A New Laplacian Method for Arbitrarily-Oriented Word Segmentation in Video

P. Shivakumara¹, M. Suhil², D. S. Guru² and C. L. Tan³
¹University of Malaya (UM), Kuala Lumpur, Malaysia
²Department of Studies in Computer Science, University of Mysore, Mysore
³School of Computing, National University of Singapore, Singapore
hudempsk@yahoo.com, mahamad45@yahoo.co.in, dsg@compsci.uni-mysore.ac.in, tancl@comp.nus.edu.sg

Abstract—Word segmentation from video text line is challenging because video poses several challenges, such as complex background, low resolution, arbitrary orientation, etc. Besides, word segmentation is essential for improving text recognition accuracy. Therefore, we propose a novel method for segmenting words by exploring zero crossing points for each sliding window over text line. The candidate zero crossing points are defined based on characteristics of positive and negative Laplacian values at text region and non-text region. The percentage of candidate zero crossing points is calculated for each sliding window and is used for identifying the seed window that represents space between words. For the seed window, we propose a novel idea of horizontal and vertical sampling based on the percentage values to estimate the width and the height of the word spacing. Then the width and the height of the word spacing are used to validate the actual word spacing. Experimental results comparing with an existing method show that the proposed method is better than the existing method in terms of recall, precision and f-measure on curved, horizontal, non-horizontal, Hua’s video data, as well as ICDAR data. We also test it on our own data containing multiscr ipt text lines to show the robustness of the proposed method.

Keywords: Video text line, Zero crossing points, Text candidates, Horizontal sampling, Vertical sampling, Word spacing.

I. INTRODUCTION

Over the last decade, text detection and recognition has become a popular topic in the field of document analysis, image processing and multimedia because of its useful applications, such as assisting blind people, extracting exciting events in sports video, labeling events with semantics. However, due to the low resolution and complex background of video, text detection and recognition is still a challenging problem [1-3]. In addition, the presence of graphics and scene text in video of any orientation makes the problem more complex [4]. Therefore, achieving good accuracy for both text detection and recognition is still an elusive goal for researchers. It is noted from the paper [5, 6] that one of the major problems led to poor accuracy for text detection and recognition is the lack of proper word segmentation methods for text line after detecting text lines from video frames. We are motivated from the work [5, 6] for text detection in video, where the methods give good accuracy at text line level but low accuracy at word level because segmenting exact word exactly from video text lines is hard in unconstrained environment, particularly due to low resolution, complex background, arbitrary text orientation and different scripts.

To overcome the problem of word segmentation and character segmentation from video text lines and to achieve good recognition rate, we can see many methods [7-11] that have been developed so far. For instance, a Fourier-moments based method [8] is proposed for word and character segmentation from text lines, which uses clustering and run-length criteria to segment words, and text height difference at character boundary for segmenting characters. A gradient vector flow based method [9] for video character segmentation was developed. Gradient flow is used for identifying seed points and the least cost path is used for segmenting the characters. The combination of profile based feature for seed point guessing and cost path [10] is proposed for character segmentation from arbitrarily-oriented text lines. This method is sensitive to seed point selection and background. However, these methods are limited to character segmentation but not words. It is stated that to achieve a good recognition rate with the available OCR, feeding words as input to OCR is better than feeding characters [8, 9]. Therefore, words are preferable than character. Besides, most of the segmentation methods use binarization to verify the gap between words and characters. Thus, these methods do not give consistent results for low resolution like video. Recently, word segmentation based on direction of text pixel is proposed [11]. However, the method performance depends on binary information given by the direction values. Word spacing is identified by using geometrical features, such as height and width of connected components. Therefore, if a binary image does not provide fine details then the method loses accuracy.

In this work, we propose a novel idea for segmenting words from video text lines of any orientation, script type, contrast and background by exploring Laplacian zero crossing points at the text region and near text regions. The main advantage of this method is that the method does not use binarization and depends on geometrical features of connected components unlike existing methods [7, 8, 10, 11]. This is the main contribution to achieve good accuracy. The basis for this method is that the percentage of zero crossing points will be higher where there is a text and will be less where there is no text. This observation is inspired by the work proposed in [12] where it is shown that Laplacian positive and negative peaks give clues for identifying presence of text and non-text in video frames. In addition, these features are invariant to rotation, scaling, script type. One example is shown in Fig. 1 where for
edge pixels we get high positive and negative peaks and low peaks where there is no text.

![Graph](image)

**Fig. 1:** High peaks for text pixels and low peaks of non-text pixels

II. PROPOSED METHOD

In this work, a text line detected by a text detection method is the input for word segmentation. For this purpose, we propose to use the method [13] as it works well and withstands the problems caused by video. In addition, this method gives good accuracy for arbitrary text detection compared to state of the art methods. Since the proposed method considers a text line as input, it is easy to identify the end of the text line. And also, text detection method gives the direction of the text line as it deploys growing methods to extract a full text line. Therefore, the proposed method takes advantage of the direction of the text line given by the text detection method to study the Laplacian positive and negative values over text lines. Since our intention is to study the Laplacian zero crossing points over text lines, we perform sliding window operation for which the method extracts zero crossing points. The sliding window size is defined as the height of the text line as width and the same width length is considered as height. This results in a square widow. For each sliding window, we extract horizontal and vertical zero crossing points and we select candidate zero crossing points based on the fact that the positive and negative peaks at two zero crossing points of text pixels have the same height, distance equal to stroke width distance and the gradient direction. It is noted that two zero crossing pixels with respect to outer contour and inner contour of the character, which represent stroke width have the same height of positive and negative peaks due to strong responses at edges. Similarly, it is noted [14] that the gradient direction of such two zero pixels have the same direction. The stroke width is estimated as in [15] using edge and gradient information. In this way, we explore positive and negative peaks of Laplacian operation to identify the candidate zero crossing points in a novel way to study the space between words. With the help of candidate zero crossing points of each sliding window, we classify low percentage of candidate zero crossing points as probable word spacing. From the probable word spacing, we select one seed window which has the lowest percentage of zero crossing points among all probable windows. Then we apply a new method called horizontal sampling to the seed window to find space between the words. This results in the width of the space. Similarly, we apply vertical sampling for the same seed window to find the height of the text. The width and height of the seed window are verified with the probable windows corresponding to the classified low percentage candidate zero crossing points to find the actual word spacing.

A. Zero-Crossing Points for Seed Window

For each text line detected by a text detection method shown in Fig. 2(a), the method finds the first end of the text line and the size of the square window with the help of growing and extraction of text lines by the text detection method. The height of the text line is identified by finding the number of pixels in the perpendicular direction of the text line. The sliding window is formed with a size equal to the height of the text line, it is then placed at the beginning of the text line and is allowed to move pixel by pixel in the direction of the text line as shown in Fig. 2(b) where the window is moving pixel by pixel over text line with complex background.

![Graph](image)

**Fig. 2:** Sliding window for candidate zero crossing points

For every pixel in sliding window, we perform horizontal and vertical Laplacian operation to count the number of zero crossing points as an example of zero crossing points shown in Fig. 1. It is true that for complex background shown in Fig. 2(a), the Laplacian operation gives many noisy zero crossing points. Therefore, we propose new criteria to select zero crossing points which represent text pixels called candidate zero crossing points.

Consider any two zero crossings in the result of Laplacian operation, say P(x1,y1) and Q(x2,y2), the following three criteria should be satisfied by them to verify that they are candidate zero crossings points.

1. **Peak Height**: Peak heights obtained by zero crossings (both positive as well as negative) at P and Q must be equal.

   \[
   \text{PeakHeight}(P) = \text{PeakHeight}(Q)
   \]

2. **Stroke Width**: The distance between these two zero crossings must be equal to the stroke width.

   \[
   \text{dist}(P,Q) = \text{strokewidth}
   \]

3. **Gradient Direction**: Gradient directions of both P and Q must be equal.

   \[
   \text{GradDir}(P) = \text{GradDir}(Q)
   \]
We compute the Percentage of Candidate Zero Crossing Points (PCZP) for each sliding window. Fig. 3(a) shows that the percentage of candidate zero crossings points for the input image shown in Fig. 2(a) for all sliding windows over text line without any conditions. Fig. 3(a) shows that since all zero crossings of Laplacian operations are considered, we get the maximum number of PCZP for windows not belonging to text regions. Therefore, PCZP without any conditions is not useful for word spacing. For instance, the minimum PCZP for the 83rd window is the window that starts near the second word in the text line shown in Fig. 2(a). But, at the same time the curve also has the minimum PCZP for the 63rd window which actually covers the space between the first and second words in the text line.

The effect of condition-1 can be seen in Fig. 3(b) where one can see most of the non-candidate zero crossings. However, the 63rd window in Fig. 3(b) is still showing the minimum PCZP whereas, the 83rd window is not.

The effect of both condition-1 and condition-2 is illustrated in Fig. 3(c) where it can be seen that some more non-PCZP are eliminated to get a slightly different curve than in Fig. 3(b). But, we still have some non-PCZP present in the Laplacian result due to the poor performance of the stroke width identification.

The effect of condition-1, condition-2 and condition-3 all together is illustrated in Fig. 3(d) where the windows from 62 to 65, 130-135, 155-158 are showing least PCZP when compared to the other windows and it is confirmed that these windows represent correct word spacing.

It is observed from Fig. 3(d) that where there is no text pixels, the corresponding window gives low PCZP and where there is text the corresponding window gives high PCZP. Since there is a gap between low and high values and our objective is to separate low PCZP from high PCZP, we apply k-means clustering with \( k = 2 \). This is because low PCZP represents probable window spacing. Since k-means clustering is unsupervised and it considers random guess for clustering, we select maximum and minimum PCZP to give as initial guess to k-means clustering. The lower mean of cluster out of the two clusters is considered as the probable word spacing cluster. The probable windows which represent word spacing classified by k-means clustering is shown in Fig. 4 where red color dots denote probable word spacing window for the text line shown in Fig. 2(a). The window which gives the lowest PCZP among probable windows is considered as the seed window. The seed window is used for identifying actual word spacing. It is presented in subsequent sections.

B. Horizontal Sampling for Identifying Potential Word Spacing Windows

For the selected seed window, we sample the window horizontally using the same criteria used for identifying the PCZP to locate word spacing. First we go to that seed window over the text line, increase the window size by one pixel, calculate again PCZP. Check whether this PCZP is increasing compared to PCZP of the original seed window. If so, this indicates there is a text but no space. Next, decrease the
window size from left to right by one pixel from the actual window size, calculate PCZP and check PCZP with the PCZP of the actual seed window. It should be decreasing or remain the same if there is space. If so, we continue decreasing the window size pixel by pixel until no decreasing is possible. Next, the same process of decreasing window size by one pixel, finding out the PCZP and comparing with the original PCZP of seed window is repeated from right to left. If there is word spacing within the seed window then, definitely the difference between the points, one point obtained from left to right and the other point obtained from right to left, would be a positive value. Because, the left to right processing takes us to the location where the first character of the second word begins and right to left processing gives us the location where the last character of the first word ends. If we count the number of pixels from the starting seed window size to the last reduced window size, it gives the width of the space and we call it the width of word spacing. We check whether this width of word spacing is equal to the word spacing of probable windows or not. The window that finds this match is considered as the potential window of word spacing as shown in Fig. 5, where for the first image in Fig. 5(a), the method detects word spacing correctly while for the second image, the method falsely detects word spacing (red mark over "O"). Therefore, we propose vertical sampling to eliminate false word spacing.

![Image](https://via.placeholder.com/150)

(a). Horizontal sampling results (potential word spacing)

![Image](https://via.placeholder.com/150)

(b). Vertical sampling results

**Fig. 5: Actual word spacing detection**

C. Vertical Sampling for Verification of Word Spacing

For each potential word spacing given by horizontal sampling shown in Fig. 5(a), we employ vertical sampling to detect actual word spacing. First, we perform vertical sampling for the seed window in the same way as horizontal sampling to find distance vertically. This results in the height of the text. Then we match this height of the seed window with the potential windows given by the horizontal sampling. If the height of the potential window is equal to the height of the seed window then it is considered as the actual word spacing or else it is considered as false word spacing. This effect can be seen in Fig. 5(b) where for the first image, there is no change while for the second image, the false word spacing shown in second image in Fig. 5(a) is removed. This is because we don’t get the same distance value as the height of the text line if there is a character like ‘O’. The method also identifies the actual word spacing for arbitrary orientation text lines in the first image in Fig. 5(b).

III. EXPERIMENTAL RESULTS

We evaluate the proposed method by testing on our own dataset as there is no benchmark dataset available in the literature for word segmentation in terms of recall, precision and f-measure. We extract 70 arbitrarily oriented text lines from video frames (including curved text but excluding non-horizontal and horizontal straight lines), 325 non-horizontal text lines, 1047 horizontal text lines, and 93 text lines from publicly available Hua’s data [16]. Note that arbitrary data may contain multi-oriented characters, words and lines, while non-horizontal data may contain only multi-oriented text lines but not characters and words. To test the robustness of the proposed method, we test it on 125 text lines extracted from the ICDAR-2003 competition dataset which contains high resolution images since these are camera based images but not video images [17]. In addition, we conduct experiments on multiscipt video text lines of 100 samples which include Kannada, Hindi, and Telugu scripts of Indian languages. Since Indian scripts are more cursive than English, it is a complex data. In summary, 1635 (70+325+1047+93+100) video text lines and 125 text lines from camera images are used for the purpose of experimentation. To study the effectiveness of the proposed method, we compare with the recently developed method [11] for word segmentation. The method basically obtains binary image for the input text line and then a series of heuristics based connected component analysis are proposed to segment the words. We use the following definitions which are used in measuring the method in [11].

**Truly Detected Word (TDW):** A segmented block that contains correctly segmented words. **Under Segmented Blocks (USB):** A segmented block which contains less number of words than expected. **Over Segmented Blocks (OSB):** A segmented block that contains more number of words than expected. Since there is no ground truth available, we manually count the Actual Number of Word (ANW) in the text lines and it is considered as ground truth for evaluation. We use the standard Recall(R), Precision(P), and F-measure(F) as performance measures. The performance measures are defined as follows. Recall (R) = TDW / ANW, Precision (P) = TDW / (TDW + USB+OSB), F-measure (F) = (2 × P × R) / (P + R).

A. Experimental Results

The qualitative results of the proposed method for word segmentation on curved, non-horizontal, horizontal, Hua’s, ICDAR and multiscipt data are shown respectively from (a)-(f) in Fig. 6, where one can notice that the proposed method detects words spaces correctly for different kinds of data including low contrast text, complex background, arbitrary orientation and multiscipt text lines (Hindi, Kannada and Telugu). It is observed from Fig. 6(e) that the proposed method is good even for high resolution data despite the fact that the method is developed for video data. Fig. 8(f) shows that the proposed method is robust to different script data especially Indian scripts where characters are more cursive than English text.

B. Comparative Study

The quantitative results of the proposed and an existing method [11] for all dataset are reported in Table 1. For fair evaluation, we use the same dataset chosen by the existing method for curved, non-horizontal, horizontal, Hua’s and ICDAR data. Since the existing method does not experiment on multiscipt data and it is our own data created by us, there is no comparative study with the existing method in Table 1. Table 1 show that the proposed method is better than the existing method for all data in terms of f-measure. This is
mainly because the existing method loses the text information while binarizing the input text line image with text pixel direction information. Therefore, it affects the subsequent steps and hence poor accuracy. It is observed from Table 1 that the proposed method gives low accuracy for curved data compared to other data because handling arbitrary data is more complex than non-horizontal and horizontal data. It is evident that the highest accuracy is given by the proposed method for horizontal data. It is also noted from Table 1 that low accuracy is reported for multiscript data compared to non-horizontal and horizontal English data. This is because multiscript data especially Indian scripts involve more cursiveness and modifiers while English does not. Therefore, we can conclude that the proposed method is good for words segmentation of different datasets.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
<td>R</td>
<td>P</td>
</tr>
<tr>
<td>Curved dataset</td>
<td>0.65</td>
<td>0.7</td>
</tr>
<tr>
<td>Non-Horizontal Straight Line</td>
<td>0.83</td>
<td>0.86</td>
</tr>
<tr>
<td>Horizontal Straight Line</td>
<td>0.88</td>
<td>0.92</td>
</tr>
<tr>
<td>Hua data</td>
<td>0.85</td>
<td>0.86</td>
</tr>
<tr>
<td>ICDAR 2003</td>
<td>0.84</td>
<td>0.87</td>
</tr>
<tr>
<td>Multiscript data</td>
<td>0.76</td>
<td>0.85</td>
</tr>
</tbody>
</table>

IV. CONCLUSION

We propose a novel idea for segmentation of words from arbitrary text lines of different data by exploring Laplacian positive and negative peaks without binarization. The three criteria are proposed based on zero crossing points to identify percentage of candidate zero crossing points for each sliding window over text line. The percentage of candidate zero crossing points are used to find a seed window. Then horizontal sampling is done on the seed window to identify the potential word spacing window. The potential word spacing windows are verified by the vertical sampling done on the seed window. Experimental results on different datasets show that the proposed method is superior to the existing method in terms of f-measure. We are planning to extend this idea for improving accuracy for curved and multiscript data in future.

REFERENCES