공학석사 학위논문

# Text Localization in Natural Images Using Multiple Feature Fusion

다양한 Feature의 조합을 통하여

자연 영상에서 글자를 찾는 방법

2015년 2월

서울대학교 대학원

전기·컴퓨터공학부

김 재 석

# Text Localization in Natural Images Using Multiple Feature Fusion

다양한 Feature의 조합을 통하여
자연 영상에서 글자를 찾는 방법

지도교수 유 석 인

이 논문을 공학석사학위논문으로 제출함
2014년 10월

서울대학교 대학원
전기·컴퓨터 공학부
김 재 석

김재석의 석사학위논문을 인준함
2014년 12월

위 원 장 Robert Ian McKay (인)
부위원장 유 석 인 (인)
위 원 Bernhard Egger (인)

# Abstract

## Text Localization in Natural Images Using Multiple Feature Fusion

JaeSuk Kim

School of Computer Science Engineering

College of Engineering

The Graduate School

Seoul National University

Text localization in natural scene images is an important first step to analyze content-based images. In this thesis, we propose an accurate method for detecting texts in natural scene images by adapting multiple feature combinations. Firstly, various color spaces are used to extract Maximally Stable Extremal Regions (MSERs) as character candidates. K-means clustering, image gradients, and Lucy and Richardson de-convolution (LRD) method are used to emphasize significant feature points and to compensate noisy images for character candidates. Secondly, important character candidates will be accentuated by applying enhanced canny edge detector to grayscale image gradients and our cumulative MSERs image. Thirdly, character candidates will be merged into text candidates by using

clustering algorithm and geometric information such that texts usually appear in a linear form. Finally, inconsequential text candidates will be eliminated by using stroke width, text region height, width, and parallel edge features.

The method was evaluated on two benchmark datasets: International Conference on Document Analysis and Recognition (ICDAR) 2013 and Street View Text (SVT) from Google Maps. Experimental results on respective datasets show that the proposed algorithm works successfully with not only the normal text images, but also highlighted, transparent, small, and blurred texts.

**Keywords : Text localization, feature fusion, maximally stable extremal regions, k-means clustering, Lucy and Richardson deconvolution**

**Student number : 2013-20774**

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Text localization and recognition in natural scene images is an important first step to many computer vision based applications. It is to detect texts from images. So Text localization is also known as text detection [10] and word spotting [13]. Since textual information often provides valuable information for understanding the meaning of contents or objects, it can be used in various fields: aids for visually impaired people, vision-based navigation system in urban environments, and image-based translators for foreigners. Although this field has attracted an enormous attention from both engineers and entrepreneur, it is still remained as a challenging problem. This is due to three main issues: 1) the diversity of text patterns such as size, font, shape, color, and orientation, 2) the complexity of scenes in natural images, and 3) the presence of text-like background objects such as bricks, windows, and leaves.

In this thesis, we only focus our research on text localization, not on recognition. This is due to two main reasons. Firstly, in many cases, low recognition rate is caused from low text detection rate; increasing the detection rate will improve the recognition performance. Secondly, there already exist a number of good text

recognition algorithms that can be used after the text is detected.

There exist two main methods to localize text in natural images: region-based and component-based approaches. Region based methods [1], [2] use a sliding window to search for possible texts at different scales in the image and then use classifiers to identify text and non-text regions. This technique is slow as the image has to be processed in multiple scales and depends highly on proper descriptor selections. Connected component-based methods remove the majority of non-text regions (or simply background) and group remaining pixels into regions using connected component analysis assuming that these pixels belonging to the same character have similar properties such as consistency of stroke width[3] and color homogeneity[4]-[6]. Recently, Maximally Stable Extremal Regions (MSERs) based approach has become the mainstream for character candidates in recent works [5]-[8] which can be categorized as connected component based methods. Although MSER-based methods show outstanding results for text localization, it has several problems to be addressed. The problems are: 1) the MSER detector contains a number of non-text candidates, and 2) it has a number of repeating components which cause a problem for character grouping algorithm. As a consequence, the MSERs algorithm still has a plenty of room for improvements in terms of accuracy, efficiency, and speed.

In this thesis, we present a robust approach to localize scene text in natural images. Firstly, MSERs are extracted as text candidates for different color spaces and features: CIELAB, HSI, YCbCr, image gradient, and Lucy and Richardson Deconvolution (LRD) applied image. Different color spaces will complement each color space's drawbacks and be able to find all possible character candidates. Since MSERs are very sensitive to image blurs, clutter, and occlusions, deblured feature is required to remedy MSERs' shortcomings. Secondly, enhanced canny edge detector

is applied to grayscale image and our cumulative MSER mask to emphasize more important character candidates; texts often contain strong edge properties. Finally, non-text candidates will be filtered out by using stroke width, text region height, width and parallel edge features. In other words, our thesis also follows the basic framework of MSERs as described in Fig. 4 with three novel ideas.

The rest of this paper is organized as follows. In section 2, an overview of previous works is introduced. The proposed scene text localization is described in section 3. Section 4 discusses experimental results. The paper is concluded in Section 5.

# Chapter 2

# Previous Work

A large number of methods are proposed in the field of text localization and recognition. A large portion of them focuses solely on text localization in real-world images [2], [3], [9], some only deal with text recognition [11], [12], and some deal with both text detection and recognition[5], [31], [32], [34]. More comprehensive surveys on text localization algorithms are mentioned in [23], [24].

This chapter is to review previous works related to only text localization that are based on connected-component approach. Four different methods will be explained: Stroke Width Transform (SWT), MSERs, mean-shift clustering, and object-based text detection. Among them, two most widely used approaches, SWT [3] and MSERs [4], [10], their methods will be overviewed in more depth and discussed with respect to its advantages and disadvantages. These two approaches will be compared to my proposed method. Region-based method will not be treated in this chapter since it has not been published in years.

## 2.1 SWT

Epshtein et al. proposed SWT that utilizes the knowledge that texts usually have the same stroke width to detect texts. The process is shown in Fig. 1. It converts an input image into the grayscale image (a) and applies canny edge detector to extract the edge map (b). Then it calculates stroke widths for each pixel (c), and extracts texts by measuring the width variance in each component and eliminates candidates by using geometric information. Finally it detects texts by finding fixed stroke width, under the assumption that texts usually have nearly constant stroke width.

SWT is implemented as follows. The initial value of each element of the SWT is set to infinity. By applying canny edge detector, a typical stroke like Fig. 2 (a) will remove the inter-pixels of outer edges and become Fig. 2 (b). Edge location of *p* is a pixel on the boundary of the stroke. Searching in the direction of the gradient at *p*, will lead to finding its corresponding pixel *q*, the other side of the stroke. Then pixels in between *p* and *q* are assigned by the minimum of its current value and the found width of the stroke, $\left\| \overrightarrow{p-q} \right\|$.

After outputting the SWT, grouping neighboring pixels with similar stroke width is performed. The next step is to identify components that may contain texts. If the variance of the stroke width within each connected component is too big, then this component will be rejected; this procedure will help to get rid of background outliers such as foliage and bricks. It also rejects those components which are too long or too narrow, or whose size is too big or too small. It also considers the font height to be between 10 and 300 pixels.

Similar approach was also proposed in [15] and [17] and some extended work was also found in [6]. Huang et al. extends SWT by incorporating color cues of text pixels, which leads to a better performance on inter-component separation and intra-component connections.

Although this method is widely used and is shown to be effective, it is even more sensitive than MSERs in noise and blurry images since it is largely dependent on a successful edge detection.


Figure 1. The process of SWT algorithm.

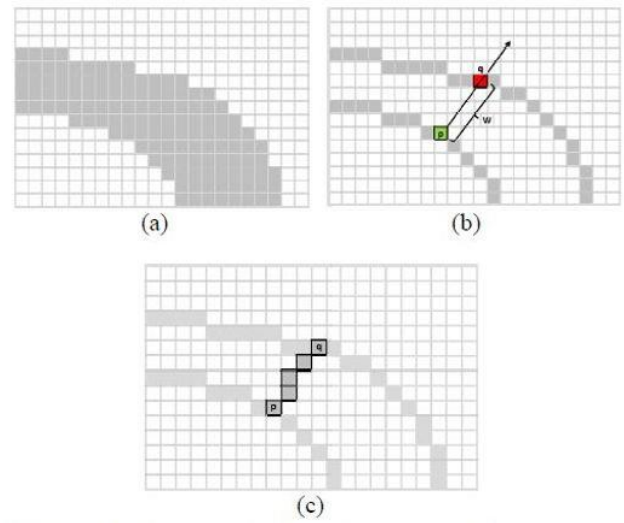


Figure 2. Implementation of SWT.

## 2.2 MSERs

MSERs are first developed in [25] as a method of blob detection in images. This was first proposed to find correspondences between image elements from two images with different viewpoints. MSER is a particular case of an Extremal Region (ER) whose size remains virtually unchanged over a range of threshold, where ER is a region whose outer boundary pixels have strictly higher values than the region itself; the effect of increasing the range of threshold for MSER is shown in Fig. 3. As the Fig. 3 shows, increasing the range will help to find more characters. However, it also groups a number of characters into one region and this could be problematic because it merges both background and characters together.

The fundamental framework of MSERs method is shown in Fig. 4; 1) Natural image is first uploaded, 2) MSERs are extracted as character candidates, 3) extracted features will be grouped into text candidates by using clustering algorithm and geometric information, 4) non-text candidates are eliminated by using a classifier.

MSERs based methods are presented in [4], [7], [8]. Neumann and Matas [4] detect characters as MSERs and use MSERs for text recognition. In [5], they calculate the probability of Extremal Regions being characters in first stage. Once the probability is calculated, only ERs with the locally maximal probability are selected for second stage, to classify whether they are characters or non-characters.

In [10], X. Yin et al. used a pruning algorithm to extract MSERs by minimizing regularized variations. Character candidates are, then, grouped into text candidates

by adapting single-link clustering method. Finally, texts are identified with a text classifier.

Even though there exist a number of modifications to MSERs, the base is to detect all possible character candidates by MSERs. Otherwise, this method will fail to detect texts since there are no character candidates to test on.


Figure 3. The effect of increasing the range of threshold (from left to right).

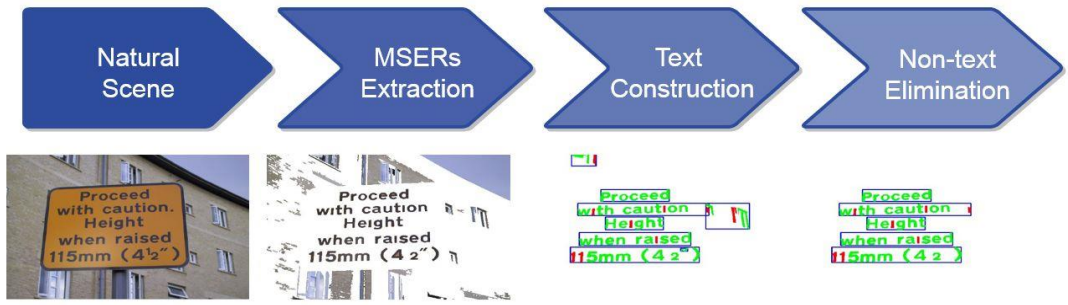


Figure 4. The framework of MSERs method.

## 2.3 Mean-Shift Clustering

Mean-shift clustering is used in [2] and [16]. Kim et al. uses a linear classifier called Support Vector Machine (SVM) to find texts and mean-shift clustering algorithm to the result of the texture analysis [2]. Le et al. uses mean-shift clustering to group similar pixels into clusters first and then considers connected components as text strokes. Although they both show some notable performance, it requires hand-tuned parameters and has high computational cost.

## 2.4 Object Approach

Wang et al. in [13] considered each character as an object. It simply adapted generic object recognition method to text detection. It show that approaching word spotting as a form of object recognition has the benefits of avoiding character segmentation and is robust to small errors in character detection. Despite its first try on considering text localization as object recognition, its performance was not very good. After a year, this method has been extended in [14] performing both text detection and recognition. It uses the region-based approach to find each character and then uses a lexicon to group characters into words. Although this method is robust to noisy images, it only works for an included lexicon; it contains at most 500 words in their experiment.

# Chapter 3

# Proposed Approach

The proposed method improves upon an idea of Neumann and Matas in [5] and Gehler et al. [35] that the combination of multiple channels (intensity, gradient, and HSI) or features can achieve a higher detection rate than a single channel. This shows that different features can extract different MSERs which eventually increases the detection rate. Our method differs from [5] in that we do not simply combine from all channels, but cumulatively give credits to appeared MSERs from each channel to accentuate certain regions. In addition to that, [5] combines the results from each channel, I combine the MSERs and draw one final result.

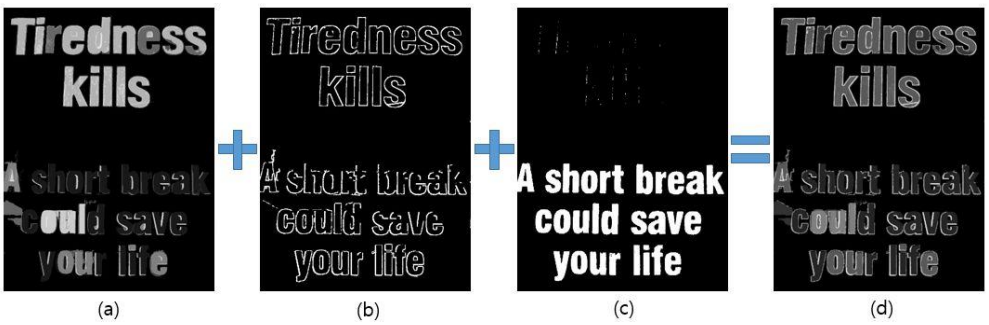


Figure 5. The process of character candidate extraction: (a) cumulative MSERs; (b) edge enhanced map; (c) k-means clustering result; (d) final candidates

## 3.1 Character Candidate Extraction

The process of our character candidate extraction is shown in Fig. 5. As shown in the figure, we use three different masks to output the final cumulative character candidates: MSERs, edge enhanced map, and k-means clustering.

### 3.1.1 MSERs

Fig. 1 shows that character candidates become more obvious as MSERs of more channels are applied. The algorithm works as follows: 1) Create a grayscale image that has the same size as the uploaded image with 0 intensity value; a grayscale image have an intensity range between 0 and 255, where 0 is black and 255 is white. 2) Detect MSERs for each channel and add 1 to the grayscale image for those pixels that are included in MSERs. From now on, we will call the grayscale image cumulative MSERs image. 3) Normalize the image to a range of 0 and 1, which helps to visualize which candidates are more often selected as MSERs. We will call this grayscale image cumulative MSERs and the process is shown in Fig. 6.

The problem of this algorithm is that as more MSERs are added, some candidates become less important; if clouded candidates are text, it becomes a crucial problem since we might lose actual text candidates. Hence we adapt edge enhanced map and k-clustering map to overcome this limitation. In addition to that, LRD is used as one of features since it deblurs the blurry image which leads to a better performance, which is shown in Fig. 7.

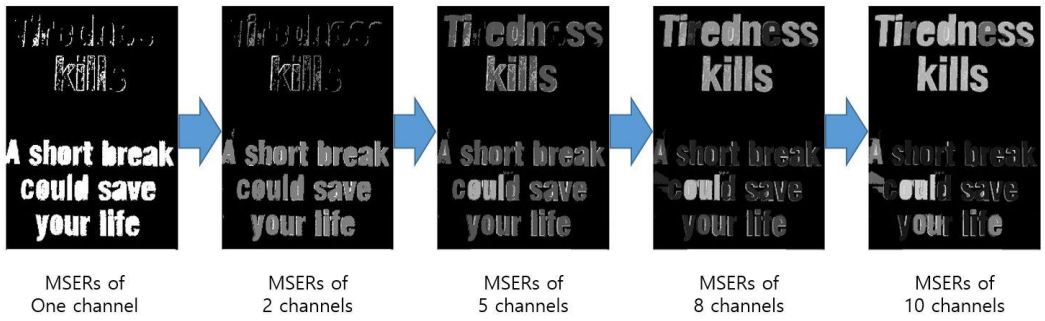


Figure 6. Emphasis of character candidates as MSERs of more channels are cumulated: MSERs of one channel to 10 channels (from left to right).

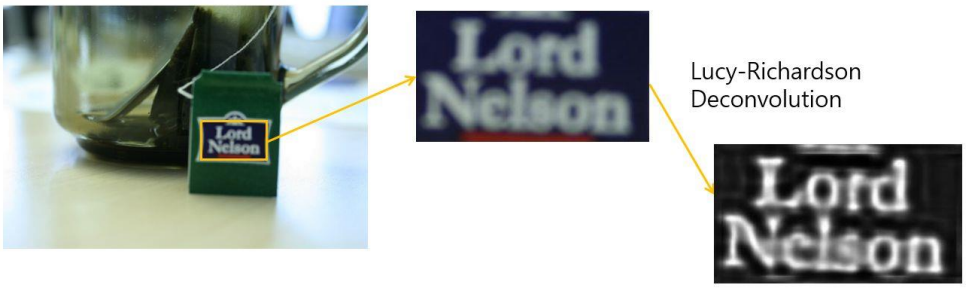


Figure 7. A result of LRD applied to blurred texts.

### 3.1.2 Edge enhanced map

Canny edge detector is used to get rid of pixels that are not part of the text. It has a four step process: 1) A Gaussian blur is applied to remove noises and speckles, 2) it obtains the image's gradient and direction, 3) non-maximum suppression is applied to determine if the pixel is a better candidate for an edge than its neighbors (non-maximum suppression is an edge thinning technique), and 4) hysteresis thresholding finds where edges begin and end. The example is shown in Fig. 8.

We use not only canny edge mask, but also the edge enhanced mask, which grows the edges outward by using image gradients around edge locations; this makes edges clearer and thicker. This mask is also added to our cumulative MSERs with three times weighted values: 3 intensity value. Enhanced canny edge detector is applied to two images: an intensity image and cumulative MSERs. Final edge map is calculated by intersecting two edge maps. Results are shown in Fig. 9.

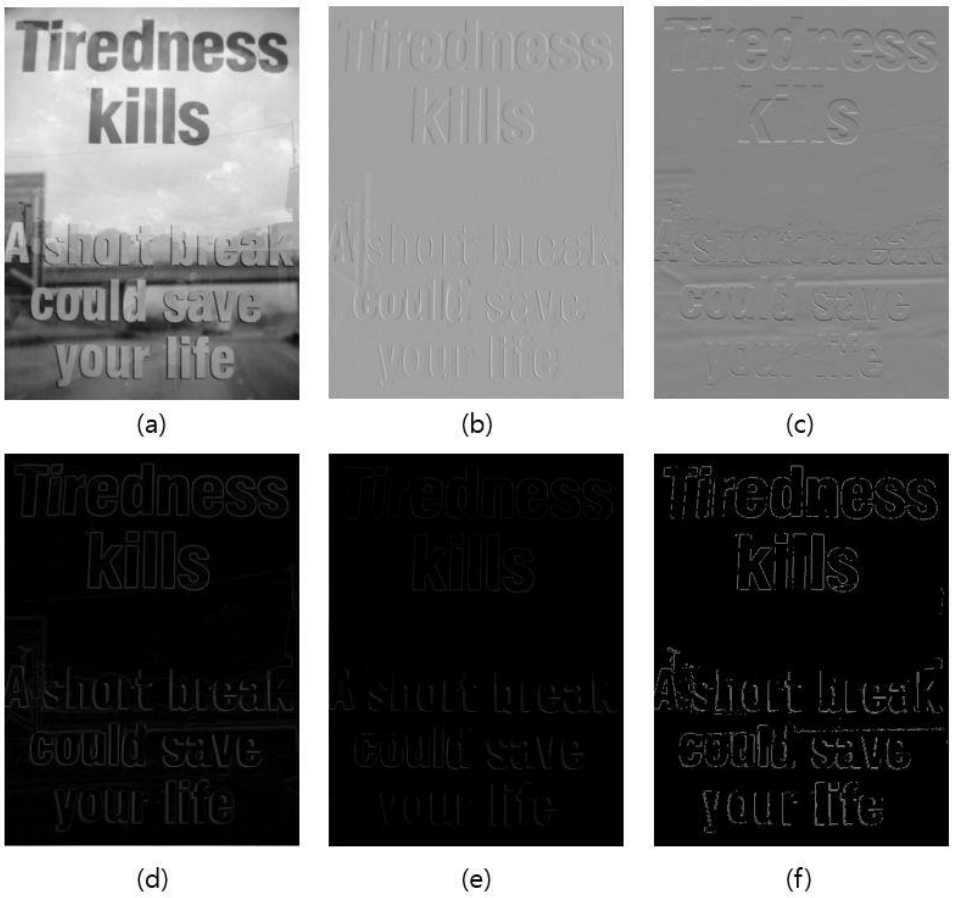


Figure 8. The process of canny edge detector: (a) an intensity map; (b) vertical gradients; (c) horizontal gradients; (d) norm of gradient; (e) after thresholding; (f) final result.

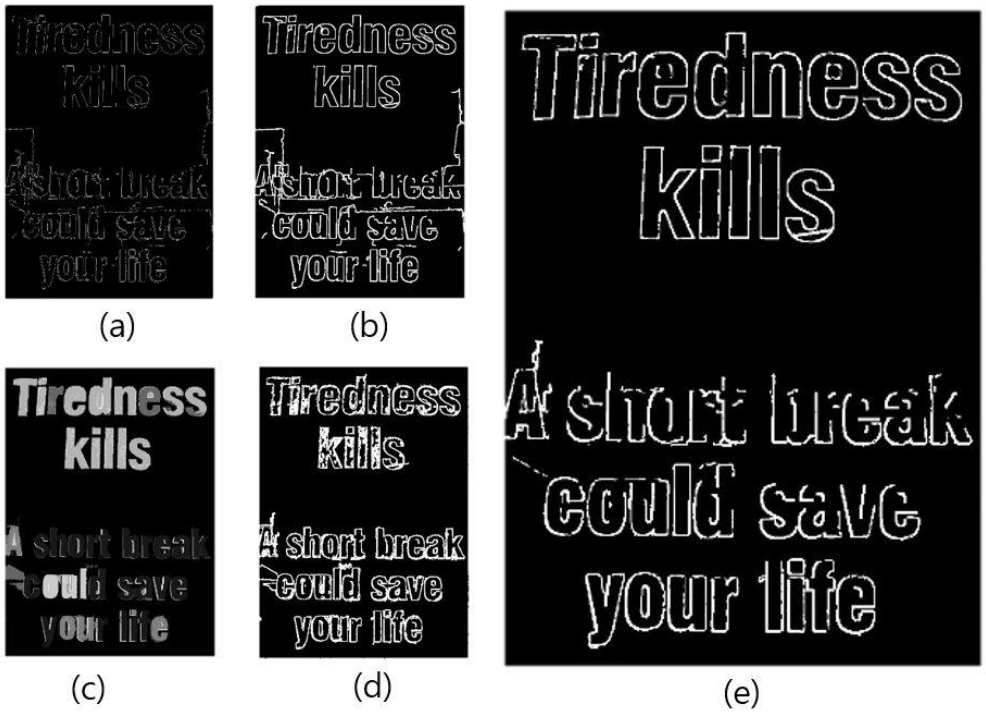


Figure 9. Edge grown maps: (a) canny edge map on the intensity image; (b) edge enhanced map on (a); (c) cumulative MSERs; (d) edge enhanced map on (c); (e) intersection of (b) and (d).

### 3.1.3 K-means clustering

K-means clustering is a very efficient method to detect text candidates since most texts have similar color and intensity values. K-means clustering is a method of vector quantization; it separates *n* observations into *k* different clusters, by minimizing the distance and variance of observation and each cluster.

Since we already have MSERs and enhanced edge map, we will use the result of k-means clustering as an assistance. The reason why we only use two clustering levels

to cluster an original image ($k$ = 2) is because it gives the best result; if we use more than 2 clusters, clustered images have too many noises and even some texts are separated. Experiments on determining the number of clusters are shown in Fig 10. And good and bad examples with two level clustering are displayed in Fig. 11 and 12; since there exists some bad cases, we only use k-means clustering as an assistance.


Figure 10. Tests on determining the number of clusters. From $k$ = 2 to $k$ = 5 (from top to bottom).


Figure 11. Good examples of k-means clustering with $k = 2$.


Figure 12. Bad examples of k-means clustering with $k = 2$.

## 3.2 Non-text Filtering

Since most characters in natural images have similar stroke widths or thickness, stroke width information is used to eliminate non-text candidates; it is not only used for non-text filtering [8], but also for text detection in some cases [3]. However, in our method, it is only used for candidate elimination since our candidate extraction method already shows high performance. In addition to stroke width information, we also use geometric information to remove non-text candidates [3], [8].

### 3.2.1 Geometric Filtering

Character candidates include text and non-text candidates. In order to obtain better detection rate, we must take out some non-text candidates based on their geometric information; height, width, aspect ratio, and object size. 1) Abnormal size of objects are rejected: very large and very small objects. 2) If the letter's aspect ratio is either very big or small, it is rejected. 3) Font height and width should be between 10 and 300 pixels as indicated in [3]. 4) We filter out candidates that contain large number of holes.

### 3.2.2 Stroke Width Filtering

Stroke width information could be used as the final test to remove non-text candidates. Stroke width filtering is adapted to our algorithm by using the function "bwdist" and "helperStrokeWidth" in MATLAB. This algorithm is based on

distance transform; distance transform is a derived representation of a digital image. It converts the binary image into distance map. It is qualified with the chosen metric, and common metrics are Euclidean distance, cityblock distance, chessboard distance, and taxicab distance. An example of a distance transform using Euclidean distance is shown in Fig. 13, and applied test image is shown on Fig. 14(b).

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Binary Image

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 2 | 2 | 2 | 1 | 0 |
| 0 | 1 | 2 | 3 | 2 | 1 | 0 |
| 0 | 1 | 2 | 2 | 2 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Distance transformation

Figure 13. Binary image into distance transform.


(a) (b) (c)

Figure 14. Stroke width images: (a) Binary image; (b) Distance transform of (a); (c) visualization of stroke width.

Once the distance transform is done, stroke width will be calculated as follows:

1) Compare whether eight neighbors are less than the current pixel for all pixels in image.
2) Propagate local maximum stroke values to neighbors recursively.

The result of stroke width transform is shown in Fig. 14(c).

After all the stroke widths are calculated, we compute normalized stroke width variation and compare to common values and if (standard deviation) / (mean) is greater than 0.6, then it is rejected.

# Chapter 4

# Experiments

## 4.1 Environment

To evaluate our method, we tested on two publicly available datasets: International Conference on Document Analysis and Recognition (ICDAR) 2013 [36] and Street View Text (SVT) from Google Maps [26]. In ICDAR 2013 dataset, it has 233 color images to evaluate, with image size varying from 536x263 pixels to 3888x2592 pixels. In SVT dataset, it has 350 color images with image size between 1024x768 pixels and 1914x898 pixels. This dataset is more challenging than ICDAR's dataset because text, presented in images, has various orientation and different font styles, and even images are more noisy and blurry. However, we were not able to test all the test images, but 50 images are selected randomly from each dataset. Experiments were conducted on a desktop computer with a 3.40 GHz Core i5-3570. And the algorithm was implemented in MATLAB R2014a.

## 4.2 Evaluation Metrics

A detection result is usually evaluated by comparing the bounding box of the detected object with the bounding box of the ground truth object: in our cases, object is text. And precision and recall are used for evaluation. However, these two measures have a number of drawbacks: they do not provide intuitive information about the proportion of the correctly detected objects and the number of false alarms. Hence we adapted a better metric algorithm based on [22].

To evaluate the performance numerically, *precision* (P), *recall* (R), and *F-measure* (F) are commonly used metrics. These measurements are computed as follows:

$$p = \frac{\sum_{r_e \in E} m(r_e, T)}{|E|}$$

$$r = \frac{\sum_{r_t \in E} m(r_t, T)}{|T|}$$

$$f = \frac{1}{\alpha / p + (1-\alpha) / r}$$

where the best match $m(r, R)$ for a rectangle r in a set or Rectangles R is defined as:

$$\mathrm{m}(\mathrm{r}, \mathrm{R}) = \max m_p(r, r') | r' \in R$$

and T and E are the sets of ground truth and estimated rectangles respectively. Finally, for f-measure, the relative weights of precision and recall are controlled by

alpha, which we set to 0.5 to give equal weight to precision and recall.

Precision is defined as the number of correct estimates divided by the total number of estimates and recall as the number of correct estimates divided by the total number of targets. Hence, algorithms that over-estimate the number of rectangles will have low precision score and algorithms that under-estimate the number of rectangles (or simply character candidates) will be punished with a low recall score.

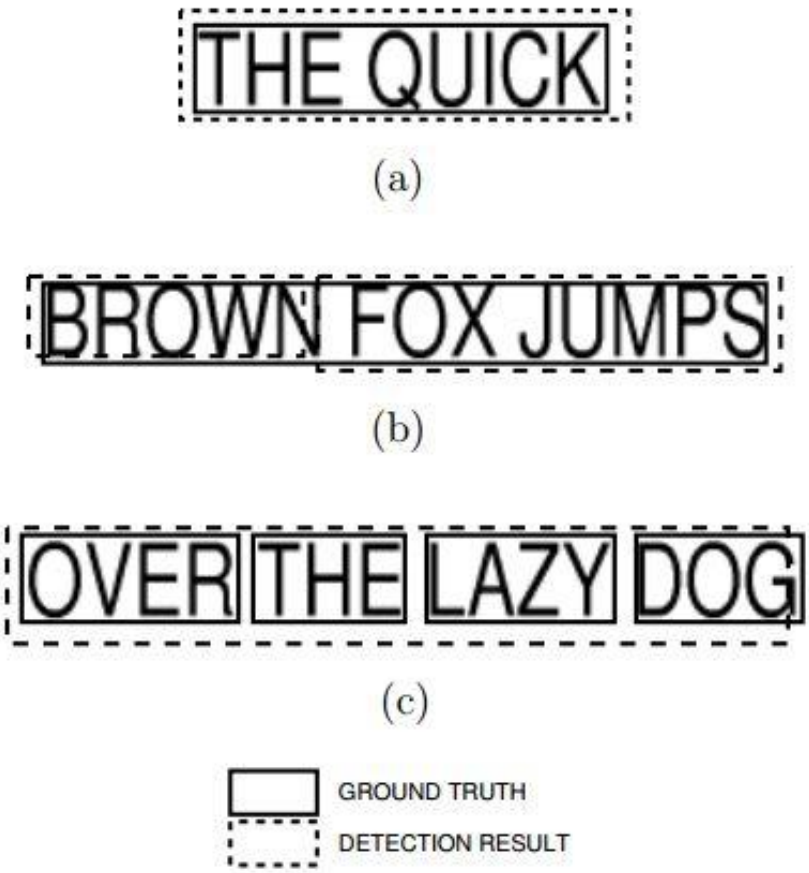


Figure 15. Different match types between ground truth and detected bounding box: (a) one-to-one match; (b) a split: a one-to-many match with one ground truth rectangle; (c) a merge: a one-to-many match with one detected rectangle.

There exists four different match types between ground truth and detected bounding box: one-to-one match, one-to-many match (split), one-to-many match (merge), and many-to-many match. However, we only considered the first three matches since many-to-many match does not occur very often in the case of text detection; even if this situation happens, it can be translated into several splits or a set of splits and one-to-one matches. So our final $m(r, R)$ be as follows:

$$\mathrm{m(r,R)} = \begin{cases} 1 & \text{if r matches against a single detected rectangle} \\ 0 & \text{if r does not match against any detected rectangles} \\ f_{sc}(k) & \text{if r matches against several detected rectangles} \end{cases}$$

where $f_{sc}(k) = \frac{1}{1+\ln(k)}$, and k is the number of rectangles; if it evaluates to 1, then no punishment is given, lower values punish more. Another way is to simply set $f_{sc}(k)$ to a constant value of 0.8 as punishment.

## 4.3 Experimental Results

Experimental results are shown in Fig. 16 to Fig. 18. Figure 16 shows that our algorithm is able to detect texts in transparent, small, and blurred images and images in strong highlights as well. Successful results with some false positives are shown in Fig. 17, and finally the unsuccessful results are shown in Fig. 18.

Our performance is listed in Table 1 for ICDAR dataset and Table 2 for SVT dataset. Although our performance did not achieve the state-of-an-art, our algorithm was able to detect texts in blurred and highlighted images that the state-of-an-art algorithm was not.

## 4.4 Performance Evaluation

We have tested our approach to ICDAR 2013 dataset and SVT. Based on the experimental results, various types of texts in the image are analyzed to evaluate the performance:

1. Small or big
2. Blurred or clear
3. With highlights or without highlights
4. Transparent or non-transparent
5. Simple background or complex background
6. Combination of above 5 kinds.

Our approach was able to detect texts that are either small or big, either blurred or clear, with highlights or without highlights, and either transparent or non-transparent. However, our approach was not able to detect well in complex background. This is due to our relatively simple non-text filtering algorithm; although we were able to find most of the character candidates, we could not efficiently eliminate non-character candidates since they also satisfied our geometric information.

## 4.5 Runtime Evaluation

Although our approach achieved a better detection rate on texts that are hard to detect, our runtime was higher than other approaches. The average processing speed

of the proposed system is 7 seconds when other recent approaches take around 1 seconds: X. Yin et al. takes about 0.43 seconds, the speed of Shi et al.'s method is 1.5 seconds, and the speed of Neumann and Matas' method (including text localization and recognition) is 1.8 seconds per image.

This can be explained in terms of tradeoff. We could increase the speed of processing time by reducing the number of channels and not performing LRD. However, by doing so, a decline in performance was observed. However, a direct comparison is not appropriate because used dataset is different; we used ICDAR 2013 dataset, where other approaches used ICDAR 2011 dataset. In addition, we focused our algorithm mainly on detecting texts that are not detected by previous approaches; accuracy was considered the first priority. This is due to the reason that there exist room to reduce the processing time: implementing in C++, using GPGPU, and optimizing the code. The processing time issue can also be solved as time goes by: improvement in hardware.


Figure 16. Successful results: (a) transparent; (b) large; (c) highlight + small; (d) highlight; (e-f) normal; (g) small + blurred; (h) transparent + separated; (i-j) normal.


Figure 17. Successful results with false positives


Figure 18. Unsuccessful results

Table 1. Experimental results on the ICDAR 2013 dataset.

| Method Name | Recall (%) | Precision (%) | F-score |
|---|---|---|---|
| USTB_TexStar | 66 | **88** | **76** |
| Text Spotter | 65 | 88 | 74 |
| Our method | 67 | 80 | 73 |
| CASIA_NLPR | 68 | 79 | 73 |
| I2R_NUS_FAR | **69** | 75 | 72 |
| TH-TexLoc | 65 | 70 | 67 |
| Text Detection | 53 | 74 | 62 |
| Baseline | 35 | 61 | 44 |
| Inkam | 35 | 31 | 33 |

Table 2. Experimental results on the SVT dataset.

| Method Name | Recall (%) | Precision (%) | F-score |
|---|---|---|---|
| USTB_TexStar | 41 | **66** | **51** |
| Phan et al.'s method [21] | **51** | 50 | 51 |
| Our method | 43 | 56 | 49 |
| Epshtein et al.'s method | 42 | 54 | 47 |

# Chapter 5

# Conclusion

As a conclusion, our approach is summarized with its contribution to current approaches such as single channel MSERs and SWT. Then, this thesis concludes by suggesting future work.

## 5.1 Summary of the Work

In order to achieve the high detection rate of texts, we have presented a novel approach for localizing text-lines in natural scene images; it employs MSERs with multiple channels such color spaces as RGB, HSI, YCbCr, and CIELAB to select basic character candidates. LRD is applied to intensity image and extracts MSERs of LRD-applied image to enhance the ability of detecting blurred texts. Enhanced canny edge detector and k-means clustering with $k = 2$ are used to overcome MSERs' shortcomings: the sensitivity to image blurs, clutter, and occlusion.

Based on obtained character candidates, they will be grouped into text candidate by applying geometric information such that texts usually appear in a linear form. Once text candidates are formed, stroke width, text region height, width and parallel edge features are used to get rid of non-text candidates.

Although our runtime was relatively higher than other methods, our experiments on ICDAR 2013 and Street View Text from Google Maps show that our approach has achieved a noticeable performance on blurred, small, and transparent texts and texts with highlights as well. Runtime issue will be handled in next work.

## 5.2 Future Work

A number of direct modification can be performed. Firstly, we could improve the execution time by implementing in C++, using GPGPU, and optimizing the code. Secondly, using various classifiers will help eliminating false positives, which will lead to a better performance [29]. This can be done by using [33]: Iqbal et al. compared classifiers that used to classify MSER-based texts in natural images. Finally, developing more detailed non-text filter will improve the performance.

There still exist several possible extensions for this work. First, how to detect curved text lines needs to be further investigated since some texts are written in non-horizontal lines. Second, probabilistic approach can be another good way to determine the existence of texts. Finally, designing word recognition algorithm to read text information from text regions will be the major work.

# Bibliography

[1] X. Chen and A. Yuille, “Detecting and reading text in natural scenes,” in Proc. IEEE Conf. CVPR, vol. 2. Washington, DC, USA, 2004, pp. 366-373.

[2] K. Kim, K. Jung, and J. Kim, “Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm,” IEEE Trans. Pattern Anal. Mach. Intel., vol. 25, no. 12, pp. 1631-1639, Dec. 2003.

[3] B. Epshtein, E. Ofek, and Y. Wexler, “Detecting text in natural scenes with stroke width transform,” in Proc. IEEE Conf. CVPR, San Francisco, CA, USA, 2010, pp.2963-2970.

[4] L. Neumann and J. Matas, “Text localization in real-world images using efficiently pruned exhaustive search,” in Proc. ICDAR, Beijing, China, 2011, pp. 687-691.

[5] L.Neumann and K. Matas, “Real-time scene text localization and recognition,” in Proc. IEEE Conf. CVPR, Providence, RI, USA, 2012, pp. 3538-3545.

[6] W. Huang, Z. Lin, J. Yang, J. Wang, “Text localization in natural images using stroke feature transform and text covariance descriptors,” IEEE Conf. ICCV, 2013.

[7] X. Yin, X. Yin, H, Hao, K. Iqbal, “Effective text localization in natural scene images with MSER, geometry-based grouping and adaboost,” ICPR, Tsukuba, Japan, 2012.

[8] H. Chen, S. Tasi, G. Schroth, D. Chen, R. Grzeszozuk, B. Girod, “Robust text detection in natural images with edge-enhanced maximally stable extremal regions,” IEEE Conf. ICIP, 2011.

[9] Y. F. Pan, X. Hou, C. Liu, “Text localization in natural scene images based on conditional random field,” In ICDAR, 2009, pp 6-10.

[10] X. Yin, X, Yin, K. Huang, H. Hao, “Robust text detection in natural scene images,” IEEE Trans. Pattern Anal. Mach. Intel., vol. 35, no. 5, May. 2014.

[11] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, Z. Zhang, “Scene text recognition using part-based tree-structured character detection,” in Proc. IEEE Conf. CVPR, 2013, pp 2961-2968.

[12] C.Lee, A. Bhardwaj, W. Di, V. Jagadeesh, R. Piramuthu, “Region-based discriminative feature pooling for scene text recognition,”, in Proc. IEEE Conf. CVPR, 2014.

[13] K. Wang, S. Belongie, “Word spotting in the wild,” in ECCV, 2010.

[14] K. Wang, B. Babenko, S. Belongie, “End-to-end scene text recognition,” in ICCV 2011, 2011.

[15] J. Zhang, R. Kasturi, “Character energy and link energy-based text extraction in scene images,” In ACCV 2010, vol. 2, pp. 832-844, Nov. 2010.

[16] H. Le, N. Toan, S. Park, G. Lee, “Text localization in natural scene images by mean-shift clustering and parallel edge feature,” in ICUMC 2011, Seoul, Korea, Feb. 2011.

[17] K. Subramanian, P. Natarajan, M. Decerbo, D. Castanon, “Character-stroke detection for text localization and extraction,” in ICDAR 2005, 2005.

[18] S. Lucas, “ICDAR 2005 text locating competition results,” in ICDAR 2005, 2005.

[19] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, “ICDAR 2003 robust reading competitions,” in ICDARD 2003, 2003.

[20] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. Bigorda, et al., “ICDAR 2013 robust reading competition,” in ICDAR 2013, 2013.

[21] T. Phan, P. Shivakumara, C. Tan, “Detecting text in the real world,” in Proc. ACM Int. Conf. MM, New York, USA, 2012, pp 765-768

[22] C. Wolf and J. Jolion, “Object count/area graphs for the evaluation of object detection and segmentation algorithms,” IJDAR, vol. 8, no. 4, 2006, pp. 280-296.

[23] J. Liang, D. Doermann, H. Li, “Camera-based analysis of text and documents: a survey,” International Journal on Document Analysis and Recognition, 2005, Vol. 7, pp. 82-200.

[24] K. Jung, K. Kim, A. K. Jain, “Text information extraction in images and video: a survey,” Pattern Recognition, p. 977-997, Vol. 5, 2004.

[25] J. Matas, O. Chum, M. Urban, T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions,” BMVC, 2002.

[26] http://research.microsoft.com/en-us/um/people/eyalofek/text_detection_database.zip

[27] H. Li, D. Doermann, and O. Kia, “Automatic text detection and tracking in digital video,” IEEE Trans. Image Process., vol. 9, no. 1, pp. 147-156, Jan. 2000.

[28] P. Shivakumara, Q. P. Trung, 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

[29] W. Shao, W. Yang, G. Xia, and G. Liu, “A hierarchical scheme of multiple feature fusion for high-resolution satellite scene categorization,” Sprigner-Verlag Berlin Heidelberg, 2013.

[30] K. Kita, and T. Wakahara, “Binarization of color characters in scene images using k-means clustering and support vector machines,” ICPR, 2010.

[31] A. Jain, X. Peng, X. Zhuang, P. Natarajn, and H. Cao, “Text detection and recognition in natural scenes and consumer vidoes,” ICASSP, 2014.

[32] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. Wu, and A. Ng, “Text detection and character recognition in scene images with unsupervised feature learning,” ICDAR, 2011.

[33] K. Iqabal, X. Yin, X. Yin, H. Ali, and H. Hao, “Classifier comparison for MSER-based text classification in scene images,” IJCNN, 2013.

[34] A. Misra, K. Alahari, C. Jawahar, “Top-down and bottom-up cues for scene text recognition,” CVPR, 2012.

[35] P. Gehler and S. Nowozin, “On feature combination for multiclass object classification,” ICCV, 2009.

[36] http://dag.cvc.uab.es/icdar2013competition/?ch=2&com=downloads

# 국 문 초 록

# 다양한 Feature의 조합을 통하여 자연 영상에서 글자를 찾는 방법

서울대학교 대학원
컴퓨터공학부
김재석

자연 영상에서 글자의 존재유무를 파악하고 위치를 찾아내는 일은 다양한 컴퓨터 비전과 컨텐츠 기반 영상을 분석하는데 매우 중요한 첫 단계이다. 본 논문에서는 Maximally Stable Extremal Region (MSER)을 이용하여 특징점들을 추출, 조합함으로써 자연 영상에 존재하는 글자를 찾는 임무를 수행한다. 다양한 색 공간, 흑백 영상, 색의 변화도, 그리고 루시앤리찰슨을 통한 영상에서 MSER을 추출함으로써 글자라고 예상되는 후보자들을 1차적으로 선출한다. 2차적으로 k-means 클러스터링을 이용하여 후보자들에 대한 정확도를 높였다. 이렇게 선출된 후보자들에 대해서는 글자들이 가지고 있는 특성들을 이용하여 글자가 아닐 것이라고 판단되는 후보자들을 제거하였다. 그 특성들로는 글자의 길이, 너비, 크기, 비율, 그리고 가장자리의 평행 정도 등이 있다.

본 논문에서 제안한 알고리즘은 ICDAR 2013과 Street View Text

(SVT)를 이용하여 검증을 하였다. 실험 결과, 기존 논문에서 찾는데 어려움을 겪었던 영상들에서 효과적으로 글자를 찾아낼 수 있었다.

**주요어 : 문자 검색, 특징점 조합, MSERs, k-means 클러스터링, 루시앤리찰슨 알고리즘**
**학 번 : 2013-20774**