A Robust Algorithm for Text Extraction from Images

Najwa-Maria Chidiac, Pascal Damien, Charles Yaacoub
Faculty of Engineering
Holy Spirit University of Kaslik (USEK)
P.O Box 446 Jounieh, Lebanon

***Abstract*—A robust algorithm that detects text from natural scene images and extracts them regardless of the orientation is proposed. All existing methods are designed to operate under a certain constraint, like detecting text only in one direction. Maximally Stable Extremal Regions (MSER) detector is chosen to extract binary regions since it has proven to be robust to lighting conditions. An enhancement technique for MSER images is designed to obtain clear letter boundaries. Images are then fed into a Stroke Width Detector and several heuristics are applied to remove non-text pixels. Afterwards, detected text regions are fed into an Optical Character Recognition module and then filtered according to their confidence measure. The recognition of characters is not part of the algorithm and the results are only about the detection of text. Our algorithm proved to be effective on blurred images and noisy images as well, based on both subjective and objective evaluations.**

Keywords—MSER, OCR, Segmentation, SWD, Text Extraction.

I. Introduction

Content based analysis in digital images and videos is capturing a lot of interest lately, since metadata, such as tags, offers a limited description about the image. The content referred to might be colors, shape of objects, and any other information deriving from the image [1]. One of the most intriguing content present in images is text, since it accommodates an important amount of useful information regarding to the subject of the image. Various types of images contain text such as book covers, CD covers, nameplates and even billboards.

Text extraction has been used in several applications; for instance, address block location [2], license plate location [3], page segmentation [4] and many others. However, many obstacles are faced when extracting text from images or videos; usually they are either related to the various properties of text existing in an image, like different size, color, font and alignments, or to the complexity of the background and the bad quality of the image. Due to these obstacles, text extraction algorithms have some kind of drawbacks since each one of them relies on a certain assumption. Some consider that text has the same size, same spacing, and same color, while others treat only horizontal or vertical text. On the other hand, even with all the problems we mentioned, text extraction algorithms are usually applied on images with clear and easy to read text, which yields to strong detectable edges on its boundaries. In addition, another property we can benefit from is that text in images appears close to each other creating clusters. Our purpose is to achieve a general algorithm that extracts text under all situations with all kinds of text properties and image qualities.

Several studies on text extraction from images and video have been suggested in the past. These studies can be categorized into two types: texture-based approaches and region-based approaches [1]. Region-based algorithms are based on the properties that distinguish text from background, such as color, edges, intensity, etc. These methods are bottom-up approaches that detect small candidate regions and then group them into text regions. For example, Srivastav et al. generated in [5] an edge map, grouped connected-components by the nearest-neighbor constraints, and then found the stroke widths in several directions in order to keep regions with same stroke widths. Region-based approaches can also be divided into several categories, like gradient/edge-based, color-based, stroke-based [5] and many others. On the other hand, texture-based methods are based on the textural properties of text and are top-down approaches since they extract textural properties from the image and then find text candidate regions. These approaches rely on statistical features, Gabor filters, frequency transform and many other methods to describe the textural properties of text. Although they lead to better classification of text/non-text regions, they are computationally demanding and sensitive to text alignment. Their complexity arises from the bad quality of the image, the complex background and many other obstacles. Other approaches for jointly spotting text locations and recognizing text also exist, such as the method proposed by Jaderberg et al. in [6].

This paper proposes a new text detection and extraction method that overcomes the weaknesses of previous approaches. The input image is first transformed into a binary image and edge detection is applied. Instead of performing a simple thresholding method, Maximally Stable Extremal Regions (MSER) are detected. These regions contain the text components and are appointed as white pixels. However, the resulting binary image does not reveal the exact boundaries of text. For this reason, MSER binary image is enhanced by performing a thresholding operation on each connected-component. Edges are then detected and fed into a stroke width detector where strokes, stroke widths, and connected components are found and filtered. Furthermore, text lines are formed prior to text extraction in order to cut down more non-text pixels and increase the accuracy of the algorithm. Our work is based on Latin text characters since the stroke property used in this study is language dependent, as will be explained later. The performance of the proposed algorithm is then subjectively and objectively evaluated. Objective evaluation is based on *Precision Rate*, *Recall Rate*, and *f-measure*, and results are compared with [9], [13]-[16].

The remainder of this paper is organized as follows. The proposed method is presented in Section II, with sample outputs being presented at different milestones within the algorithm. Results are presented and discussed in Section III, and finally, conclusions are drawn in section IV.

II. PROPOSED METHOD

In this section, we present our method of detecting and extracting text characters in natural scene images. This method is mainly based on MSER and on stroke width. A flowchart diagram of the proposed algorithm is shown in Fig. 1. In the sequel, each block in Fig. 1 is thoroughly explained with a sample output as a practical example.

A. MSER detection

The Maximally Stable Extremal Region (MSER) detector is able to localize all text characters regardless of their size and font, and distinguish them from other non-text regions. In most images, distinguished regions that have distinctive and stable properties are found. In this paper, we are interested in one certain type of distinguished regions called extremal regions. According to J. Mattas [7], these regions are closed, no matter the continuous transformation of the image coordinates or of the image intensities. Text in an image presents a high intensity contrast with background, while presenting a uniform intensity within every letter. Due to these characteristics, we are attentive to finding the extremal regions in the text image; in particular, the maximally stable ones. MSER detector has been labeled as one of the best text detectors since it is robust to many changes, such as changes in lighting for example [8]. Hence, the input to the MSER detector is the intensity image. The purpose of this stage is to obtain a binary image containing all text letters in it. Therefore, MSER regions are labeled in white whereas the others in black, as shown in the example of Fig. 2.

B. MSER enhancement

The binary image plays an important role in detecting and extracting text. For this purpose, MSER detector is adopted whereas simpler thresholding techniques are avoided. However, the resulting image still needs further enhancements, since letter boundaries are not exactly identifiable, especially since the holes within the letters are not distinguishable as can be seen in the example of Fig. 3. Connected components and their bounding boxes are then distinguished from the resulting MSER binary image. Afterwards, for each bounding box collocated position in the original non-binary image, global thresholding is performed

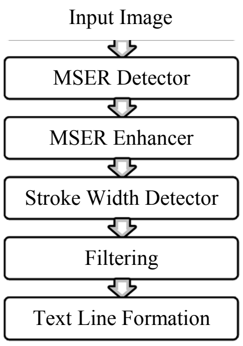


Fig. 1. Flowchart of the proposed text extraction method.

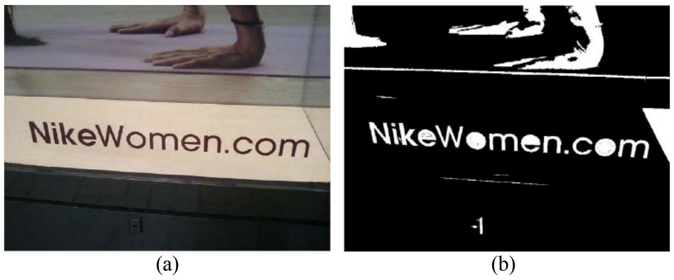


Fig. 2. (a) Grayscale scene image. (b) MSER binary image.

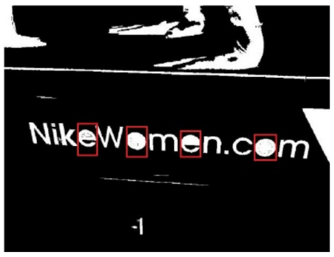


Fig. 3. Emphasizing the drawback in MSER binary images.

in order to obtain a binary component. This binary component is compared to the corresponding binary component from the MSER image in order to keep only text candidate pixels from both components. Fig. 4 shows an example of a MSER binary image before and after enhancement. This stage is very essential to our algorithm since the result identifies the holes within each letter, thus the true stokes.

C. Stroke Width Detection

Stimulated by Epshtein's study based on the Stroke Width Transform [9], we found that stroke information, in particular stroke width information, is very useful to separate text from other components. Stroke is defined as a continuous component in an image which has a band of constant width. According to Epshtein et al. [9], text is distinguished from other components in the image by its constant stroke width. Therefore, in order to extract candidate text components, we apply stroke width transform (SWT) which computes the width of every stroke in order to get at the end an output image where each pixel consists of the width of the stroke it is linked to.

The input to the SWT algorithm is the edge image obtained after applying the Canny edge detector in order to get the object boundaries from the enhanced MSER image, or in other words, to get the strokes. Then, image gradient is computed and an orientation map is thus obtained for the original image. In order for a pixel to belong to a stroke boundary, the gradient direction must be approximately perpendicular to the stroke orientation.

The next step is to find the pixel on the other side of the stroke. Therefore, the ray vector r = [r_x, r_y] is iteratively computed as:

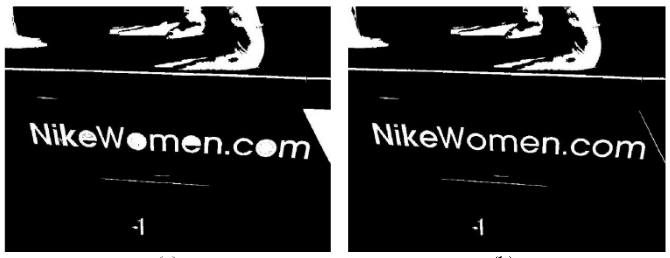
$$\begin{cases} r_x = p_x + n\cos(\theta) \\ r_y = p_y + n\sin(\theta) \end{cases}, \qquad (1)$$

where p = [p_x, p_y] represents the boundary pixel position, n is the iteration index, and θ represents the gradient direction. The computation is done by increasing the value of the index n until a boundary pixel in the edge image is reached. When the two boundary pixels are found, their gradient directions are compared; if they are opposite to each other, all elements belonging to the segment joining the two pixels is assigned with the corresponding width in the SWT image. Contrarily, if the gradient directions of the two pixels are not opposite, the ray is discarded. Fig. 5 reveals the process of finding the width of a stroke. A problem remains with corners, since their true width value is not correctly obtained. As a solution to this problem, we compute, for each non-discarded ray, the median value of all its pixels, and any value found to be above the median is set equal to it. A sample result is shown in Fig. 6.

Neighboring pixels belonging to the same stroke can now be grouped together, thus forming connected-components. For two neighboring pixels to belong to the same group, they must have approximately similar stroke width values, i.e. the SWT ratio between them must be below 3 as mentioned in [9]. This rule allows strokes with similar width to be grouped together as can be seen in the example of Fig. 7.

D. Filtering

The aim of the filtering stage is to determine the components that are unlikely to contain text characters and to remove them. Several heuristics are first applied:



(a) (b)

Fig. 4. (a) MSER binary Image. (b) Enhanced MSER Image.

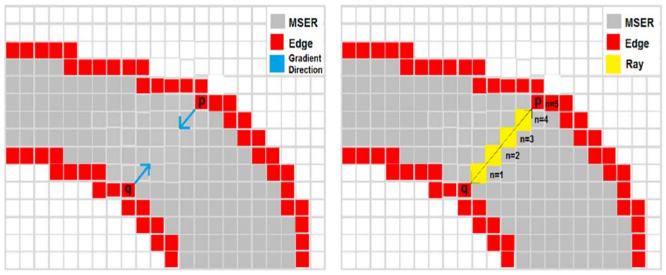


Fig. 5. Example of stroke width detection

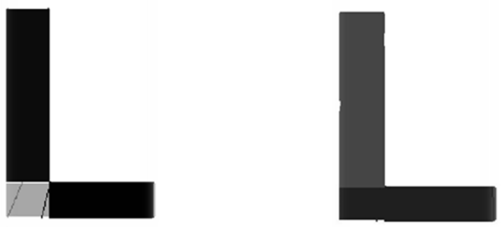
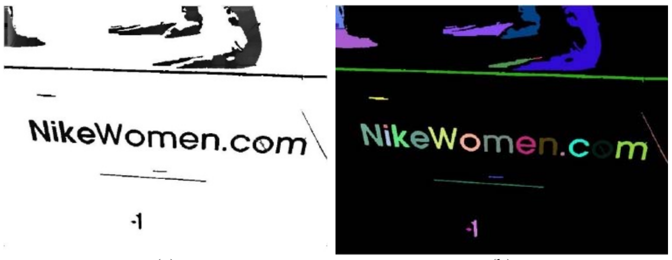

(a) (b)

Fig. 6. (a) Result from SWT detector. (b) Result after median filtering of the non-discarded ray.



(a) (b)

Fig. 7. (a) SWT image. (b) Connected Component Image.

- The *variance of stroke width* within each component must be small since a letter component has constant stroke width as explained earlier. This helps with eliminating a significant number of non-text pixels.

- The *aspect ratio* of each component must be below 1 in order to reject long and narrow components. However, this threshold value is carefully tuned in order to keep letters like "l", "i", "m" and "w".

- The *diameter to stroke ratio* of each component must be below a certain threshold in order to remove long and narrow components.

- The *size* of each component must be readable. This is done by filtering according to the height, width, and pixel count of the component.

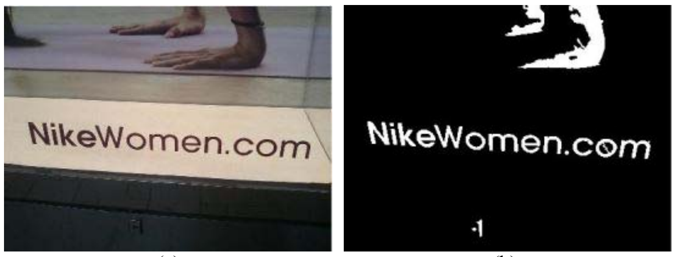
After filtering candidate components by using the above set of rules, some false candidates remain in the resulting images as it can be seen in the example of Fig. 8. A high performance classifier is required since the remaining false candidates are text lookalike components. Among all classifiers, Optical Character Recognition (OCR), which has achieved a state of art in converting different types of text images into editable and searchable data, is selected. Nevertheless, this is not the reason it is selected for, but rather for its confidence measure. The OCR confidence information reveals the certainty/uncertainty of the character. Therefore, it can be used to remove characters with high uncertainty. This helps in removing non-text pixels that have the same heuristics properties as text. Fig. 9 shows the result obtained at the end of the filtering stage for the sample image of Fig. 8. It can be clearly observed that the OCR confidence information allowed to successfully remove non-text components that remained (Fig. 8-b) after applying the different heuristics.

E. Text Lines Formation

The final step consists of forming text lines by grouping connected component with similar height, similar stroke width and constant spacing between letters, and placing them in bounding boxes as shown in the example of Fig. 10.

III. Experiments and Results

In order to test our proposed approach for text localization and extraction, we consider image sets from two different databases: the ICDAR dataset [10-11] and the KAIST Scene Text Database [12]. Although images with different text languages exist, we are only interested in the ones with Latin characters. These databases include different types of scene



(a) (b)

Fig. 8. (a) Scene Image. (b) Result after filtering according to heuristics.

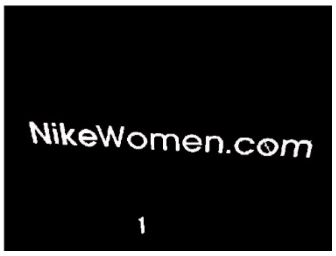
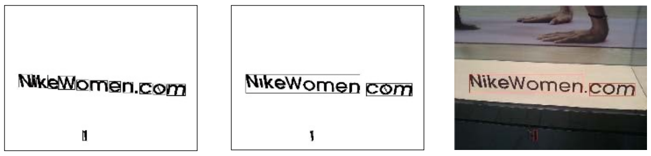


Fig. 9. Result after filtering according to OCR confidence.



(a) (b) (c)

Fig. 10. (a) Connected-Component image. (b) Text Lines image. (c) Text Detection result.

images (e.g. personal, social, commercial, etc…) captured with different conditions and settings (e.g. different imaging devices, lighting conditions, resolutions, etc…).

For each image, the output of our algorithm is a group of bounding boxes indicating the detected text lines. These allow for visual inspection (i.e. subjective evaluation) of the result, as it can be seen in Fig. 11, where 10 different sample results are shown with different text fonts, sizes, colors, and orientations. It can be noticed in Fig. 11 that the proposed algorithm was able to successfully extract all the text in 9 (out of 10) images, whereas only a few characters were missed in one image (second row, left column in Fig. 11). This failure was also encountered with different algorithms, due to the very low contrast between the text characters and the background.

Objective evaluations were also performed, based on the comparisons between the detected bounding boxes obtained from our algorithm and the ground truth bounding boxes supplied with the datasets where text regions have been manually labeled. Hit regions are defined as the overlapping of ground truth bounding boxes with the detected ones. Several metrics were used for objective assessment of our algorithm's performance:

- *Precision Rate* (PR): the ratio of the number of pixels in the hit regions and the number of pixels in the detected regions.

- *Recall Rate* (RR): the ratio of the number of pixels within the hit regions and the number of pixels in the ground truth regions.

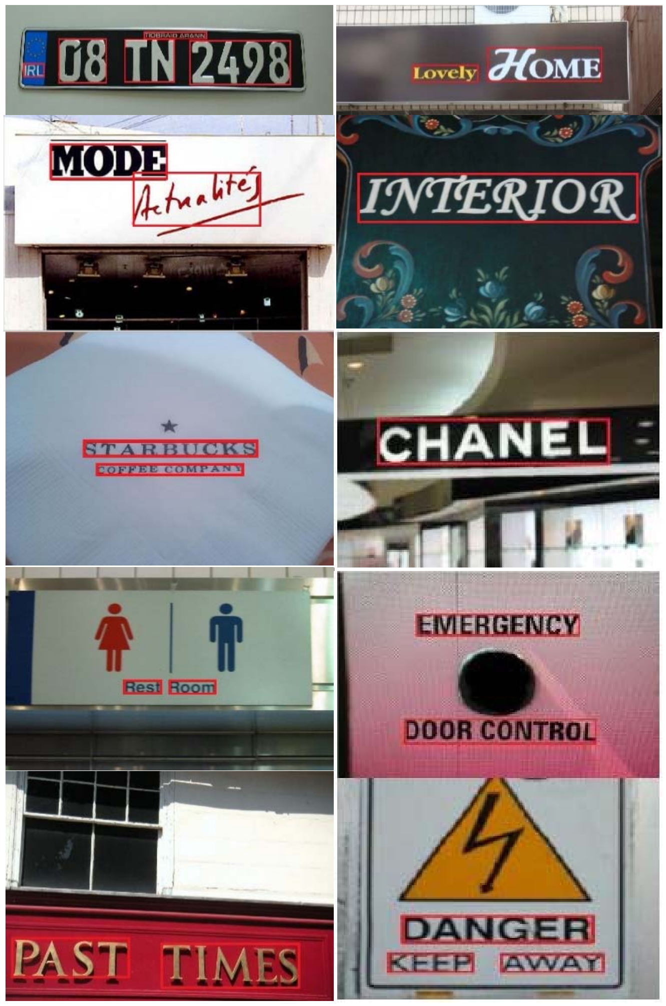


Fig. 11. Sample results.

- The *f-measure*, a quality indicator expressed in (2) as the harmonic mean between PR and RR:

$$f\text{-}measure = \frac{2PR \cdot RR}{PR + RR} \; . \qquad (2)$$

Our results are compared with some of the most performing algorithms found in the literature [9], [13]-[16], and reported in Table I. For fair comparison, we use the same dataset as in [9], [13]-[16]. Our algorithm's performance reached a 90% precision rate, an 88% recall rate, thus an 89% *f*-measure value. It can be clearly noticed that, based on the performance scores in Table I, our proposed algorithm outperforms the other methods in [9], [13]-[16], where the highest RR and *f*-measure values reached were 86% and 76% respectively [14], and the best PR obtained was 82% [15]. As such, our method shows a performance gain of 8% in terms of PR, 2% in terms of RR, and 13% in terms of *f*-measure.

Fig. 12 shows an example where the proposed algorithm fails to detect text areas within the image. The shadow effect in Fig. 12-a results in a distorted MSER image as shown in Fig. 12-b, and thus, subsequent operations in the proposed approach cannot recover from this degradation and as a result, text areas are not successfully detected. Fig. 13 shows another

TABLE I. PERFORMANCE COMPARISION.

Algorithm	METHOD	PR	RR	*f*
Proposed		**0.90**	**0.88**	**0.89**
Risnumawan et Al. [13]	Edgeless stroke based	0.82	0.69	0.75
Xuwang et Al. [14]	MSER-Based	0.68	0.86	0.76
Yi et Al. [15]	Boundary Clustering, Stroke Segmentation, String Fragment Classification	0.81	0.72	0.71
Epshtein et Al.[9]	Stroke-Based	0.73	0.60	0.66
Wahyono et Al. [16]	Fast Stroke Width Transform	0.61	0.63	0.62

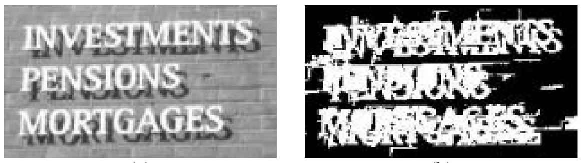


Fig. 12. Failure example. (a) Original image. (b) MSER image.

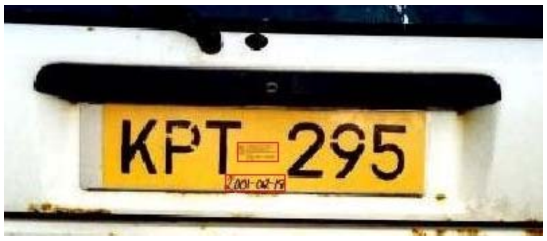


Fig. 13. Failure example: rectangular markers indicate undetected text regions.

failure example, where undetectable text areas are marked with red rectangles; these regions cannot be detected due to character size and very thin strokes.

IV. CONCLUSION

In this paper, we proposed a text detection algorithm based on MSER and stroke width detectors. MSER regions were extracted from the original scene image and refined using a MSER enhancement technique that we proposed, in order to better detect text edges and distinguish holes within each letter.

Enhanced MSER images were fed into a Stroke Width Detector and the different components were filtered according to a set of geometric rules, and then according to their OCR confidence. Text lines were finally obtained by grouping connected components, i.e. components that satisfy some similarity criteria. Our algorithm proved to be effective on blurred images and noisy images as well. However, the algorithm failed in detecting text with shadowing effect, as well as characters with very small size and/or thin strokes. Objective evaluation results showed significant improvement compared to other existing text extraction approaches, where an increase of 13% was obtained in terms of *f*-measure.

REFERENCES

[1] 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.

[2] B. Yut, E. Lansing, and S. Jose, "Address block location on complex mail pieces," Proc. Fourth Int. Conf. Doc. Anal. Recognit., vol. 2, pp. 897 – 901, 1997.

[3] Dong-Su Kim and Sung-Il Chien, "Automatic car license plate extraction using modified generalized symmetry transform and image warping," ISIE 2001. 2001 IEEE Int. Symp. Ind. Electron. Proc. (Cat. No.01TH8570), vol. 3, pp. 2022–2027, 2001.

[4] A. K. Jain and Y. Zhong, "Page Segmentation Using Texture Analysis," Pattern Recognition, Elsevier, vol. 29, no. 5, pp. 743–770, 1996.

[5] A. Srivastav and J. Kumar, "Text detection in scene images using stroke width and nearest-neighbor constraints," TENCON 2008 - 2008 IEEE Reg. 10 Conf., pp. 1–5, 2008.

[6] M. Jaderberg, A. Vedaldi and A. Zisserman, "Deep Features for Text Spotting", 13th European Conference on Computer Vision, Zurich, Switzerland, Sept. 6-12, 2014.

[7] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," Br. Mach. Vis. Conf., pp. 384–393, 2002.

[8] P. Shivakumara, W. Huang, T. Q. Quy Phan, and C. L. Lim Tan, "Accurate video text detection through classification of low and high contrast images," Pattern Recognit., vol. 43, no. 6, pp. 2165–2185, 2010.

[9] B. Epshtein, E. Ofek, and Y. Wexler, "2010 Detecting Text in Natural Scenes with stroke width transform," IEEE Conf. Comput. Vis. Pattern Recognit, no. d, pp. 2963–2970, 2010.

[10] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," 7th Int. Conf. Doc. Anal. Recognition, 2003., vol. 1, no. Icdar, pp. 682–687, 2003.

[11] S. M. Lucas, "ICDAR 2005 Text Locating Competition Results," Int. Conf. Doc. Anal. Recognit. (Icdar), 2005, pp. 0–4, 2005.

[12] Online: http://www.iapr-tc11.org/mediawiki/index.php/KAIST_Scene_Text_Database

[13] A. Risnumawan and C. S. Chan, "Text Detection via Edgeless Stroke Width Transform," the 2014 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), pp. 336-340, Dec. 2014.

[14] X. X.-C. Yin, K. Huang, and H.-W. Hao, "Robust Text Detection in Natural Scene Images.," IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 5, pp. 1–14, 2013.

[15] C. Yi and Y. Tian, "Localizing text in scene images by boundary clustering, stroke segmentation, and string fragment classification," IEEE Trans. Image Process., vol. 21, no. 9, pp. 4256–4268, 2012.

[16] Wahyono, M. Jeong, and J. Kang-Hyun, "Multi Language Text Detection Using Fast Stroke Width Transform," 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV), Jan. 2015.