A Novel Text Detection System Based on Character and Link Energies

Jing Zhang and Rangachar Kasturi, *Fellow, IEEE*

***Abstract*— We propose a novel method by using three new character features to detect text objects comprising two or more isolated characters in images and videos. A new text model is constructed to describe text objects. Each character is a part in the model and every two neighboring characters are connected by a link. Two characters and the link connecting them are defined as a text unit. For every candidate part, we compute character energy based on our observation that each character stroke forms two edges with high similarities in length, curvature, and orientation. For every candidate link, we compute link energy based on the similarities in color, size, stroke width, and spacing between characters that are aligned along a particular direction. For every candidate text unit, we combine character and link energies to compute text unit energy which measures the likelihood that the candidate is a text object. We evaluated the performance of the proposed method on ICDAR 2003/2005 data set, Microsoft Street view data set, and video analysis and content exploitation video data set. The experimental results demonstrate that our method can capture the inherent properties of characters and discriminate text from other objects effectively.**

***Index Terms*— Text extraction, video indexing, image tags, content based information retrieval.**

I. Introduction

WITH increasing availability of low cost portable cameras and video recorders, the amount of images and videos captured and uploaded to the internet are growing at an explosive rate. This makes it increasingly difficult to locate specific images or videos of interest.

As a well-defined model of concepts for humans' communication, text provides much semantic information related to the content. If this text information can be extracted and harnessed efficiently, it can provide a much truer form of content–based access to images and videos. Hence, text detection from image and video is an important research topic in computer vision. However, this is a very challenging task due to the presence of various fonts, colors, sizes, orientations, complex backgrounds, varying illuminations, and distortions.

The goal of text detection is to localize text regions in an image/video-frame and generating tight bounding boxes around all text objects. A large number of text detection methods have been reported in the literature. Many supervised text detection approaches are based on Support Vector Machines (SVM), Boosting algorithms, Neural Networks, classifier fusion and combination, and so on [1]–[9]. However, the quality and quantity of training samples may affect their performance significantly. Many other published approaches use well known image features, such as edge, gradient, color, moments, spatial variance, Histogram of Oriented Gradient (HOG), Gabor filters, FFT, DCT, and Wavelet for text detection [10]–[15]. However, these are general purpose object features which do not capture characteristics specific to text objects. We propose a novel unsupervised method using a new pictorial structure-based text model and three new text-specific features to detect text objects. In the text model, each character is a part and two neighboring parts are connected by a link. Two parts and the link connecting them form a text unit. We present new character features and compute *character energy* and *link energy* for each part and link. By combining character and link energies, text unit energy is computed for each text unit to indicate the probability that a candidate text model is a real text object.

Our contributions are: (1) a new text model that can capture the characteristics of characters and the structure of text objects simultaneously; (2) three new character features to describe the inherent properties of characters. Our method is robust to the font, size, color, and orientation of text and can discriminate text objects from others effectively. However, our method works only when text is composed of two or more isolated characters which are placed in an orderly manner (typically along a line or curve). It is not designed to detect single characters since there are typically many such candidates that can only be accurately labeled as text after an optical character recognition (OCR). It is also not appropriate for languages in which many characters in a word are connected together or a single composite character is formed by disjoint character strokes.

The rest of the paper is organized into four parts. In Section II, we review the related work. In Section III, we present our text detection method. We evaluate and discuss the performance of our method in Section IV using three image and video datasets and present the results. Finally, we conclude the paper in Section V.

II. Related Work

Hundreds of papers have addressed the problem of text detection. Chen *et al.* [16], Jung *et al.* [17], and Zhang *et al.* [18] have presented comprehensive surveys of

Manuscript received May 28, 2012; revised April 26, 2013 and November 8, 2013; accepted July 1, 2014. Date of publication July 23, 2014; date of current version August 21, 2014. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mary Comer.

J. Zhang was with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620 USA. He is now with the Department of Radiology, Duke University, Durham, NC 27707 USA (e-mail: jingzhangusf@gmail.com).

R. Kasturi is with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620 USA (e-mail: r1k@cse.usf.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2014.2341935

text detection. According to the features used and the ways they work, text detection methods can be divided into two categories: region based and texture based.

Region based approach utilizes differences in descriptors between text and background regions to detect text objects. Color, edge, and connected component (CC) are often used in this approach. By noticing that there are transient colors between background and caption text, Kim *et al.* [19] first define a pixel as a transition pixel if the intensity changes of its two neighbors are larger than a threshold. Then, the small gaps between transition pixels are filled to generate CCs. The probability of caption text is computed based on the density of transition pixels and the number of different local binary pattern in the region. Finally, text regions are refined by vertical and horizontal projections and temporal redundancy in videos. Harris corners are used by Zhao *et al.* [20] to localize text in video documents. After finding the corner points by Harris detector, morphological dilation is used to merge nearby corners into one region. False positive regions are filtered out by spatial constraints. The text detection results are combined with motion vectors to detect moving text regions. Yi *et al.* [21] use stroke and color information to localize text regions. For stroke based partition, candidate text regions are extracted using stroke properties and refined using intensity magnitude and aspect ratio. For color based partition, the colors are clustered using a K-means method based on the histogram of non-edge pixels. Then adjacent character is grouped by geometric constraints and text line is grouped by Hough transform based on angles and lengths of line segments that connect neighboring characters. Shivakumara *et al* [22] detect text in video in frequency domain. After Fourier Laplacain filtering, their method uses K-means clustering to extract candidate text regions. Text string straightness and edge density based on skeletons are used to eliminate false positives.

Texture based approach uses distinct texture properties of text to separate text objects from background. Machine learning methods are often used in this approach. Pan *et al.* [15] localize text using conditional random field (CRF). First, the confidence map showing the probability of a region containing text is computed using a WaldBoost algorithm based on HOG and a boosted classifier calibration method. Then, CRF is used to analyze CCs obtained by Niblack's binarization. A CRF model is constructed by using 14 unary and binary features extracted from CCs and their neighbors. Text regions are labeled by minimizing the energy function of the graph. Finally, minimum spanning tree is used to group text components into text objects. Tu *et al.* [23] calculate the average intensity and statistics of the number of edges from training text samples. Adaboost is used to classify the candidate blocks. Text boundaries are matched with pre-generated deformable templates based on shape context and informative features. Celine *et al.* [24] divide an image into three clusters, textual foreground, background, and noise, based on Euclidean distance and Cosine similarity. The textual foreground is extracted by finding the largest regularity of CCs using a combination of spatial and frequency information.

The papers related to our work are [25]–[28]. Our previous work [25] uses HOG and graph spectrum to detect and group

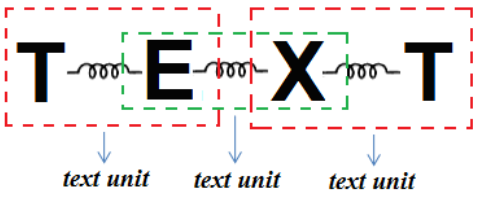


Fig. 1. Text model and text unit.

text regions. Compared with [25], the method proposed in this paper computes not only the length similarity of stroke edges, but the similarities of orientation, curvature, and stroke width. In [26], Wang *et al.* use pictorial structure model to compute word configuration after detecting characters by a HOG feature based supervised method. Compared with [26], our method first constructs a pictorial structure-based text model, then initializes and refines the parts and connections based on the proposed character features in an unsupervised way. Epshtein *et al.* [27] uses Stroke Width Transform (SWT) to detect text. Both [27] and our method use the gradient directions of edges to capture the properties of character. However, closed boundaries are not detected in [27], hence only the consistency of stroke width is used to extract candidate character regions, and corresponding points must be calculated twice in order to capture bright text on dark background and vice-versa. Our method can find corresponding points in one step using the closed boundary of character. Besides stroke width, the orientations and curvatures of stroke edges are used as well to detect character in our method. Therefore it is more robust to stroke width changes. Moreover, because each character is the region bounded by a closed boundary, our method can capture text properties effectively by computing the similarity between neighboring characters based on geometrical features. [28] is the preliminary version of this paper.

III. A Novel Algorithm for Text Detection

Inspired by the pictorial structure [29] that models an object using a collection of parts to indicate local properties of the object and spring-like links between parts to indicate deformable configuration of the object, we present a new *text model* by assuming that text objects always contain at least two characters. In this model, each character of a text object is a part and every two neighboring characters are connected by a spring-like link. We also define a *text unit* as two neighboring characters and the link connecting them. Hence, any text object is a combination of one or more text units. Fig. 1 shows a text model that is made up of three text units.

The proposed text detection method contains five steps as shown in the flowchart in Fig. 2. In this section, we describe each step in detail.

A. Initialization of Candidate Text Objects

The initialization of a candidate text object is based on two assumptions: (1) the boundary of a character is closed in the image because typically the character has relatively big contrast to its background; (2) text is made up of one

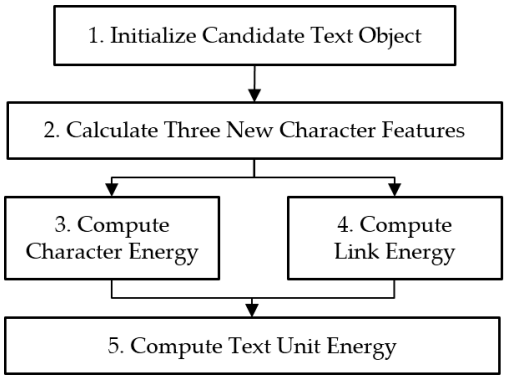


Fig. 2. Flowchart of the proposed method.

or more text units and each character should have at least one neighboring character.

First, we localize the candidate parts by extracting closed boundaries in the edge map generated by a zero-crossing based edge detector [30]. Due to variations in the edge strength along the text to background boundaries, using a single threshold to detect edge points forming closed boundaries is not robust. When the threshold is low, we obtain closed boundaries, but many noise fragments in non-text regions are also detected. When the threshold is high, we obtain fragmented edge sets which do not form closed boundaries, but the noise is significantly reduced. Hence we use a low threshold and a high threshold in our algorithm. We accept only those edges for which at least 75% of its pixels are present in both low and high thresholded images. After that, for each edge we compute its Euler number, which is defined as the number of edge (=1) minus the number of holes of this edge. The closed boundaries are obtained by extracting the closed parts of the edges whose Euler numbers are less than one (i.e., the edges containing at least one hole). The regions bounded by the closed boundaries are discarded if they fall outside the range of typical characteristics of characters (aspect ratio < 5, size $<$ Image-size/2, and number of holes > 4). Then, the remaining bounded regions are marked as candidate parts.

Second, we initialize candidate links by finding the neighbors of each candidate part based on the second assumption. Let v_i and v_j be two candidate parts with widths W_{v_i} and W_{v_j}, heights H_{v_i} and H_{v_i}, and centroids C_{v_i} and C_{v_i} . If

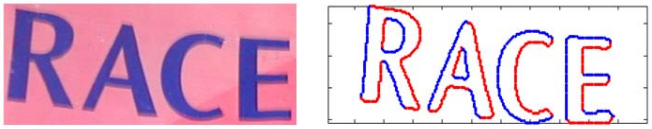
$$dist(C_{v_i}, C_{v_j}) \leq w_d \cdot \min(\max(W_{v_i}, H_{v_i}),\ \max(W_{v_j}, H_{v_j})) \tag{1}$$

we say v_j is a neighbor of v_i, where $dist(C_{v_i}C_{v_j})$ is the Euclidean distance between C_{v_i} and C_{v_i}. The weight w_d is empirically set to 2. Since we require at least two-character strings, any candidate part that has no neighbors is removed from further consideration. The candidate links are initialized by connecting every pair of candidate parts.

Finally, the candidate character parts that are reachable by one another via one or more links are grouped to form a candidate text object. Fig. 3 shows 6 and 19 candidate text objects initialized in two images from ICDAR 2003/2005 text localization dataset.

(a) (b)

Fig. 3. Initialization of candidate text objects. (a) Original image. (b) Candidate text objects.



(a) (b)

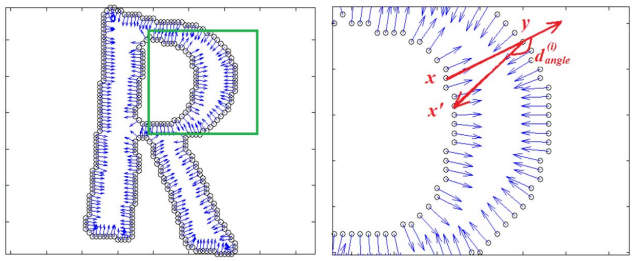
Fig. 4. Similarities of stroke edge pairs. (a) Original image. (b) The edge pairs of strokes.

B. Character Features

Because only edge and spatial information are used to initialize text objects, a candidate text may contain many non-character parts, even no character parts. Hence, we need to use the unique characteristics of character and text to refine initialized candidates.

One important characteristic that can discriminate text object from other objects is that characters are made up of strokes that typically have approximately uniform thickness resulting in two near parallel edge sets in their boundaries. The two edge sets have high similarities in length, orientation, and curvature. Fig. 4 illustrates this characteristic. The boundaries of four characters have been manually painted, for illustrative purposes only, into two edge sets shown in blue color and red color. We can see that the red and blue edges have similar lengths, orientations, and curvatures.

We capture the similarities of stroke edges using the gradient vector of each edge point on the boundary. In Fig. 5-a, the blue arrows show the gradient vectors of the character 'R' in Fig. 4. As illustrated in Fig. 5-b, which is the close-up of the green box in Fig. 5-a, given an arbitrary edge point x and its gradient vector V_x, we can find another edge point y along the direction of V_x. We define y as the *corresponding point* of x. Point set (x, y) is a *corresponding pair*. If x and y are two edge points of a character, the gradient direction of y should be approximately opposite to that of x due to the similar orientations and curvatures of stroke edge pairs.



(a) (b)

Fig. 5. Gradient vectors of edge points. (a) Gradient vectors of 'R' (arrows). (b) The closeup of the green box in (a).

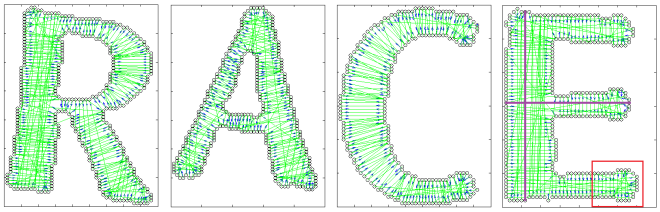


Fig. 6. Corresponding pairs and links (Purple lines and red box are explained later in the text).

Note that if the corresponding point of x is y, the corresponding point of y may or may not be x. For example, in Fig. 5-b, the corresponding point of x is y, but the corresponding point of y is x'. All corresponding points of four characters 'RACE' are connected by green lines in Fig. 6. We use the opposite directions of gradient vectors to find corresponding pairs when the gradient vectors point to the outside of a part, which means the background is darker than the character part.

Based on the definition of corresponding point and the similarities of stroke edge pairs, we know that:

1) For a character, it has two near parallel edge sets and the gradients of an edge point and its corresponding point should have approximately opposite directions; e.g., in Fig. 6, the directions of blue arrows of most corresponding pairs are approximately opposite.
2) For a character, the distances between the points and their corresponding points are similar because the change of the stroke width is usually small; e.g., in Fig. 6, most green lines that connect corresponding points have similar lengths.

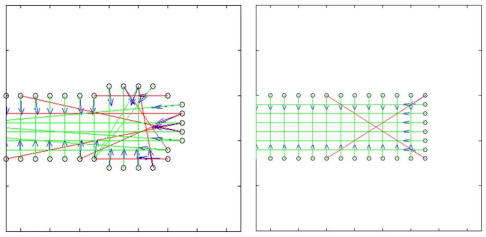
Consequently, we are able to propose three new character features using the two statements above.

C. Average Angle Difference of Corresponding Pairs (D_{angle})

Let N denote the number of edge points of a candidate part. $P^{(i)}(1 \le i \le N)$ is the i^{th} edge point with the corresponding point $P^{(i)}_{corr}$. $\theta^{(i)}_{P}$ and $\theta^{(i)}_{P_{corr}}$ are the gradient directions of $P^{(i)}$ and $P^{(i)}_{corr}$. The difference of the gradient directions of the corresponding pair $(P^{(i)}, P^{(i)}_{corr})$ is defined as:

$$d^{(i)}_{angle} = abs(\theta^{(i)}_{p} - \theta^{(i)}_{p_{corr}}) \tag{2}$$

where function abs() computes absolute values. The range of $d^{(i)}_{angle}$ is $[0\ \pi]$. An illustration of this is shown in Fig. 5-b.



(a) (b)

Fig. 7. Noise connections (red) and non-noise connections (green). (a) The stroke in the red box in Fig. 6. (b) A noise-free stroke.

TABLE I

D_{angle} AND $F_{non\text{-}noise}$ OF 'R,A,C,E'

	R	A	C	E
D_{angle}	0.889	0.865	0.925	0.897
$F_{non\text{-}noise}$	0.754	0.684	0.897	0.806

Accordingly, D_{angle} is defined as:

$$D_{angle} = \frac{1}{N \cdot \pi} \sum_{i=1}^{N} d^{(i)}_{angle} \tag{3}$$

D_{angle} measures the average gradient direction difference of all corresponding pairs of a candidate part. For an ideal character whose every corresponding pair has exactly opposite gradient directions, D_{angle} reaches the maximum value 1.

D. Fraction of Non-Noise Pairs ($F_{non\text{-}noise}$)

In some cases, however, a character may have a smaller D_{angle} due to noise or deformations. We compute $F_{non-noise}$ to measure the noise and deformation levels of a part based on $d^{(i)}_{angle}$. Assume a candidate character part has N corresponding pairs and β is a pre-defined angle, $F_{non\text{-}noise}$ is defined as follows:

$$F_{non\text{-}noise} = \frac{1}{N} \sum_{i=1}^{N} h(d^{(i)}_{angle}, \beta) \tag{4}$$

where

$$h(d^{(i)}_{angle}, \beta) = \begin{cases} 1 & if\ d^{(i)}_{angle} > \beta \\ 0 & else \end{cases} \tag{5}$$

$F_{non\text{-}noise}$ is the fraction of all pairs for which the angle difference $d^{(i)}_{angle}$ is greater than β (150^{o} in our experiments). If the angle difference $d^{(i)}_{angle}$ of the i^{th} pair is smaller than β, this pair is a *noise pair* and its connection is a *noise connection*. Otherwise, it is a *non-noise pair* and its connection is a *non-noise connection*. Fig. 7-a is the close-up of the red box in Fig. 6 showing non-noise and noise connections as green and red lines, respectively. Compared with a noise-free stroke shown in Fig. 7-b, it has much more noise connections.

Table I lists the D_{angle} and $F_{non\text{-}noise}$ features of 'R,A,C,E' shown in Fig. 4-a.

D_{angle} and $F_{non\text{-}noise}$ features of all candidate parts in Fig. 3 are illustrated as color-coded intensity images using "jet" colormap in Matlab in Fig. 8. Note the color scale to

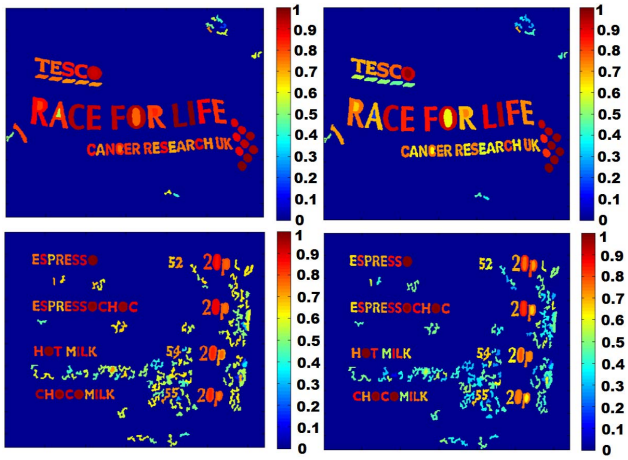


Fig. 8. (a) D_{angle} and (b) $F_{non-noise}$ of two example images.

the right of each figure. Shades of red with values closer to 1 represent higher confidence in characters. Clearly, most character parts have much bigger D_{angle} and $F_{non\text{-}noise}$ than non-character parts.

However, we notice that the holes of the characters 'R' and 'O' and the cluster of nine circles in the two top images of Fig. 8 also have big D_{angle} and $F_{non\text{-}noise}$ values. It is because that every edge point of a circle has a corresponding point that has an exactly opposite gradient direction. Hence, $D_{angle} = 1$ and $F_{non\text{-}noise} = 1$ for a perfect circle. Circle-like objects have high D_{angle} and $F_{non\text{-}noise}$ values for the same reason. To solve this problem, we divide the non-noise connections into two types: *stroke-length connection* and *stroke-width connection*. By doing so, we can separate circle-like objects and compute the feature *vector of stroke width*.

In Fig. 6, there are some connections whose lengths are the stroke lengths of the character with many intersections with other connections. For instance, the lengths of the two thick purple lines in Fig. 6 are the lengths of the vertical and horizontal strokes of 'E' (there are many such connections but we have colored only two for illustration). Both of them have many intersections with other connections.

Let $k^{(i)}(1 \leq i \leq N)$ be one of N non-noise connections of a part and have $I_k^{(i)}$ intersections with other non-noise connections. We define stroke-length connection and stroke-width connection as follows:

$$k^{(i)} \in \begin{cases} stroke-length\ connection & if\ \frac{I_k^{(i)}}{N} > T_{IS}; \\ stroke-width\ connection & otherwise. \end{cases} \tag{6}$$

T_{IS} is a confidence threshold and was set to 0.2 in our experiments. Hence, if a connection intersects with more than 20% of all connections, we consider it as a stroke-length connection.

For a circle, every connection intersects with all other connections at its center. Hence, all non-noise connections of a circle are stroke-length connections. Similarly, circle-like objects have much more stroke-length connections than stroke-width connections. The first column of Fig. 9 shows two circle-like non-characters in the green boxes and four characters.

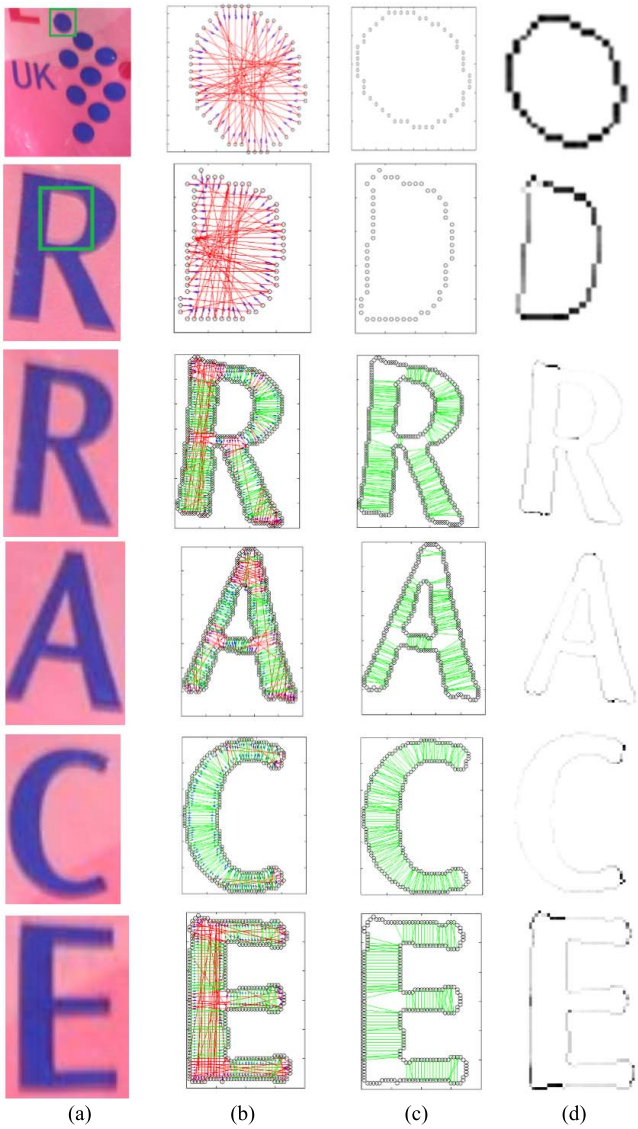


Fig. 9. Stroke-width connections of six parts. (a) Six original parts; (b) The connections of six parts. Noise and stroke-length connections are shown in red and stroke-width connections are shown in green; (c) Stroke-width connections; (d) The edges of six parts. The intensity of each edge point represents the number of intersections of the line that originate at this point with other lines.

The second column shows their connections. Noise and stroke-length connections are in red and stroke-width connections are in green. The third column shows only the stroke-width connections. Note that, for two circle-like objects, no connections are left after removing the noise and stroke-length connections. The last column shows the edges of six parts. The intensity of each edge point represents the number of intersections of the line that originate at this point with other lines. Darker color corresponds to more intersections. Clearly, the characters have much more stroke-width connections than the non-characters. This observation points to a simple procedure to eliminate non-characters.

We compute the number of stroke-width connections as a fraction of all connections for each candidate part. If this fraction is less than 0.5, we label the part as a non-character part.

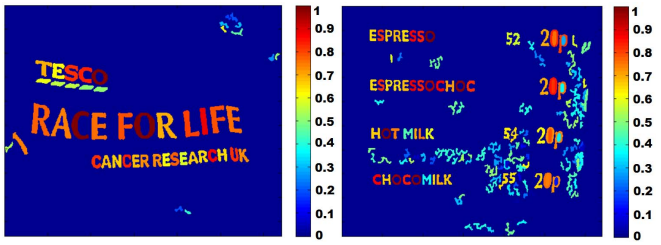


Fig. 10. Percentages of stroke-width links of two example images.

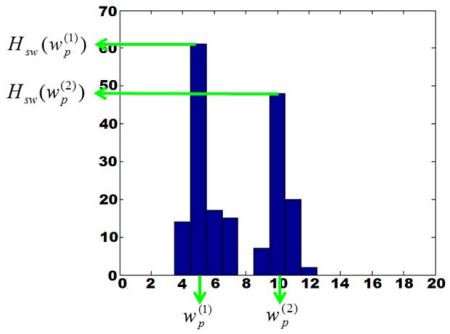


Fig. 11. Peaks of H_{sw} of the character 'E'.

Fig. 10 shows this fraction for each part in our examples. We can see that many circle-like non-character parts have a shade of blue and are eliminated.

E. Vector of Stroke Width (V_{width})

Characters typically have one or two dominating stroke widths depending on their fonts. In our algorithm, we estimate two dominating stroke widths for each candidate part using stroke-width connections.

Let H_{sw} be the histogram of the lengths of stroke-width connections (Euclidean distance measured in pixel width units rounded to nearest integer value) and $H_{sw}(j)$ be the value of H_{sw} at the j^{th} bin. We estimate the dominating stroke width values $W_d^{(i)}$ $(i \in [1,2])$ of a part as follows: we first locate $W_p^{(1)}$ and $W_p^{(2)}$ that are the bins in ascending order where H_{sw} reaches two highest peaks as shown in Fig. 11 (H_{sw} of the character 'E'). Then, we estimate dominating stroke width $W_d^{(i)}$ $(i \in [1,2])$ through a weighted average computation using $W_p^{(i)}$ $(i \in [1,2])$ and its two immediately adjacent neighbors:

$$W_d^{(i)} = \frac{r_1 \times (W_p^{(i)} - 1) + W_p^{(i)} + r_2 \times (W_p^{(i)} + 1)}{r_1 + 1 + r_2} \tag{7}$$

where two weights $r_1 = H_{SW}(W_P^{(i)} - 1)/H_{SW}(W_P^{(i)})$ and $r_2 = H_{SW}(W_P^{(i)} + 1)/H_{SW}(W_P^{(i)})$. If H_{sw} has only one peak, let $W_p^{(1)} = W_p^{(2)}$.

If the sum of $H_{SW}(W_P^{(i)} - 1)$, $H_{SW}(W_P^{(i)})$, and $H_{SW}(W_P^{(i)} + 1)$ is less than a *Confidence Threshold* T_c, equal to 20% of the total number of stroke-width connections for the part, we discard $W_d^{(i)}$ as a false peak. If there is only one dominating stroke width, we set $W_d^{(1)} = W_d^{(2)}$. If there are no dominating stroke widths, we set $W_d^{(1)} = W_d^{(2)} = 0$. The vector of stroke width V_{width} is defined as:

$$V_{width} = [W_d^{(1)},\ W_d^{(2)}] \tag{8}$$

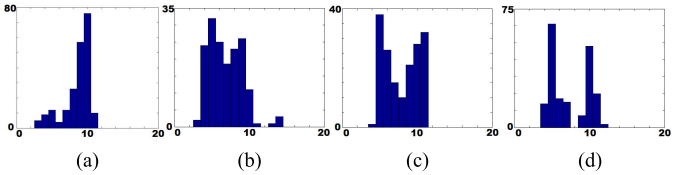


Fig. 12. Histograms of the lengths of stroke-width connections. (a) 'R'. (b) 'A'. (c) 'C'. (d) 'E'.

TABLE II
V_{width} OF 'R,A,C,E'

	R	A	C	E
$W_d^{(1)}$	4.76	5.01	5.38	5.03
$W_d^{(2)}$	9.68	8.80	9.53	10.17

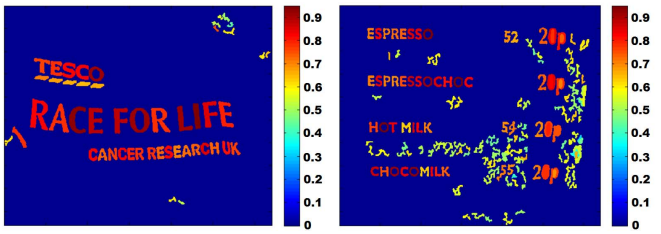


Fig. 13. Character energy (E_{Char}) of two examples images.

Fig. 12 is the histograms of the lengths of stroke-width connections (green lines shown in the third column of Fig. 9) of 'R,A,C,E'. The values of V_{width} of 'RACE' are listed in Table II. We can see that all four characters have similar dominating stroke widths.

Now we have computed three new character features. Next we will use D_{angle} and $F_{non-noise}$ to compute character energy and V_{width} to compute link energy.

F. Character Energy

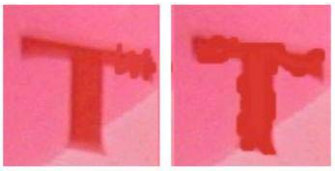
For a part v_i, we consider that its $D_{angle}^{(i)}$ and $F_{non-noise}^{(i)}$ are equally important for text detection and define the *character energy* ($E_{Char}^{(i)}$) of v_i as follows:

$$E_{Char}^{(i)} = \frac{D_{angle}^{(i)} + F_{non\text{-}noise}^{(i)}}{2} \tag{9}$$

Note that $0 \le E_{Char}^{(i)} \le 1$. It can be treated as a measure of the probability that v_i is a character. Fig. 13 shows E_{Char} of two example images.

It is clear that the characters have larger E_{Char} than non-characters. Therefore, E_{Char} can discriminate text objects from other objects and is robust to the changes in font, size, color, and orientation of characters.

Please note that D_{angle} and $F_{non\text{-}noise}$ are correlated because d_{angle} is used in computing both D_{angle} and $F_{non\text{-}noise}$. However, they describe different characteristics of



(a) (b)

Fig. 14. Two characters with different noise/deformation levels. (a) A character with high noise/deformation only at one end of its horizontal stroke. (b) A character with moderate noise/deformation.

a character and complement each other for text detection. We illustrate an example in Fig. 14 using two 'T's with different noise/deformation levels. For (a), only one end of its horizontal stroke has high noise/deformation and its $D_{angle} = 0.8846$, $F_{non\text{-}noise} = 0.5950$, and $E_{Char} = 0.7398$. For (b), its entire boundary has moderate noise/deformation and its $D_{angle} = 0.8847$, $F_{non\text{-}noise} = 0.5261$, and $E_{Char} = 0.7054$. We can see that D_{angle} of two 'T's are almost the same, but the $F_{non\text{-}noise}$ of (a) is about 7% higher than that of (b). That means D_{angle} alone may not distinguish two 'T's. It is necessary to use E_{char}, which combines D_{angle} and $F_{non\text{-}noise}$, to capture more information from characters. In this case, although the stroke with big noise/deformation makes (a) has slight lower D_{angle}, it still has higher character energy than (b) because most parts of its edges are well paralleled to each other.

G. Link Energy

According to our assumption, a text object contains more than one character. Therefore, the relationships between two neighboring characters can also provide important information for text detection.

In this subsection we compute *link energy* for every candidate link to measure the probability that two parts connected by the link are both characters. Link energy is computed by measuring two values: (1) Similarity in the properties of neighboring parts, such as the color, stroke width, and size. (2) Spatial consistency in the direction and distance between neighboring parts in a string of parts.

For two connected parts v_i and v_j, we use color, stroke width (V_{width}), character width, and character height to capture the similarities between them. Table III shows the computations, where $Simi(R) = min(R, 1/R)$, C_i and C_j are the means of RGB channels of v_i and v_j. $V_i(k)$, W_i, H_i and $V_j(k)$, W_j, H_j are the the k^{th} (k$\in$ [1, 2]) stroke width, character width, and character height of v_i and v_j, respectively. Thus, if two neighboring parts have similar colors, stroke widths, widths, and heights their joint similarity measures will be high (close to 1). The link energy between v_i and v_j is defined as:

$$E_{Link}^{(i,j)} = \frac{1}{4}\sum_{k=1}^{4} w_k \cdot S_{i,j}^{(k)} \tag{10}$$

where w_k are non-negative weights summing to 1. We set w_k to 0.25 in our experiments to give equal weight to every similarity. Note that w_1 can be set to a small value or 0 to detect the words whose characters are in various colors. The higher $E_{Link}^{(i,j)} (0 \le E_{Link}^{(i,j)} \le 1)$ means v_i and v_j have higher similarities.

TABLE III
THE SIMILARITY COMPUTATION OF TWO CHARACTERS

Color	$S_{i,j}^{(1)} = \frac{1}{3}\cdot\sum_{C=\{R,G,B\}}(1-\frac{\lvert C_i - C_j\rvert}{255})$
V_{width}	$S_{i,j}^{(2)} = \frac{1}{2}\cdot\sum_{k=1}^{2} Simi(R_{i,j}^{V}(k)),\quad R_{i,j}^{V}(k) = \frac{V_i(k)}{V_j(k)}$
Character Width	$S_{i,j}^{(3)} = Simi(R_{i,j}^{W}),\quad R_{i,j}^{W} = \frac{W_i}{W_j}$
Character Height	$S_{i,j}^{(4)} = Simi(R_{i,j}^{H}),\quad R_{i,j}^{H} = \frac{H_i}{H_j}$

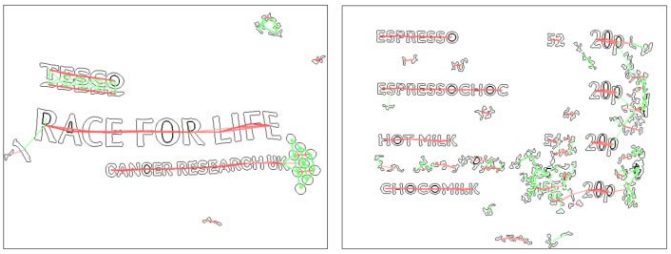


Fig. 15. Link energy (E_{Link}) of two example images.

To capture the spatial relationship between two connected parts v_i and v_j, we use the length of the link to depict the distance interval between v_i and v_j and the direction of the link to depict the alignment of v_i and v_j.

We can visualize the link energies of candidate links as follows:

1) The intensity of a link indicates the value of the link energy. Higher intensity means larger similarity of the two parts connected by this link;
2) The length of a link indicates the spatial distance between two linked parts;
3) The color of a link indicates the alignment of the parts. For any two links that are joined at the same part, if their direction difference is smaller than a pre-defined angle θ, the two links are shown in the same color. Otherwise, the two links are shown in different colors.

The characters of a text object typically have similar properties and are aligned along certain direction with similar intervals. Hence, the candidate parts that are connected by the links with high intensities, similar lengths, and in the same color have a high probability to be a text object. Fig. 15 shows E_{Link} of two example images ($\theta = 20°$). We can see that all character parts are connected by the links with high intensities and similar lengths in the same color. However, some aligned non-character parts that have similar appearances are also connected by the links with high intensities. In the next subsection, we will remove these non-character parts using text unit energies, which are computed by combining character and link energies.

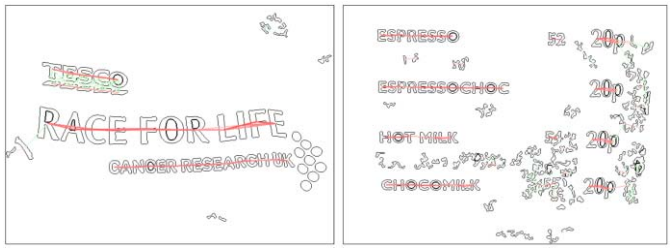
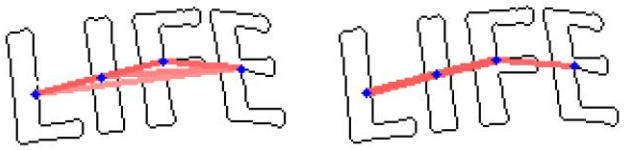


Fig. 16. Text energy (E_{Text}) of two example images.



(a) (b)

Fig. 17. Maximum spanning tree of an initialized text object. (a) Links before MST. (b) Links after MST.

H. Text Unit Energy

For the text unit containing two parts v_i and v_j, the text unit energy $E_{Text}^{(i,j)}$ is computed using character energies $E_{Char}^{(i)}$ and $E_{Char}^{(j)}$ and link energy $E_{Link}^{(i,j)}$:

$$E_{Text}^{(i,j)} = \frac{1}{2}[(\frac{E_{Char}^{(i)} + E_{Char}^{(j)}}{2}) + E_{Link}^{(i,j)}] \tag{11}$$

Fig. 16 shows the text unit energies of all text units in two example images using the color scale. The higher intensity of a link indicates the higher text unit energy. Compared with Fig. 16, the intensities of the links that connect non-character parts have become much smaller, while the intensities of the links that connect character parts are still high.

Based on our definition of text model, a character part at most has two links that connect it to its previous and next character parts. However, when some parts are close to each other, the parts may have more links due to multiple neighbors found by (1). For example, every part in Fig. 17-a is fully connected with others due to their proximity. In order to obtain the same structure as the text model, *Maximum Spanning Tree* (MST) is used to remove the redundant connections. We use Prim's algorithm [31], text unit energy, and the distance between two parts to find MST. Fig. 17-b is the MST of Fig. 17-a. The blue stars are the centroids of the parts and the links are shown in red color. After MST, the output text has the same structure as the text model defined in Fig. 1.

To refine the detected text objects, text units whose text unit energies are smaller than a pre-defined threshold T_{Text} are removed from the text objects. While the choice of this threshold depends upon the characteristics of the datasets, a threshold of 0.7 worked well for several standard test datasets we used for testing our algorithm. The thresholded results and final detection outputs of two example images are shown in Fig. 18. The detected characters and text objects are bounded by the green boxes and yellow boxes, respectively.

Fig. 18. Thresholded E_{Text} and text detection outputs.

IV. Experiments

We used three publicly available datasets to evaluate the performance of our method: ICDAR 2003/2005 text locating dataset [32], [33], Microsoft street view text detection dataset [34], and Video Analysis and Content Exploitation (VACE) dataset [35]. The three datasets contain a total of 558 images and 50 videos.

We compared our algorithm's performance with that of other text detection methods. For ICDAR 2003/2005 text locating dataset and Microsoft street view text detection dataset, *precision*, *recall,* and standard f defined in [36] were used as evaluation measures. For VACE video dataset, three measures, *Multiple Object Detection Precision* (*MODP*), *Multiple Object Detection Accuracy* (*MODA*), and *Sequence Frame Detection Accuracy* (*SFDA*), presented in [35] were used to evaluate the proposed method. The detailed definitions and explanations of the measures can be found in [35].

The proposed algorithm was implemented in Matlab 7.8.0 (2009a) and run on a PC with Intel Core 2 Quad CPU at 2.66GHz and 4 GB memory under Microsoft Windows 7. The execution time for an image was 15 to 20 seconds depending on the size of the image and the number of closed boundaries extracted.

A. Results on IDCAR 2003/2005 Dataset Objects

ICDAR 2003/2005 text locating competition dataset is the most widely used benchmark for scene text detection. The dataset contains 258 training and 251 test images with various sizes from 307×93 to 1280×960. Fig. 19 shows the detected text objects and the corresponding detection results on selected images in this dataset.

We can see that our method can detect most text objects in various conditions successfully, including different colors (Fig. 19-a), different lightings (Fig. 19-b), complex background (Fig. 19-c), different sizes (Fig. 19-d), different stroke widths (Fig. 19-e), flexible surface (Fig. 19-f), other similar symbols (Fig. 19-g), non-horizontal direction (Fig. 19-h and Fig. 19-i), and low contrast (Fig 19-j, Fig. 19-k, and Fig. 19-l).

Fig. 19. Examples of detection results on ICDAR 2003/2005 dataset. (a)–(l) show the images with text objects in various conditions.

TABLE IV
EVALUATION RESULTS ON ICDAR 2003/2005 DATASET

Algorithm	Precision	Recall	f
Ashida [36]	0.55	0.46	0.45
Hinneck Becker [37]	0.62	0.67	0.62
Zhang [26]	0.67	0.46	0.55
SWT [28]	0.73	0.60	0.66
Our Method	*0.74*	*0.62*	*0.67*

The evaluation results of our method on ICDAR 2003/2005 dataset is listed in Table IV along with the best algorithms reported in [36] and [37] and the algorithms proposed in [26] and [28]. We can see that the proposed text detection method achieved better performance than other listed algorithms.

B. Results on Microsoft Street View Dataset

This dataset consists of 307 natural images with sizes ranging from 1024×1360 to 1024×768 for text detection [28]. Because most of the images are street views with the presence of vegetations that may cause many closed boundaries, windows that create repeating patterns with similar properties, and text objects that are at low resolutions, this database is much harder than ICDAR 2003/2005 dataset. Fig. 20 illustrates some text detection results on this dataset. Compared with detection outputs of ICDAR 2003/2005 dataset, the detection results here have more false positives and more text objects are missed.

The evaluation results of SWT method [28] and our method are listed in Table V. The precision and recall of our method are slightly lower than those of SWT method. The two main reasons are: (1) the boundaries of some text objects are not closed due to the small sizes or low resolutions; (2) instead of detecting text objects containing two or more characters as our method does, SWT method only detects text objects with three or more characters. In this way, SWT method can remove many two-part false positives and miss only a few two-character text objects, since there are much more two-part false positives than two-character text objects in the dataset.

C. Results on VACE Dataset

VACE dataset has 50 broadcast news videos from CNN and ABC. The entire source videos were presented in a consistent format: MPEG–2 standard, progressive scanned at 720×480 resolution, GOP (Group of Pictures) of 12 and frame-rate at 29.97 fps (frames per second). In the experiments, 4500 successive frames from each video were used as the input and the text detection algorithm was applied on Intra-coded frames, which are self-contained with no motion compensation and the

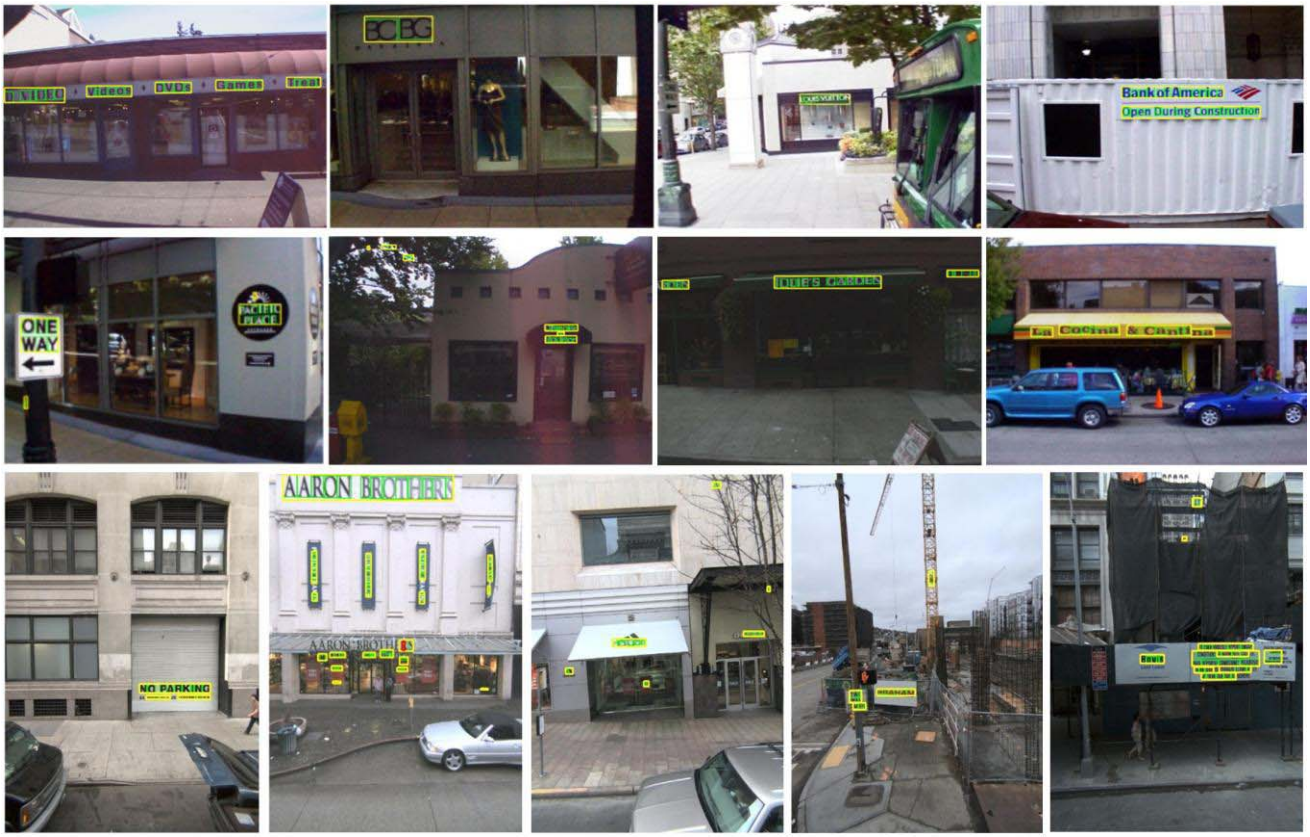


Fig. 20. Examples of text detection results on Street view dataset.

TABLE V
EVALUATION RESULTS ON MICROSOFT STREET VIEW DATASET

Algorithm	Precision	Recall	f
SWT method [28]	0.54	0.42	0.47
Our Method	*0.52*	*0.38*	*0.44*

lowest compression ratios. Beside the scene text objects, this dataset has a large number of caption text objects, which are artificially overlaid on the video frames at the time of editing.

Text detection from video frames presents some challenges over detection from images due to lossy video compression, noise, interlacing, color bleeding, loss of contrast, and blocking artifacts. Fig. 21 shows the examples of text detection results on VACE dataset, including the detection of scene text objects and caption objects. The evaluation results are listed in Table VI.

D. Discussion

We can see that the proposed method can detect the text with various fonts, sizes, colors, and orientations. However, there are still some missed text objects and false positives. In this subsection, we discuss the limitations of our method and potential solutions.

1) Repeating patterns: Besides text objects, some repeating patterns with high similarities of edge pairs may have high character and link energies as well. An example is shown in Fig. 22-a. The seven vertical black bars are marked as a text object incorrectly, because they have high similarities of stroke edges and almost the same width, length, color, and stroke width. This limitation may be solved using a shape filter [38] to remove repeated identical objects.
2) Single characters and text objects with connected characters: This limitation is caused by our assumption that any text object has at least two characters. Single characters are removed as false positives because they have no neighboring characters. Similarly, for text objects with connected characters, they are regarded as single characters because they have only one closed boundary. Fig. 22-b and 22-c illustrate two examples. Supervised methods can detect single characters using the information learned from training data. However, for text objects with connected characters, even supervised methods can hardly detect them.
3) Small characters: Because the proposed method initializes the candidate character parts based on the boundaries, when the sizes of text objects are too small, the detected boundaries cannot reflect the shapes of characters and the proposed character features may fail to capture text objects. Consequently, some small characters may have low character energies and be eliminated as false positives. An example of small characters and their boundary is shown in Fig. 22-d. We can see it is hard to recognize "UPS" if only based on the edges.

Fig. 21. Examples of text detection results on VACE dataset.

TABLE VI
EVALUATION RESULTS ON VACE DATASET

Algorithm	MODP	MODA	SFDA
VACE baseline [35]	0.418	0.301	0.527
Our Method	*0.495*	*0.383*	*0.589*

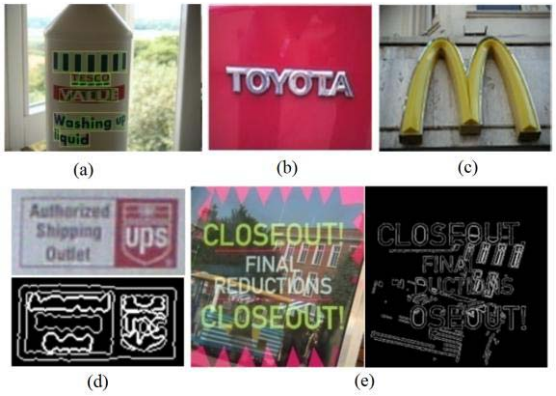


Fig. 22. Limitations of the proposed method. (a) Repeating patterns. (b) Connected characters. (c) Single character. (d) Small characters. (e) Transparent characters.

4) Transparent characters: If a character is transparent, its edge may connect with background edges and are not closed. In this case, transparent characters are removed due to unreliable edges. Fig. 22-e illustrates an example of transparent text object. The characters "ION" of "REDUCTIONS" are mixed with the background edges, and "C" of "CLOSEOUT" is removed due to its unclosed boundary. More color information is needed to solve this limitation.
5) Language dependency: We tested our method only on English language datasets. The characters of many eastern languages, such as Japanese, Korea, and Chinese, are composed of unconnected strokes. Therefore, we have to first group strokes to form a character, then group characters to form a text object. The proposed method can detect the strokes of each character, but the spatial relationship among strokes may not satisfy the constraints of link energy. A modified approach is necessary to address this problem.
6) Empirical parameters: We selected a number of parameters and thresholds empirically using three datasets widely used by the community. Since these test sets include a collection of images with wide variation in their text and background characteristics, we have demonstrated that our algorithm is robust and not very sensitive to these empirical parameters. Since an image or video frame typically contains a number of artifacts with characteristics similar to those of text, any text detection algorithm such as ours cannot be expected to be perfect in separating them from true text. Through our evaluations on common datasets we have shown that these parameters satisfactorily detect text objects in a variety of images and video. Since text detection is a preprocessing stage in a system with an ultimate goal of automated tagging and indexing of images and video, we believe that our algorithm serves its design purpose.

V. CONCLUSIONS

This paper presented a new unsupervised method to detect scene text objects by modeling each text object as a pictorial structure. A text object is described as a collection of characters with spring-like connections between two neighboring characters. For each character, it is observed that the edges of a stroke can be considered as a combination of two edge sets that have high similarities in length, orientation, and curvature. Taking advantage of this observation, we proposed three new character features, Average Angle Difference of Corresponding Pairs, Fraction of Non-noise Corresponding Pairs, and Vector of Stroke Width, to capture the similarities of stroke edge pairs and compute character energy for each candidate part. Since the characters of a text object typically have similar properties and are aligned along a particular direction, we compute link energy to describe the spatial relationship and property similarity between two neighboring characters. Text unit energy, which is a combination of character energy and link energy, is used to measure the probability that a candidate

text model is a text object and generate final detection result. We evaluated the proposed method using three datasets that contain 588 scene images and 50 videos and demonstrated good text detection capabilities. While there is always room for further improvement, we believe that we have presented an algorithm that meets the primary objective of our goal, that of identifying bounding boxes that contain predominantly text objects in images and videos.

REFERENCES

[1] C. S. Shin, K. I. Kim, M. H. Park, and H. J. Kim, "Support vector machine-based text detection in digital video," in *Proc. IEEE Signal Process. Soc. Workshop Neural Netw. Signal Process.*, vol. 2. Dec. 2000, pp. 634–641.

[2] K. I. Kim, K. Jung, and J. H. Kim, "Texture-based approach for text detection in images using support vector machine and continuously adaptive mean shift algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1638, Dec. 2003.

[3] Q. Ye, Q. Huang, W. Gao, and D. Zhao, "Fast and robust text detection in images and video frames," *Image Vis. Comput.*, vol. 23, no. 6, pp. 565–576, 2005.

[4] D. Chen, H. Bourlard, and J. Thiran, "Text identification in complex background using SVM," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2. 2000, pp. II-621–II-626.

[5] S. M. Hanif and L. Prevost, "Text detection and localization in complex scene images using constrained AdaBoost algorithm," in *Proc. 10th Int. Conf. Document Anal. Recognit.*, Jul. 2009, pp. 1–5.

[6] L. Tang and J. R. Kender, "A unified text extraction method for instructional videos," in *Proc. IEEE Int. Conf. Image Process.*, vol. 3. Sep. 2005, pp. 11–14.

[7] D. Chen, J.-M. Odobez, and H. Bourlard, "Text detection and recognition in images and video frames," *Pattern Recognit.*, vol. 37, no. 3, pp. 595–608, 2004.

[8] J. Gllavata, E. Qeli, and B. Freisleben, "Detecting text in videos using fuzzy clustering ensembles," in *Proc. 8th IEEE Int. Symp. Multimedia*, Dec. 2006, pp. 283–290.

[9] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2. Jun./Jul. 2004, pp. II-366–II-373.

[10] M. R. Lyu, J. Song, and M. Cai, "A comprehensive method for multilingual video text detection, localization, and extraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 243–255, Feb. 2005.

[11] Y. Liu, H. Lu, X. Xue, and Y.-P. Tan, "Effective video text detection using line features," in *Proc. 8th Control, Autom., Robot., Vis. Conf.*, Dec. 2004, pp. 1528–1532.

[12] P. Dubey, "Edge based text detection for multi-purpose application," in *Proc. 8th Int. Conf. Signal Process.*, vol. 4. 2006.

[13] J. Zhou, L. Xu, B. Xiao, R. Dai, and S. Si, "A robust system for text extraction in video," in *Proc. Int. Conf. Mach. Vis.*, Dec. 2007, pp. 119–124.

[14] H. Tran, A. lux, T. H. L. Nguyen, and A. Boucher, "A novel approach for text detection in images using structural features," in *Proc. 3rd Int. Conf. Adv. Pattern Recognit.*, 2005, pp. 627–635.

[15] Y.-F. Pan, X. Huo, and C.-L. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 800–813, Mar. 2010.

[16] D. Chen, J. Luettin, and K. Shearer, "A survey of text detection and recognition in images and videos," Institute Dalle Molled Intelligence Perceptive (IDIAP), Martigny, Switzerland, Res. Rep. IDIAP-RR 00-38, 2000.

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

[18] J. Zhang and R. Kasturi, "Extraction of text objects in video documents: Recent progress," in *Proc. 8th IAPR Int. Workshop Document Anal. Syst.*, Sep. 2008, pp. 5–17.

[19] W. Kim and C. Kim, "A new approach for overlay text detection and extraction from complex video scene," *IEEE Trans. Image Process.*, vol. 18, no. 2, pp. 401–411, Feb. 2009.

[20] X. Zhao, K.-H. Lin, Y. Fu, Y. Hu, Y. Liu, and T. S. Huang, "Text from corners: A novel approach to detect text and caption in videos," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 790–799, Mar. 2011.

[21] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2594–2605, Sep. 2011.

[22] P. Shivakumara, T. Q. Phan, and C. L. Tan, "A Laplacian approach to multi-oriented text detection in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 412–419, Feb. 2011.

[23] Z. Tu, X. Chen, A. L. Yuille, and S.-C. Zhu, "Image parsing: Unifying segmentation, detection, and recognition," *Int. J. Comput. Vis.*, vol. 63, no. 2, pp. 113–140, 2005.

[24] C. Mancas-thillou and B. Gosselin, "Color text extraction with selective metric-based clustering," *Comput. Vis. Image Understand.*, vol. 107, nos. 1–2, pp. 97–107, 2007.

[25] J. Zhang and R. Kasturi, "Text detection using edge gradient and graph spectrum," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 3979–3982.

[26] K. Wang and S. Belongie, "Word spotting in the wild," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 591–604.

[27] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2963–2970.

[28] J. Zhang and R. Kasturi, "Character energy and link energy-based text extraction in scene images," in *Proc. Asian Conf. Comput. Vis.*, 2011, pp. 308–320.

[29] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *Int. J. Comput. Vis.*, vol. 61, no. 1, pp. 55–79, 2005.

[30] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Roy. Soc. London B*, vol. 207, no. 1167, pp. 187–217, 1980.

[31] R. C. Prim, "Shortest connection networks and some generalizations," *Bell Syst. Tech. J.*, vol. 36, no. 6, pp. 1389–1401, 1957.

[32] *Datasets—Algoval—University of Essex*. [Online]. Available: http://algoval.essex.ac.uk/icdar/Datasets.html, accessed Jul. 16, 2014.

[33] *ICDAR 2005 Competitions: Text Locating—Algoval*. [Online]. Available: http://algoval.essex.ac.uk:8080/icdar2005/index.jsp?page=textlocate.html, accessed Jul. 16, 2014.

[34] *Microsoft Text Detection Database*. [Online]. Available: http://research.microsoft.com/en-us/um/people/eyalofek/text_detection_database.zip, accessed Jul. 16, 2014.

[35] R. Kasturi *et al.*, "Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 319–336, Feb. 2009.

[36] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, vol. 2. 2003, p. 682.

[37] S. M. Lucas, "ICDAR 2005 text locating competition results," in *Proc. 8th Int. Conf. Document Anal. Recognit.*, Sep. 2005, pp. 80–84.

[38] Z. Liu and S. Sarkar, "Robust outdoor text detection using text intensity and shape features," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

Jing Zhang received the Ph.D. degree from the University of South Florida, Tampa, FL, USA, in 2012. He is currently a Post-Doctoral Associate with Carl E. Ravin Advanced Imaging Laboratories, Duke University, Durham, NC, USA. His research interests are in medical image analysis, computer vision, and pattern recognition.

Rangachar Kasturi (M'82–SM'88–F'96) has been the Douglas W. Hood Professor of Computer Science and Engineering with the University of South Florida, Tampa, FL, USA, since 2003. He was a Professor with Pennsylvania State University, State College, PA, USA, from 1982 to 2003. He received the Ph.D. degree from Texas Tech University, Lubbock, TX, USA, in 1982. His research interests are in computer vision, pattern recognition, and document image analysis. He has authored the textbook, *Machine Vision* (McGraw-Hill, 1995). He served as the Editor-in-Chief of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (1995–98), a Fulbright Scholar (1999), the President of the International Association for Pattern Recognition (IAPR) (2002–2004), and the President of the IEEE Computer Society (2008). He is a fellow of IAPR.