IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 3, Issue 1 (Sep. – Oct. 2013), PP 10-14
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

An Edge-Based Method for Detection of Multi-Oriented and Multi-Scale Scene Text in Images

Anisha. R. S[1], Salim Paul[2]
[1](*Student, Dept.of ECE, SCT college of Engg. India)*
[2](*Asst. Professor Dept.of ECE, SCT college of Engg., India)*

Abstract: *Text in images provides essential information of the image content and thus text detection is fundamental to indexing of an image database. The existing methods do not provide effective results for images with different text orientation and text sizes. This paper presents an edge based algorithm that uses connected component analysis for handling non-horizontal text strings and Gaussian scale space for multi-scale texts. The experimental results show the effectiveness of this approach in detecting text blocks irrespective of its orientation and size.*
Keywords*: Text detection, Text orientation, Text size, connected component analysis, Gaussian scale space*

I. Introduction

With the rapid growth of available multimedia documents, retrieval of video clips or images of a particular event from large collections without any knowledge of the original data becomes a challenging problem in multimedia indexing. Traditional method of labeling the database using manually tagged keywords is laborious and inconsistent, since different users may use different keywords for the same document. An alternative to this method is to generate keywords from the text information already present in the video or image, in the form of artificially added graphics text, or scene text as part of natural scenes. Scene text includes road signs, advertisement boards, license plate of vehicles etc.

Texts in images include much semantic information related to the content of image or video. This is advantageous than the other semantic contents like text, face, vehicle, and human action in that: (i) it is very useful for describing the contents of an image; (ii) it can be easily extracted compared to other semantic contents, and (iii) it enables applications such as keyword-based image search, automatic video logging, and text-based image indexing [1]. Thus, text detection and extraction can be made useful for the automatic annotation, indexing, and structuring of images. Other than this, extraction of text information has numerous applications including document analysis, vehicle license plate extraction, technical paper analysis, and object-oriented data compression etc. [1].

Text detection deals with detecting and locating those regions that contain texts from a given image and generating an output that can be recognised by a character recognition system (OCR). Although many methods have been proposed for this purpose, text variations related to size, style, colour, orientation, and alignment, as well as low contrast and complex backgrounds make the problem of automatic text detection extremely challenging. The different text detection methods can be classified as: connected component (CC)-based, edge-based and texture-based.

CC-based methods use a bottom-up approach by region growing until all regions are identified in the image; or a top-down approach by successive splitting of image regions. These methods are based on the assumption that the text pixels in the connected region share some common properties [2] [3]. CCs may not preserve the full shape of characters for images with complex background and low contrast text lines.

To overcome the problem of low contrast, edge-based methods are proposed. Edge-based approaches are considered useful for text detection since text regions contain rich edge information. The commonly adopted method is to apply an edge detector to the video frame and then identify regions with high edge density and strength [4]. This method performs well if there is no complex background and it becomes less reliable as the scene contains more edges in the background. Edge-based methods are fast but they produce many false positives for images with complex backgrounds.

To overcome the problem of complex background, the texture-based approach [5] considers text as a special texture. These methods apply Fast Fourier Transform, discrete cosine transform, wavelet decomposition and Gabor filters for feature extraction and usually involves the use of classifiers such as SVM and neural networks and thus, they are trainable for different databases. However, these classifiers require a large training set of text and non-text samples. It is especially hard to ensure that the non-text samples are representative.

The existing methods focus on detecting horizontal text detection and are not effective for texts occurring at arbitrary orientation and varying sizes. A method proposed in [6] is for extracting multi-oriented

text but this method is limited to graphics text of a few directions (0, 15, 30 degrees and so on). The method proposed in this paper is an edge based technique that can be used to detect and extract texts of different size and arbitrary orientation. The false alarms that are frequent in the text detection process are minimised in this algorithm. Character recognition is not considered here.

II. Methodology

The block diagram of the method is shown in Fig.1. As shown in the figure, first step is preprocessing stage in which noise removal from the input image ensured. Then, for analysing the text in different scales, scale space analysis is performed. Then for each scale spaces edge detection is performed where text edges are detected along with the background edges. Then to magnify the difference in text and non-text areas, maximum difference map is generated and then binarised. The text areas along with false positives are now represented as connected components. These connected components are then segmented so that those can be processed separately. False positives are then eliminated from these segments by using some assumed properties of text strings.

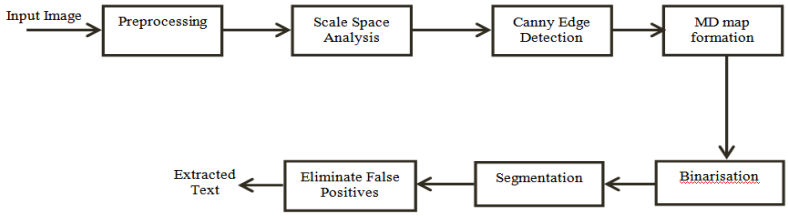

Fig.1: Block diagram of the proposed method

2.1. Preprocessing:

While considering the images with very low contrast against complex local backgrounds, the preprocessing of the input image is important, so that the difference between text and non-text regions is highlighted. The high frequency components of the image contain information about noise and thus should be cut off. A low pass filter can be used to remove high spatial frequency noise from a digital image, but the discontinuities in step function results in edge distortion. Median filtering is used to preserving edges in an image while reducing random noise. In a median filter, a window slides along the image, and the median intensity value of the pixels within the window becomes the output intensity of the pixel being processed.

2.2. Text Area Localization:

a) Linear Scale Space Analysis:

Scale space representation is for handling image structures at different scales, and therefore can be used for dealing with size variations of text in the images. By means of scale-space representation, a single image can be represented as a one-parameter family of smoothed images, parameterized by the size of a smoothing kernel.

The essential requirement for generating the linear scale space is that the use of the smoothing kernel does not create new structures when going from a fine scale to any coarser scale. Several different axiomatic derivations confirm the uniqueness of Gaussian kernels for such an application.

A Gaussian kernel is defined as:

$$g(x, y; t) = \frac{1}{2\pi t} e^{-(x^2+y^2)/2t} \quad (1)$$

Then the Gaussian scale-space representation of an image $f(x, y)$ is a family of derived signals $L(x, y; t)$ such that:

$$L(:, :; t) = g(:, :; t) * f(:, :) \quad (2)$$

where * refers to convolution operation over the variables *x, y*, and the scale parameter *t* indicates which scale level is being defined [7]. *t* is the variance of the Gaussian filter. For $t = 0$ the filter *g* becomes an impulse function such that $L(x, y; 0) = f(x, y)$, that is, the scale-space representation at scale level $t = 0$ is the image *f* itself. Higher values of *t* result in representations at different scales.

b) Edge Detection:

To detect the edges in the image, Canny operator [8] is regarded as optimal, and ensure three main performance criteria for edge detection: (i) low error rate; that is no edges occuring in images are missed and there is no response to non-edges; (ii) positioning accuracy; that is the distance between the edge pixels as found by the detector and the actual edge is at a minimum; (iii) the probability of multiple response to a single edge is low. First step in Canny edge detection is preprocessing to eliminate noise by smoothing. However, since the image is already preprocessed for noise elimination, this step can be avoided. Canny algorithm uses four filters

to detect horizontal, vertical and diagonal edges in the image. Using the image gradient and direction calculated from these filters, the algorithm then tracks along these regions and suppresses any pixel that is not at the maximum (non-maximum suppression).

c) MD Map formation:

Maximum difference (MD) is the difference between the maximum value and the minimum value within a local 1 × N window and is computed as:

$$\begin{aligned} \max(x,y) &= \max_{\forall k \in [-\frac{N}{2},\frac{N}{2}]} e(x, y-k) \\ \min(x,y) &= \min_{\forall k \in [-\frac{N}{2},\frac{N}{2}]} e(x, y-k) \\ MD(x,y) &= \max(x,y) - \min(x,y) \end{aligned} \tag{3}$$

The MD map is obtained from the edge detected image by moving the window over the image. Text regions have larger MD values than non-text regions. This helps in further increasing the gap between text and non-text areas so that these areas can be easily separated later.

d) Binarisation:

Now that the difference in text and non-text areas are magnified, possible areas of text have to be detected. This can be seen as classifying the pixels into two clusters, text and non-text. A simple K-means clustering classify the pixels based on the Euclidean distance of MD values. K-means is convenient since it does not require any threshold values for clustering. The cluster with the largest mean is chosen as the text cluster and is assigned a logical value '1' and the other is the non-text cluster assigned logical value '0'. The result of this step is a binary image with white pixels as the possible text locations.

An alternative approach is to use Otsu's method to automatically perform histogram shape-based image thresholding. The algorithm assumes that the image to be thresholded contains two classes of pixels or bi-modal histogram (i.e, foreground and background) then calculates the optimum threshold separating the two classes so that their intra-class variance is minimal. Then the MD map is binarised at the calculated threshold. The small artifacts that may be present in the resulting image are removed by the morphological operation opening.

2.3. Segmentation of text blocks:

The detected text blocks are generally displayed by bounding boxes. However, when dealing with non-horizontal text lines, the rectangular boxes will enclose unnecessary background pixels or result in overlapping boxes. Therefore, the text lines can be displayed using Connected Components (CC). These CCs may be an intersection of different text regions or an intersection of text regions with false positive regions. The false positives correspond to regions with edges of the background in the image. Accordingly, there are two types of CCs: simple and complex. A simple CC is either a single text string or a false positive. On the other hand, a complex CC contains multiple text strings which are connected to each other and to false positives in the background. High contrast text often appears as simple CCs while low contrast text often appears as complex CCs.

In a complex CC, the text part has to be retained to be displayed in the final result, while the non-text part has to be suppressed. Hence, a complex CC has to be segmented into multiple simple CCs so that each can be analysed separately. The segmentation of CC can be done easily after skeletonisation. Skeleton is a well-defined concept in digital image processing to represent the structure of a region. The intersection points of a skeleton show the locations where the sub-components of different orientation are connected to each other. A skeleton segment is a continuous path from an intersection point to either an end point or another intersection point. Thus, a skeleton can be segmented by simply deleting all the intersection points. Then the corresponding sub-component from the complex CC has to be extracted for each segment

2.4. False Positive Elimination:

The result of previous step is a collection of simple CCs which is either a text segment or a false positive. Thus, the final step is to eliminate the CCs corresponding to the false positive so that the end result can be obtained from the remaining CCs. In order to do this, some assumptions have to be made regarding the segments according to the text properties. Those segments that do not satisfy these properties are considered as the false positives and are eliminated. Two of these properties are:

a) Straightness:

It is assumed that the text strings occurring in images appear in a straight line, while the false positives may take irregular shapes. The irregularly shaped text strings may appear as a part of an emblem and are insignificant. Thus it seems fair to assume that the CCs with straightness factor below a particular threshold are

the false positives to be eliminated. The straightness factor for i^{th} segment can be calculated from its skeleton segment as:

$$straighness\,(i) = \frac{length(i)}{mindist\,(i)} \tag{4}$$

where *length(i)* is the length of the skeleton segment and *mindist(i)* is the minimum distance or the straight line distance between the endpoints of the segment. The criteria for elimination is $straighness\,(i) > T_{str}$, a threshold value greater than and close to 1.

b) Edge density

It is also fair to assume that the edges are denser in the text region than in the non-text region. Edge density is calculated as:

$$edgedens\,(i) = \frac{edgelen(i)}{ccarea(i)} \tag{5}$$

where *edgelen(i)* corresponds to total length of all the edges that comes within the region of a particular CC segment, and *ccarea(i)* is the area of the CC segment. Here, the elimination criteria is if $edgedens\,(i) < T_{edns}$, a chosen threshold.

III. Experimental Results

The standard datasets for image text detection like ICDAR mainly focuses on variations in lighting conditions, shadow effects etc., and the images are mostly horizontal and the text size doesn't normally vary within an image. Hence, for the experimental purpose a variety of images were selected with non-horizontal scene text and graphics text of different orientation and scale. A dataset of 100 images was used for evaluating the performance of the proposed method. An example is shown in fig 2. Character recognition is not considered in this work, and therefore the performance evaluation is done at the block level rather than at word level or character level. The performance measures [5], recall and false positive rate are calculated as:

$$Recall = \frac{Number\ of\ correctly\ detected\ text\ blocks}{Total\ number\ of\ actual\ text\ blocks} \tag{6}$$

$$False\ positive\ rate = \frac{Number\ of\ falsely\ detected\ text\ blocks}{Number\ of\ detected\ text\ blocks} \tag{7}$$

Recall for the proposed method was calculated as 90% and False positive rate was 2.2%.

(a) Original image (b) Located text area (c) Final result

Fig.2: Sample result

IV. Conclusions And Future Work

An algorithm for detecting text in different orientation and size is presented in this paper. The use of connected components for representing text regions ensures that the non-horizontal orientation of text in images doesn't hinder its detection. Scale space analysis helps in detecting texts of different scales. The experimental results show effective recall and false positive rates for non-horizontal texts of different size.

In the future, extending the work to develop a character recognition system that takes the obtained result as input to automatically generate keywords is being considered. Also, this should be extended to extract and recognize text from video frames for maintaining a video database.

References

[1] K. Jung, K.I. Kim and A.K. Jain, "Text information extraction in images and video: a survey", Pattern Recognition, 37, 2004, pp.977-997.

[2] Gatos B., Pratikakis I., and Perantonis S.J. (2005), "Text detection in indoor/outdoor scene images", First International Workshop on Camera-based Document Analysis and Recognition (CBDAR'05), Aug. 29, Seoul, Korea, pp. 127-132.

[3] K. Sobottka, H. Bunke and H. Kronenberg, "Identification of Text on Colored Book and Journal Covers", In Proc. ICDAR 1999, pp. 57.
[4] T.Pratheeba, Dr.V.Kavitha and S.Raja Rajeswari, "Morphology Based Text Detection and Extraction from Complex Video Scene", International Journal of Engineering and Technology Vol.2(3), 2010, 200-206.
[5] Q. Ye, Q. Huang, W. Gao and D. Zhao, "Fast and robust text detection in images and video frames", Image and Vision Computing 23, 2005, pp. 565-576.
[6] D. Crandall, S. Antani and R. Kasturi (2003), "Extraction of Special Effects Caption Text Events from Digital Video", Int J Doc Anal Recog 5(2–3):138–157, 2003.
[7] Lindeberg, Tony, "Principles for automatic scale selection", In: B. Jähne (et al., eds.), Handbook on Computer Vision and Applications, volume 2, pp 239--274, Academic Press, Boston, USA, 1999.
[8] http://dasl.mem.drexel.edu/alumni/bGreen/www.pages.drexel.edu/_weg22/can_tut.html