{p \in E} |\theta_q - (-\theta_p)|}{N_E} \quad (1)$$

where N_E is the number of point in E and q is the corresponding point of p. The smaller G_{angle} of a patch means the patch more likely to be a part of dot text (called dot text patch for short). When G_{angle} of a patch is larger than T_{angle}, this patch is more likely to be a non-dot text patch and needs to be eliminated, T_{angle} is a threshold.

The segment $[p, q]$ represents an approximate stroke of dot text and the stroke width d_p is defined as the number of pixels along the segment, see Fig. 4b green pixels. Let $D = \{d_p | p \in E\}$ and $hist(D)$ is the stroke width histogram of D as seen in Fig. 4c. $hist(D, j)$ is the value of the histogram of D at the $j^{th} (1 \leq j \leq J)$ bin, where J is the maximum length of stroke length d_p. For dot text patch, all approximate strokes have similar stroke width. The mode and the mean of stroke width histogram should be very close. The mode of stroke width histogram is defined as:

$$SW_{mode} = \max_{1 \leq j \leq J} (hist(D, j)) \quad (2)$$

and the mean of stroke width histogram is defined as:

$$SW_{mean} = \sum_{j=1}^{J} j \cdot \frac{hist(D, j)}{\sum\limits_{j=1}^{J} hist(D, j)} \quad (3)$$

Then the ratio between the mode and the mean of stroke width histogram is defined as:

$$R_{sw} = \frac{\min(SW_{mode}, SW_{mean})}{\max(SW_{mode}, SW_{mean})} \quad (4)$$

The higher R_{sw} of a patch means the patch more likely to be a part of dot text. When R_{sw} of a patch is smaller than T_{rsw}, this patch is more likely to be a non-dot text patch and needs to be eliminated, T_{rsw} is a threshold. Fig. 3c shows the patches which are extracted from the neighborhood of FAST points and Fig. 3d shows the patches which are remained after eliminating non-dot text ones by G_{angle} and R_{sw} features. Comparing Fig. 3d with Fig.3c, we can see G_{angle} and R_{sw} can eliminated some non-dot text patches effectively.

Besides, the stroke width of a patch can be represented by its mode of stroke width histogram, i.e. $V_{sw} = SW_{mode}$. In general, for dot text patches, the gray values of pixels along a stroke are similar, thus the gray value of the stroke can be represented by the median gray value of these pixels. All strokes also have similar gray values. Therefore, the gray value of a patch is defined as:

$$V_{gray} = median(A_E) \quad (5)$$

$$A_E = \{median(B_p) | p \in E\} \quad (6)$$

where B_p is a set which consists of the gray values of pixels along the segment $[p, q]$ and q is the corresponding point of p, $median(X)$ is the median value of set X.

If patches are belong to the same dot text, their V_{sw} and V_{gray} features are similar. So V_{sw} and V_{gray} can be used to merge patches and generate candidate regions.

B. Generate Candidate regions

In this step, some candidate regions are generated with the patches above. One patch can be considered as a vertex of a non-directed graph. Two patches are connected if they satisfy following conditions: They are spatially adjacent and the distance between their centers should be smaller than T_{dist}; Their stroke width values are proximate and the difference between their stroke width values should be smaller than T_{vsw}; Their gray values are also proximate and the difference between their gray values should be smaller than T_{vgray}. T_{dist}, T_{vsw} and T_{vgray} are thresholds. When the non-directed graph is constructed, the candidate regions will be generated from same connected component. In general, text region is represented with a horizontal rectangle, see black rectangle in Fig. 5. However, this representation isn't suitable to describe slant text and extract FAST points' spatial distribution. Therefore, the rectangle needs to be adjusted. Note that, the FAST points of a candidate region only are ones in patches which consist corresponding connected component.

According to the orientation of text, the text is classified in three types: non-rotation text (NR), counterclockwise-rotation text (CCR) and clockwise-rotation text (CR), see Fig. 5. The rotation angle's range is $[0, \pi/2)$. The type of text can be inferred by FAST points' number in the neighborhood of four vertexes (the neighborhood size is the same as a patch). If the number of FAST points in the neighborhood of vertex A and C are zero, not all B and D are zero, the text is CCR text (see Fig. 5b); If the number of FAST points in the neighborhood of vertex B and D are zero, not all A and C are zero, the text is CR text (see Fig. 5c); In the other cases, the text is NR text (see Fig. 5a).

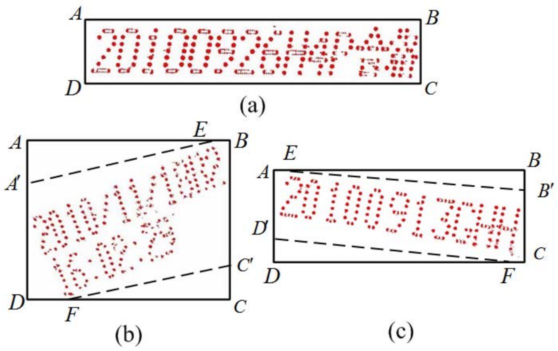

Figure 5. Three types of dot text. (a) is NR text, (b) is CCR text, (c) is CR text.

There is no need to adjust rectangle of NR text. For CCR text, the adjustment procedure is as follows:

1) *Estimate the rotation direction of text:* Assume E is the projection point onto AB of the nearest FAST point to the top rectangle border AB among all the FAST points, similarly F is onto DC . Move vertex A along the left rectangle border AD . At the same time, move C along CB. At the beginning, the number of FAST points in $\triangle AEA'$ and $\triangle CFC'$ are zero. When the number of FAST points in $\triangle AEA'$ and $\triangle CFC'$ are more than a threhold T_{num}, $\theta = \angle AEA' = \angle CFC'$ is the rotation direction of text.
2) *Estimate the position of rectangle vertex:* Given the rotation direction of text θ, the FAST point image of candidate region is rotated clockwisely around its center by θ degree. We called the minimum bounding rectangle of the result as its rotated candidate region. The position of rectangle vertex can be estimated with the vertex of the rotated candidate region.

The adjustment procedure for CR text is similar as CCR text. Fig. 6 gives some rotated candidate regions. From the figure, the FAST points' spatial distribution of dot text regions is different from non-dot text ones, this information can be used to filter out non-dot text regions. Because there is no evident difference of dot text regions which contain one or two characters and non-dot text ones, we assume dot text regions always contain more than two characters.

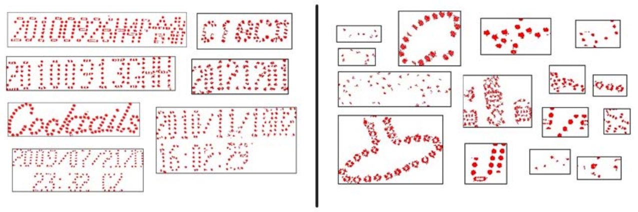

Figure 6. Some rotated regions. Regions in left are dot text regions and ones in right are non-dot text regions.

C. Filter out non-text regions

In this step, non-dot text regions can be filtered out from candidate regions above by support vector machines (SVM) [9] classification.

Since the spatial distribution of FAST points is different between dot text and non-dot text, we will consider the feature extraction from the first and second order moments of FAST points in rotated candidate regions from 3 block patterns, see Fig. 7. These block patterns form 5 regions. For each region, the average and variance of distance between FAST points and the region's border are considered as features. Therefore, the number of features is $1 \times 4 \times 2 + 4 \times 3 \times 2 = 32$. For a rectangle region, such as region1, the features of this region are normalized by dividing corresponding region width or region height. For a triangle region, such as region2, the features of this region are normalized by dividing the distance between the border and the opposite vertex of it.

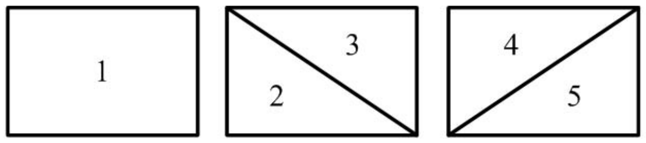

Figure 7. Block patterns.

III. Experimental Results

Up to now, we haven't found other work on dot text detection in the literature. In order to evaluate the performance of our method, we grabbed 475 images with dot text from goods, medicines and Google image. 355 images are used as training data and other 120 ones are used as testing data. The typical size of images is 640×480. The parameters for non-dot patches elimination are set to $T_{angle} = \pi/6$, $T_{rsw} = 0.85$, for candidate regions generation are set to $T_{dist} = 35$, $T_{vsw} = 1$, $T_{vgray} = 15$ and for candidate regions adjustment is set to $T_{num} = 10$. All parameters in our method are empirically determined. In order to accommodate both bright text on dark background, we reverse the gray of image and apply the method twice.

Table I
Dataset Description for Dot Text Classifier

	Training data		**Testing data**	
	Dot text region	***Non-dot text region***	***Dot text region***	***Non-dot text region***
numbers	354	476	100	100
total numbers	830		200	

Table I summarizes the details of the datasets used for training classifier to discriminate dot text regions from non-dot text ones, see Fig. 6. All experiments are carried out on a PC with a CPU of Core 2 Quad 2.5 GHz and the program is implemented in Visual Studio 2005.

Similar as [10], the following categories are defined for each detected region and the performance evaluation is at the region level.

- Truly Detected Region (TDR): A detected region overlaps at least 90% of the ground-truth region.
- False Detected Region (FDR): A detected region that does not contain dot text or misses more than 10% of the ground-truth region.
- Actual Text Regions (ATR): The true text regions.

The performance measures are defined as follows:

- $Recall(R) = TDR/ATR$
- $Precision(P) = TDR/(TDR + FDR)$
- $F\text{-}measure(F) = 2 \times P \times R/(P + R)$

The performance measures are as follows: $R = 88.3\%$, $P = 82.9\%$, $F = 85.5\%$. Besides, we evaluate the time consumption of the proposed method with 120 testing images. The average time consumption of our method is about $0.06s$. These results demonstrate that our method is effective in detecting dot text. Fig. 8 gives some additional results of our method.

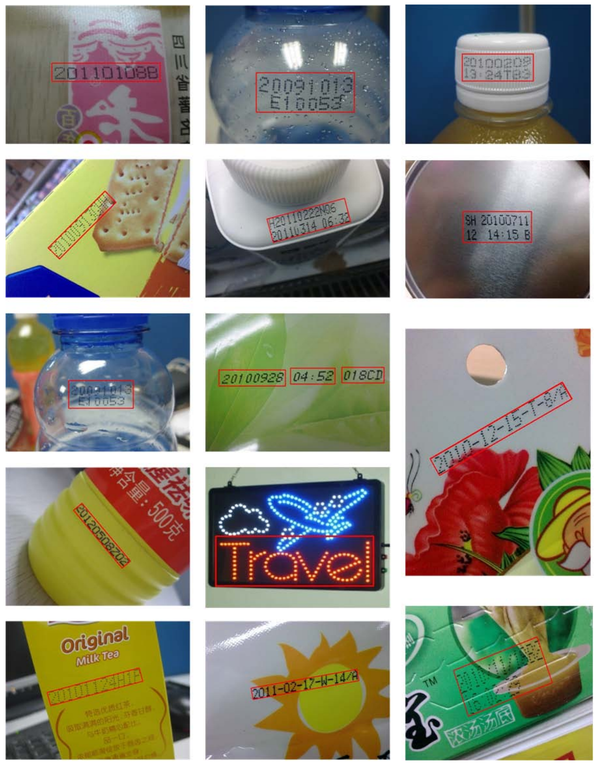

Figure 8. Additional examples of dot text detection. The results are marked with red rectangle.

In addition, the shortages of the proposed method are illustrated in Fig. 9. (1) The proposed method fails to detect dot texts whose surround contains texts with similar stroke, see Fig. 9a. We assume that text regions contain more than two characters, regions contain less than two characters are removed as false positives. (2) Some texts that are not dot texts have similar FAST points spatial distribution as dot texts, so our method consider these candidate regions as positive ones, as shown in Fig. 9b.

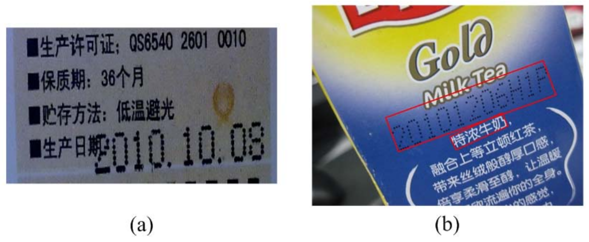

Figure 9. Failures of the proposed method. (a) missing detections. (b) false positives.

IV. Conclusion and Future Work

In this paper, we propose an effective and fast method for dot text detection based on FAST points. Firstly, patches are extracted from FAST points and some non-dot patches are eliminated from patch features. Then candidate regions are generated with the features' similarity of the remaining patches. Finally, non-text regions are filtered out by SVM classification. Experiment results demonstrate the impressive performance of our method.

Since dot text is a kind of text, the four proposed features maybe fit arbitrary text. Besides FAST points, edge points can also be considered as a kind of interesting points. In the future, we would like to extend our method to arbitrary scene text detection based on edge points.

References

[1] K. Jung, K. Kim, and A. Jain, "Text information extraction in images and video: a survey," *Pattern recognition*, vol. 37, no. 5, pp. 977–997, 2004.

[2] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting Text in Natural Scenes with Stroke Width Transform," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pp. 2963–2970, 2010.

[3] J. Zhang and R. Kasturi, "Character energy and link energy-based text extraction in scene images," *Computer Vision–ACCV 2010*, pp. 308–320, 2011.

[4] X. Chen and A. Yuille, "Detecting and Reading Text in Natural Scene," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pp. 366–373, 2004.

[5] Y. Pan, X. Hou, and C. Liu, "Text localization in natural scene images based on conditional random field," in *2009 10th International Conference on Document Analysis and Recognition*, pp. 6–10, 2009.

[6] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Computer Vision–ECCV 2006*, pp. 430–443, 2006.

[7] http://svr-www.eng.cam.ac.uk/~er258/work/fast.html.

[8] J. Canny, "A Computational Approach To Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–714, 1986.

[9] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[10] P. Shivakumara, T. Q. Phan, and C. L. Tan, "A Laplacian Approach to Multi-Oriented Text Detection in Video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 412–419, 2011.