Fast Scene Text Detection with RT-LoG Operator and CNN

Keywords: Scene text detection, keypoint grouping, RT-LoG, character pattern.

Abstract: Text detection in scene images is of particular importance for the computer-based applications. The text detection methods must be robust against variabilities and deformations of text entities. In addition, to be embedded into mobile devices, the methods have to be time efficient. In this paper, the keypoint grouping method is proposed by first applying the real-time Laplacian of Gaussian operator (RT-LoG) to detect keypoints. These keypoints will be grouped to produce the character patterns. These patterns will be filtered out by using a CNN model before aggregating into words. Performance evaluation is discussed on the ICDAR2017 RRC-MLT and the Challenge 4 of ICDAR2015 datasets. The results are given in terms of detection accuracy and time processing against different end-to-end systems in the literature. Our system performs as one of the strongest detection accuracy while supporting at nearly 15.6 frames per second to the HD resolution on a regular CPU architecture.

1 INTRODUCTION

Scene text detection is a key topic in the literature (Long et al., 2018) since several years ago due to its influence in the real-life. It can be listed by several useful real-world applications such as traffic sign recognition, blind assistance, augmented reality and so on. However, detecting and localizing scene text still remains a challenge due to degradations. This covers different aspects such as the texture, illumination changes, the differences in languages, scales of characters and the background / foreground transitions Figure 1. Robust methods must be designed against variabilities, deformations of text entities.

Moreover, another crucial problem is to adapt the methods to be time-efficient such as they can embedded into mobile devices. This involves an almost complete reformulation of the methods to make them real-time in order to respect the time constraint for detection (Neumann and Matas, 2015).

The real-time methods in the literature apply a two-stage strategy for localization then text classification. The localization determines the positions of candidate text elements in the image at a low complexity level. The main goal is then to process with a strong recall to not miss text elements. After that, the classification specifies which candidate is the text or not. It filters out the false alarms by using verification procedures. The two-stage strategy is opposite to the end-to-end strategy merging localization and classification (Long et al., 2018).

A core component of the two-stage strategy is the local operator. The local operator extracts candidate keypoints at the locations of text elements in the image. Different real-time operators have been proposed in the literature for scene text detection as the FASText operator (Busta et al., 2015), Canny Text Detector (Cho et al., 2016) and the Maximally Stable Extremal Regions (MSER) operator (Gomez and Karatzas, 2014).

(a) (b) (c)

Figure 1: Examples of text in natural scenes with specific degradations (a) blurring (b) different sizes of character (c) illumination changes.

In this paper, we propose a new system applying a two-stage strategy for the real-time detection of scene text. Compared with the other systems in the literature, our system applies in the first stage an RT-LoG operator. The RT-LoG operator is the real-time implementation of the Laplacian of Gaussian (LoG) operator (Fragoso et al., 2014). It has paid more attention in the community over the last recent years (Nguyen et al., 2019). It is competitive for time processing and can be adapted to text detection. It gives a slightly better performance for detection and provides meaningful scale-space and contrast information.

This paper gives several key contributions.

- We propose a new grouping method to embed the RT-LoG operator into a two-stage system.
- We highlight how the RT-LoG operator can support the full pipeline for scene text detection.

- We provide a performance evaluation where our overall system achieves one of the strongest detection accuracy of the literature, while requiring less than two orders of magnitude for the processing resources compared to our competitors.

The rest of paper is organized as follows. Section 2 presents our system. Then, performance evaluation of the system is discussed in section 3. At last, section 4 will summarize and give out some perspectives. For convenience, Table 1 provides the meaning of the main symbols used in the paper.

Table 1: The main symbols used in the paper.

Symbols	Meaning
x, y	Image coordinates
$g = f(x,y)$	f the image function returning the gray-level g at any location (x,y)
w	Stroke width parameter $w \in [w_{min}, w_{max}]$ with $\{w_{min}, ..., w_{max}\}$ the set of values
m	Scale parameter $m = (w_{max} - w_{min}) + 1$
$h(x,y)$	Map of the responses h to all the pixels in the image with $h \in [-1, 1]$
β	A thresholding parameter
$p = (x, y, g, h, w)$	A keypoint
S	A set of keypoints $\{p_1, .., p_n\}$
n	The number of keypoints
R	$\{R_1, ., R_i, .R_q\}$ A set of RoIs R_i is a given RoI
q	The number of regions

2 PROPOSAL APPROACH

The general architecture is presented in Figure 2. At the first stage, we employ the RT-LoG operator to detect the keypoints among the strokes composing the characters. Next, a dedicated algorithm is proposed to group the keypoints using the spatial / scale-space representation of the RT-LoG operator. The grouping method results in Regions of Interest (RoIs) constituted of character patterns (characters, parts of characters, connected characters). These RoIs are classified into text / non text regions by a CNN. Before the classification, these RoIs are normalized using the RT-LoG and geometric features. This normalization relaxes the classification with the CNN from invariant problems such as the contrast, scale and orientation. A method for text line grouping is applied in the final stage. We will detail in following subsections.

2.1 The RT-LoG Operator

To start, we process the image with the RT-LoG operator to detect the keypoints constituting the characters Figure 3. The RT-LoG operator is the real-time implementation of the Laplacian of Gaussian operator (LoG). Optimization of the operator is obtained with estimation of the LoG function and fast Gaussian convolution (Fragoso et al., 2014). Adaptation to stroke detection is given by the stroke model for scale-space representation (Nguyen et al., 2019; Liu et al., 2014).

The operator is controlled with a range of widths $w \in [w_{min}, w_{max}]$ of the strokes to detect. Within this range, $m = \{w_{max} - w_{min}\} + 1$ is not only the number of stroke widths to detect but also the parameter of the scale-space problem. An additional parameter β serves to threshold the operator responses and to tune the precision / recall of the detection.

The operator results in a set of n keypoints $S = (p_1, ...p_i, ...p_n)$ where a keypoint $p = (x, y, g, h, w)$ is given with (x,y) the spatial coordinates, g the gray level at the keypoint location, h the operator response and w the stroke width parameter. The operator response h is either positive or negative depending the background / foreground transition of the character. Figure 3 gives an example of keypoint detection with the corresponding response map $h(x,y)$ providing the h responses at every location (x,y).

The RT-LoG operator gives a slightly better performance for detection compared to the MSER operator. In addition, it is also competitive for time processing and provides meaningful scale-space and contrast information. This can drive the grouping method to obtain the character patterns and to guide the CNN. We will discuss these aspects in next subsections.

2.2 The Spatial / scale-space Grouping

The RT-LoG operator gives out, for each image, a set of keypoints belonging to the strokes composing the characters. These keypoints must be grouped together to constitute character patterns. The grouping of keypoints is well-known topic in the image processing and computer vision field (Dan et al., 2015). The main challenge here is to outline keypoints belonging to particular objects or RoIs in the image. Different strategies can be applied as the area-split with load imbalance, the clustering (e.g. K-means, Density-based spatial clustering), the grouping with machine learning (while characterizing the keypoints with descriptors) or the geometric consistency.

In this paper, we propose a new method for the grouping of keypoints. This method uses the spatial / scale-space representation of the RT-LoG operator and is then dedicated to the operator. Compared to the other methods in the literature, the grouping of keypoints is made consistent due to the meaningful information provided by the operator. Our method uses three main steps as detailed in Figure 4. We will detail

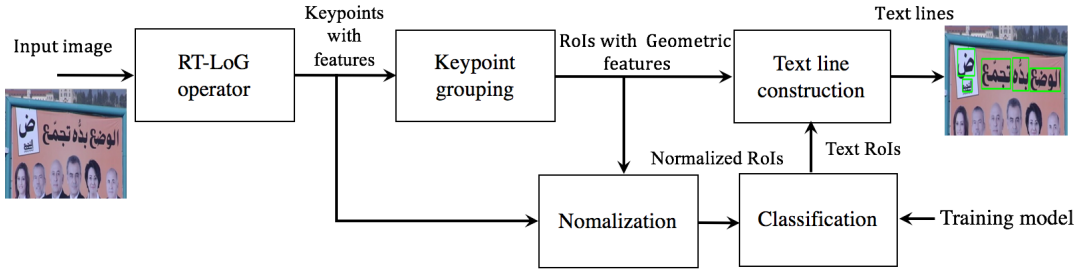
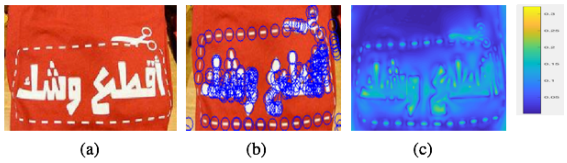


Figure 2: The detail of our approach.



(a) (b) (c)

Figure 3: (a) input image (b) detected keypoints with circle / radius at $w/2$ (c) the corresponding feature map $h(x,y)$.

these steps in next paragraphs.

Foreground / background partitioning (step 1): in this step, we need to cope with both conditions in scene text detection such as a light character in a dark background producing minus responses and vice-versa. A keypoint $p = (x, y, g, h, w)$ is given with a normalized response of the operator h. This response $h \in [-1, 1]$ results in two conditions of white $h \in [-1, 0[$ and dark $h \in]0, 1]$ foregrounds, respectively. The keypoints belonging to a same character pattern are supposed to have a similar response. This step splits the set of keypoints into two subsets based on their responses.

Scale-space partitioning and grouping (steps 2,3): the RT-LoG operator is scale-invariant and provides a stroke width parameters w for each of the detected keypoint. This is an useful information that can be used as a local threshold to group the keypoints. The RT-LoG operator applies a Non-maximum suppression (NMS) within 3×3 local neighborhood. This guarantees an overlapping between the keypoints, while setting the operator with $w_{min} \geq 2$.

The grouping performs for each of the keypoint p_i in the subsets obtained from the step 1. A keypoint p_i is grouped with a keypoint p_j if their Euclidean distance fits with Eq (1), illustrated in Figure 5. In that case, the label of the keypoint p_i is propagated to the keypoint p_j. Similar to the two-pass connected component algorithms (Cabaret and Lacassagne, 2017), the algorithm applies forward / backward requests and propagation to merge the labels between the keypoints.

$$\|p_i - p_j\|_2 \leq \frac{w_i + w_j}{2} \quad (1)$$

The requests could be time-consuming depending significantly on the used request algorithm and the number of keypoints. For optimization, the keypoints are indexed first with fast grouping method KD-tree based DBSCAN (Vijayalaksmi and Punithavalli, 2012) that ensures fast requests. A large numer of keypoints belonging to a character pattern has a close stroke width parameter Figure 6 (a). For optimization, we apply a two-stage strategy for grouping. The keypoints are partitioned first according to their stroke width parameter $w \in \{w_{min} \ldots . w_{max}\}$. The forward / backward requests are performed first within these scale-space partitions on small subsets of keypoints Figure 6 (b). This ensures a fast grouping for the main part of the keypoints. Then, the grouping is extended to the in-between scale-spaces Figure 6 (c).

Finally, we compute a set of geometric features from the keypoints belonging to a RoI to get the centroid, the area / perimeter, the bounding box and the orientation. These features will be used in the scene text detection pipeline to drive the image normalization and text line construction Figure 2.

2.3 Verification and Text Line Construction

The grouping algorithm results in the detection of RoIs. As the RT-LoG operator detects strokes, the detected RoIs could be either character patterns or background elements from the natural scene image. The RoIs must be classified to filter out the character patterns from background. This is the text verification problem that takes part in the scene text detection.

The traditional approach for scene text detection is to apply hand-crafted features with classification. Recently, CNN becomes prominent into the field where a main issue is to design end-to-end system (Long et al., 2018). When applying to text verification, the CNN classifies the RoIs of images to verify if a RoI belongs to a text part or not.

Several CNN models have been proposed in the literature for the text verification (Yang et al., 2015;

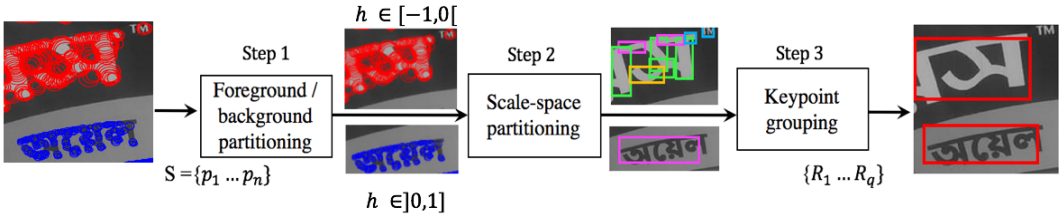


Figure 4: The detail of spatial / scale-space grouping method.

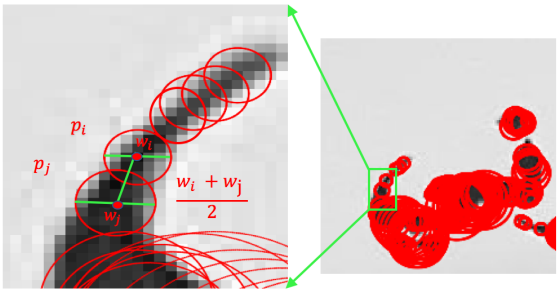


Figure 5: The grouping rule of two keypoints.

Table 2: CNN models for text verification.

Image size	(24 × 24) - (32 × 32)		
Layer No	Filter size	Filter No	Pooling
Conv1	(5 × 5)-(8 × 8)	x20 - x96	(2 × 2)-(5 × 5)
Conv2	(2 × 2)-(5 × 5)	x50 - x256	(2 × 2)

Ray et al., 2016; Turki et al., 2017; Wang et al., 2018; Zhang et al., 2015). Table 2 gives a summation. The fundamental model is to process with two convolutional layers while using a ReLU function for the non-linearity and a max pooling for optimization. A fully-connected layer (FC) is used in the final stage for classification set with a softmax function. Some works have investigated deeper models using up to four convolutional layers such as (Zhang et al., 2015).

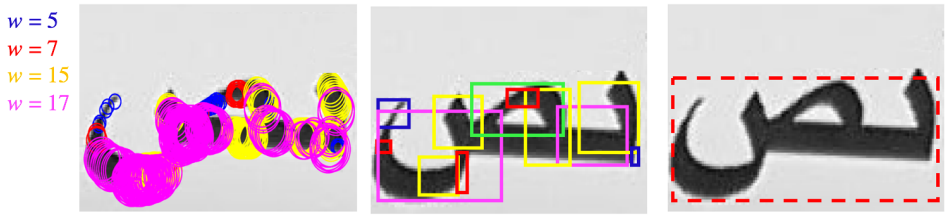
In the final step, after classification with normalization, the close text regions must be grouped to get the text lines. Similar to (Cho et al., 2016), we use the method of minimum-area encasing rectangle to provide consistent bounding boxes.

For the sake of performance evaluation, we customize our model inspired from (Turki et al., 2017; Wang et al., 2018). It is illustrated in Figure 7 with two convolutional layers and one FC layer. For each convolutional layer, ReLu activation and average pooling layer are followed.

A core problem for text verification is the variability character patterns. These patterns are not normalized and suffered from distortions Figure 8 (a) - (d). Another problem comes from the look-like cursive characters that result in word patterns after detection and grouping Figure 8 (e). The normalization of these patterns to 32×32 images introduces other distortions as the down-scaling and the modification of the aspect ratio. This tends to burden the learning period of the CNN training for text verification.

To solve these problems, we apply an image normalization before to classify the character patterns with the CNN. We use the RT-LoG and geometric features to drive the normalization as they provide meaningful information about the character patterns. Our process is detailed here.

- Skew correction: we use the geometric features to correct the skew of characters with image rotation Figure 8 (a).
- Background / foreground normalization: the keypoints belonging to a character pattern provide a common operator response h. This response is whether positive or negative depending on the background / foreground transition. We look for all the character patterns having a positive response $h > 0$ and invert the image for normalization Figure 8 (b).
- Contrast normalization: the character patterns appear with different background / foreground amplitudes. This is a problem of contrast normalization that can be solved from the operator response h. Indeed, the operator response $|h| \in [-1, 1]$, where $|h| = 1$ is obtained from bilevel black and white image having a maximum amplitude. We use the response h and the g values to get a lookup table to normalize the contrast Figure 8 (c).
- Scale normalization: the normalization of character patterns to 32×32 image distorts the aspect ratio Figure 8 (d). We use the average stroke width $\overline{w}$ of character patterns as a scale estimator. This estimator serves to compute a scale ratio $\overline{w_0}/\overline{w}$ with $\overline{w_0}$ an offline parameter obtained from the training database. The parameter $\overline{w_0}$ is fixed in order to embed the character patterns into a 32×32 image, as an average. If not, we split the character patterns into 32×32 patches while respecting an overlapping threshold Figure 8 (e).



(a) (b) (c)

Figure 6: (a) Keypoints within the scale-space partitions (b) grouping within the scale-space partitions (c) grouping within in-between scale-spaces.

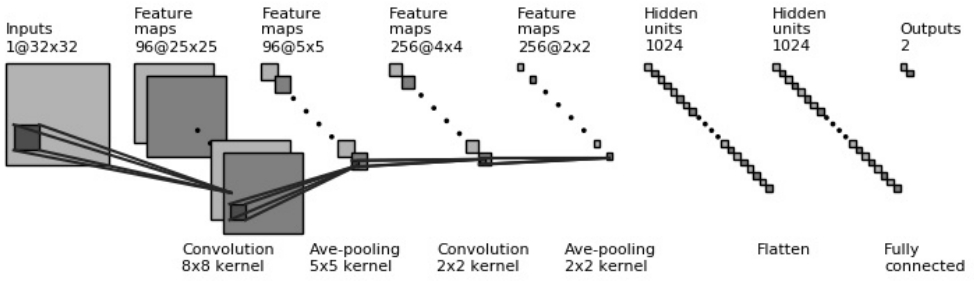


Figure 7: CNN model used for RoIs classification.

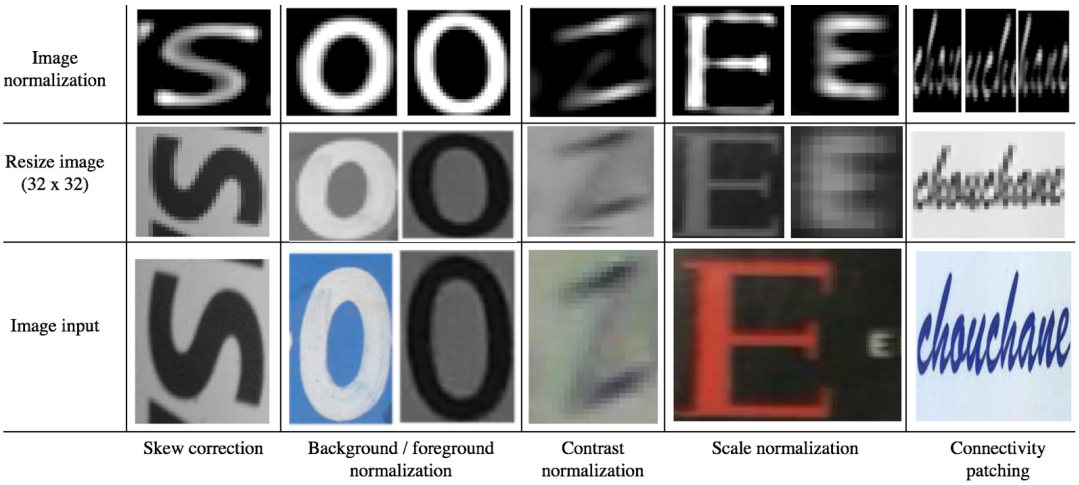


Figure 8: Image normalization.

Figure 9: Images from the ICDAR2017 RRC-MLT dataset (Nayef et al., 2017).

3 PERFORMANCE EVALUATION

In this section, we present the performance evaluation of our system. Section 3.1 introduces the used datasets. Section 3.2 details the characterization metrics. Results are discussed in terms of detection accuracy and time processing in sections 3.3 and 3.4.

3.1 Datasets

Several public datasets have been proposed over the years for performance evaluation of text detection methods. The ICDAR2017 and ICDAR2019 RRC-MLT are the two up-to-date datasets in literature (Nayef et al., 2017; Nayef et al., 2019). The ICDAR2019 RRC-MLT is a recent dataset and a slight update of the ICDAR2017 RRC-MLT[1]. For our performance evaluation, we have selected the ICDAR2017 RRC-MLT dataset where more comparative results are available in the literature. It includes 7200 training images, 1800 validation images and 9000 test images. The images are given at different resolutions (VGA, HD, Full-HD, Quad-HD, 4K). This dataset has a particular focus on the multi-lingual scene text detection in 9 languages and offers a deep challenge for scalability. Figure 9 shows some visual examples of images.

For the processing time, it is more common in the literature to use the Challenge 4 of ICDAR2015 dataset (Karatzas and Gomez-Bigorda, 2015). This dataset contains 1000 training images and 500 test images at HD resolution (1280×720).

3.2 Characterization Metrics

For the characterization metrics, we have followed the recommendations of the international contest (Nayef et al., 2017). The characterization is achieved at two levels while applying the Intersection over Union (IoU) criterion and computing the F-measure. The output of the text detection system is provided with bounding boxes. A detection (i.e. a true positive) is obtained if a detected bounding box has more than 50% overlap (the IoU criterion) with a bounding box in the groundtruth. The unmatched boxes in the detection and the groundtruth are false positives and negatives, respectively. The detection cases serve to compute the regular metrics precision (P), recall (R), F-measure[2]. Let's note that some degraded texts in the dataset are marked as "don't care" boxes and ignored in the evaluation process.

[1]The ICDAR2019 RRC-MLT dataset includes 1K images more on the existing set of 9K images.

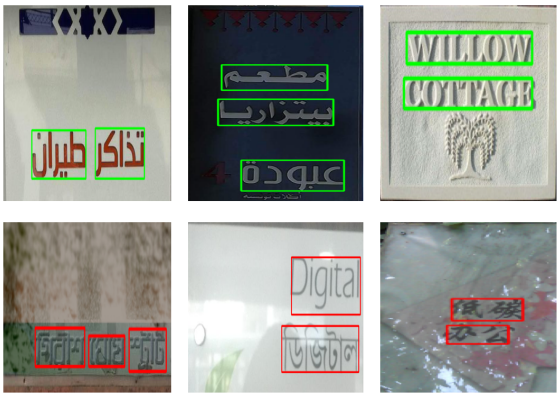


Figure 10: Text detection samples with (green) the true detections and (red) the missed-cases.

3.3 Scene Text Detection

Table 3 shows our evaluation in comparison with the state-of-the-art. For clarification, we have stood on the top 10 of competitive systems of the literature.

As highlighted in Table 3, our system appears in the top 6 for the F-measure score. Furthermore, our system achieves the strongest recall score of the literature. This is ensured by the use of the RT-LoG operator and the dedicated grouping method, which allows a near complete detection of text elements. Some visual examples of true and missed detections are shown in Figure 10. The operator fails to detect text in images with very low contrast. This can be explained by the real-time implementation of the LoG operator, that is not contrast invariant (Nguyen et al., 2019).

3.4 Processing Time

We characterize and compare in this section the processing time of our system against other systems in the literature. We have computed first the processing time of each step of our method using a single thread / core implementation Table 4. Our implementation

[2]We are not detailing these aspects here and report to (Nayef et al., 2017).

Table 3: Comparison of methods
(P) precision (R) recall and (F) F-Measure on ICDAR2017 RRC-MLT.

Rank	Methods	P(%)	R(%)	F(%)
1	PMTD (Liu et al., 2019)	**85.15**	72.77	**78.48**
2	FCN-MOML (He et al., 2018)	82.66	72.53	77.26
3	R-CNN-PAN (Huang et al., 2019)	80	69.8	74.3
4	LOMO MS (Zhang et al., 2019)	80.2	67.2	73.1
5	MOSTD (Lyu et al., 2018b)	74.3	70.6	72.4
6	Proposed method	62.5	**82.7**	71.2
7	Fots (Liu et al., 2018)	81.86	62.30	70.75
8	AF-RPN (Zhong et al., 2018)	75	66	70
9	Attention Model (Wang et al., 2019)	72	63.5	67.48
10	SCUT DLVClab1 (Nayef et al., 2017)	80.3	54.5	65

Table 4: Average processing time in milliseconds (ms) / amounts of pixels, keypoints and RoIs of each step of the proposed method.

Method \ Types	HD	Data workflow
RT-LoG	320 ms	1.2 Mpixel
Grouping	200 ms	5.2 Kkeypoints
Verification	336 ms	90 RoIs

Table 5: Frame rate per second (FPS) among methods on the Challenge 4 of ICDAR2015 dataset.

Methods \ Processing types	FPS	Architecture	Performances TFLOPS
FOTs-RT(Liu et al., 2018)	**22.6**	TITAN-Xp GPU	12.15
Ours	15.6	CPU 2.2 GHz	**0.032**
SSTD (He et al., 2017)	7.7	TITAN X GPUs	6.691
EAST (Zhou et al., 2017)	6.52	TITAN-Xp GPU	12.15
MTS(Lyu et al., 2018a)	4.8	Titan Xp GPU	12.15
MOSTD(Lyu et al., 2018b)	3.6	Tesla K40m GPU	5.046

is done with the C++ language and tested on the Mas-OS and an Intel(R) Core(TM) i7- 4770HQ CPU, 2.2 GHz having a near 32 GFLOPS SP performance. The processing times have been obtained on th HD resolution while using the ICDAR2015 dataset.

We provide in addition in Table 4 information about the data workflow in the pipeline (the amounts of pixels, keypoints and RoIs). As highlighted the RT-LoG operator results in a large reduction of the data to process. Despite the different of amount of data, the processing time is close between the RT-LoG operator and the rest of pipeline (grouping with CNN). This can be explained by the real-time implementation of the operator that ensures a strong optimization of the spatial convolution.

We have evaluated, in a second step, the Frame rate per second (FPS) of our system on the ICDAR2015 dataset Table 5. This evaluation has been done with a full parallelism support on the CPU while applying multi-core / threading. We provide in addition the FPS of the top systems in literature. For a fair comparison, Table 5 addresses the test architectures of different systems (either GPU or CPU) with their relative performances[3].

As emphasized in Table 5, our system has a second highest FPS rate while processing with a difference of two orders of magnitude in term of processing resources. All the the top systems in the literature perform with end-to-end CNN model requiring a GPU architecture.

[3]https://www.techpowerup.com/

4 CONCLUSIONS AND PERSPECTIVES

This paper presents a new two-stage system for scene text detection. It applies in the first stage the RT-LoG operator. This operator supports the full pipeline for scene text detection. A dedicated algorithm is applied to group the keypoints into RoIs using the RT-LoG features. The RoIs are then classified into text / non text regions by a CNN. Before the classification, the RoIs are normalized with the RT-LoG features. This normalization relaxes the classification from invariant problems. The proposed system is in the top 6 for the F-measure score and achieves the strongest recall of the literature. It obtains the second highest FPS rate while processing with a difference of two orders of magnitude in term of processing resources.

As a main perspective, the precision rate of the system can be consolidated. This can be obtained by making the RT-LoG operator contrast-invariant, to deal with the missed detection cases. Deeper CNN models could be applied to make more robust at the text verification stage. At last, acceleration of the RT-LoG operator could be obtained by taking advantages of the Gaussian kernel distribution and decomposition with box filtering.

REFERENCES

Busta, M., Neumann, L., and Matas, J. (2015). Fastext: Efficient unconstrained scene text detector. In *International Conference on Computer Vision (ICCV)*, pages 1206–1214.

Cabaret, L. and Lacassagne, L. (2017). Distanceless label propagation: an efficient direct connected component labeling algorithm for gpus. In *International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6.

Cho, H., Sung, M., and Jun, B. (2016). Canny text detector: Fast and robust scene text localization algorithm. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3566–3573.

Dan, G., Khan, M., and Fodor, V. (2015). Characterization of surf and brisk interest point distribution for distributed feature extraction in visual sensor networks. volume 17, pages 591–602.

Fragoso, V., Srivastava, G., Nagar, A., and Li, Z. (2014). Cascade of box (cabox) filters for optimal scale space approximation. In *International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 126–131.

Gomez, L. and Karatzas, D. (2014). Mser-based real-time text detection and tracking. In *International Conference on Pattern Recognition (ICPR)*, pages 3110–3115.

He, P., Huang, W., He, T., and Zhu, Q. (2017). Single shot text detector with regional attention. In *International Conference on Computer Vision (ICCV)*, pages 3047–3055.

He, W., Zhang, X., Yin, F., and Liu, C. (2018). Multi-oriented and multi-lingual scene text detection with direct regression. volume 27, pages 5406–5419.

Huang, Z., Zhong, Z., Sun, L., and Huo, Q. (2019). Mask r-cnn with pyramid attention network for scene text detection. In *Conference on Applications of Computer Vision (WACV)*, pages 764–772.

Karatzas, D. and Gomez-Bigorda, L. (2015). Icdar 2015 competition on robust reading. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160.

Liu, J., Liu, X., Sheng, J., and Liang, D. (2019). Pyramid mask text detector. *arXiv preprint:1903.11800*.

Liu, X., Liang, D., S.Yan, and Chen, D. (2018). Fots: Fast oriented text spotting with a unified network. In *Conference on computer vision and pattern recognition (CVPR)*, pages 5676–5685.

Liu, Y., Zhang, D., Zhang, Y., and Lin, S. (2014). Real-time scene text detection based on stroke model. In *International Conference on Pattern Recognition (CVPR*, pages 3116–3120.

Long, S., He, X., and Ya, C. (2018). Scene text detection and recognition: The deep learning era. *arXiv:1811.04256*.

Lyu, P., Liao, M., .Yao, C., and Wu, W. (2018a). Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *The European Conference on Computer Vision (ECCV)*.

Lyu, P., Yao, C., and W. Wu, S. Y. (2018b). Multi-oriented scene text detection via corner localization and region segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7553–7563.

Nayef, N., Patel, Y., Busta, M., and .Chowdhury, P. (2019). Icdar2019 robust reading challenge on multi-lingual scene text detection and recognition–rrc-mlt-2019. *arXiv preprint:1907.00945*.

Nayef, N., Yin, F., Bizid, I., and Choi, H. (2017). Icdar2017 robust reading challenge on multi-lingual scene text detection and script identification-rrc-mlt. In *International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 1454–1459.

Neumann, L. and Matas, J. (2015). Real-time lexicon-free scene text localization and recognition. volume 38, pages 1872–1885.

Nguyen, D., Delalandre, M., Conte, D., and Pham, T. (2019). Performance evaluation of real-time and scale-invariant log operators for text detection. In *International Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, pages 344–353.

Ray, A., Shah, A., and Chaudhury, S. (2016). Recognition based text localization from natural scene images. In *International Conference on Pattern Recognition (ICPR)*, pages 1177–1182.

Turki, H., Halima, M., and Alimi, A. (2017). Text detection based on mser and cnn features. In *International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 949–954.

Vijayalaksmi, S. and Punithavalli, M. (2012). A fast approach to clustering datasets using dbscan and pruning algorithms. volume 60.

Wang, H., Rong, X., and Tian, Y. (2019). Towards accurate instance-level text spotting with guided attention. In *International Conference on Multimedia and Expo (ICME)*, pages 994–999.

Wang, Y., Shi, C., Xiao, B., Wang, C., and Qi, C. (2018). Crf based text detection for natural scene images using convolutional neural network and context information. *Neurocomputing*, 295:46–58.

Yang, H., Wang, C., Che, X., Luo, S., and Meinel, C. (2015). An improved system for real-time scene text recognition. In *International Conference on Multimedia Retrieval (ACM)*, pages 657–660.

Zhang, C., Liang, B., Huang, Z., En, M., and Han, J. (2019). Look more than once: An accurate detector for text of arbitrary shapes. *arXiv preprint:1904.06535*.

Zhang, C., Yao, C., Shi, B., and Bai, X. (2015). Automatic discrimination of text and non-text natural images. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 886–890.

Zhong, Z., Sun, L., and Huo, Q. (2018). An anchor-free region proposal network for faster r-cnn based text detection approaches. *arXiv preprint:1804.09003*.

Zhou, X., Yao, C., Wen, H., and Wang, Y. (2017). East: an efficient and accurate scene text detector. In *International conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5551–5560.