J Real-Time Image Proc manuscript No.
(will be inserted by the editor)

Fast RT-LoG operator for scene text detection

Cong Nguyen Dinh · **Mathieu Delalandre** · **Donatello Conte** · **The Anh Pham**

Abstract This paper proposes a new real-time Laplacian of Gaussian (RT-LoG) operator for scene text detection. This method takes advantage of the Gaussian kernel distribution in the spatial/scale-space domains and kernel decomposition with the box filtering method. Two levels of optimization are given. The first level of optimization within the spatial domain is obtained by box mutualization. The second level of optimization within the spatial/scale-space domains is performed using a mixed method for box selection. The proposed RT-LoG operator is evaluated on the ICDAR2017 RRC-MLT dataset in terms of robustness and time processing. The results are compared with the state-of-the-art real-time operators for scene text detection. The proposed operator appears as the top performance with the best trade-off between robustness and time processing. The proposed operator can support approximately 30 frames per second (FPS) up to the Quad-HD resolution on a regular CPU architecture with a low-level latency. In addition, the proposed operator can support the full pipeline for scene text detection. Our system is competitive with the top accurate systems of the literature while processing with a difference of two orders of magnitude in term of processing resources.

Keywords Scene text detection · RT-LoG · stroke model · box filter · box selection · real-time · predictability

Cong Nguyen Dinh
Hong Duc University, Thanh Hoa City, Vietnam
E-mail: nguyendinhcong@hdu.edu.vn

Mathieu Delalandre
Tours University, Tours City, France
E-mail: mathieu.delalandre@univ-tours.fr

Donatello Conte
Tours University, Tours City, France
E-mail: donatello.conte@univ-tours.fr

The Anh Pham
Hong Duc University, Thanh Hoa City, Vietnam
E-mail: phamtheanh@hdu.edu.vn

1 Introduction

Scene text detection in natural images is an active topic in the image processing and pattern recognition fields. Recent contributions are discussed in surveys [1, 2], and the international contest dedicated to this topic is detailed in [3]. The fundamental and earliest problem investigated in the literature is to make text detection methods robust against variabilities and deformations of text entities in images, which covers different aspects, such as texture and illumination changes, the different scales of characters, the background /foreground transitions, as shown in Fig. 1.

However, another core problem is to adapt the methods to be time-efficient and real-time, which involves an almost complete reformulation of the methods [4]. The design of real-time methods and systems is a well-known topic in the literature [5]. There are two points, that distinguish the real-time systems from another kind of systems, that are timeliness and predictability.

Predictability is related to the design of methods with sharp upper and lower bounds on the execution times. The execution times of methods are guaranteed in order to prevent from the trashing cases and missed deadlines within the system. To be time-efficient, most of the image processing algorithms apply a pruning strategy that is not suitable with the predictability [6]. Hence, predictability requires to create specific methods with a trade-off between the optimization and the variation of the execution times.

The timeliness property looks for the respect of a deadline which describes the maximum time among the execution times. The results of the system have to be accurate not only in their values but also in the time domain. For the camera-based applications, the deadline depends on the kind of applications and expected frame rate per second (FPS), the image resolution and coding, the hardware architecture and the complexity of the given algorithm.

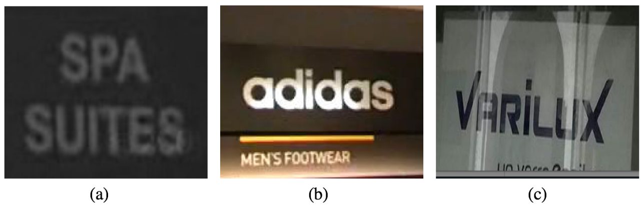


Fig. 1: Examples of text in natural scenes with specific degradations (a) blurring (b) different sizes of character (c) illumination changes.

To be time-efficient, a two-stage strategy is applied in the literature: localization followed by text verification [1, 4]. Localization determines the positions of the candidate text elements in the image at a low complexity level. The main goal is to process with strong recall to not miss text elements. Then, text verification specifies which candidate is text or not. It filters out the false positives using verification procedures and/or machine learning methods.

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 an image. Different time-efficient operators have been proposed in the literature for scene text detection, such as the FASText [7], the Canny Text [8], the Stroke Width Transform (SWT) [9], and the BSV [10]. However, most of time-efficient methods are dominated by the Maximally Stable Extremal Regions (MSER) operator [11, 12]. All these operators can fit with the time-efficient and real-time requirements. However, they are sensitive to noise and/or or scale-dependent. They are little accurate as an average.

The recent trend in the literature is to process with end-to-end deep architectures and systems [2]. However, these systems are time-consuming approach that is minimally compatible with a low-level energy consumption requirement on the low-cost hardware architectures [13]. In addition, GPU-based processing is not compatible with real-time constraints due to the highly parallel computation and memory transfer [14]. The GPU/CNN based systems are more dedicated to offline recognition processing on a workstation and/or a server [15, 16].

Thus, finding an accurate method and system, fitting with the time-efficient and real-time constraints, is still an open problem in the literature. We address this issue in this paper. To design an accurate, time-efficient and real-time system we have considered a two-stage strategy while designing a new real-time operator. Recently, the Laplacian of Gaussian (LoG) operator with a time-efficient and predictable implementation has received attention [17]. We call this operator RT-LoG for short. Adaptation of this operator to scene text detection, in order to make it scale-invariant, has been investigated in [18, 19]. In this paper, a novel RT-LoG operator is proposed. Our contributions are follows.

- A state-of-the-art RT-LoG operator for scene text detection is discussed. Fast spatial filtering is obtained with a difference of Gaussian (DoG) function approximation and box filtering for Gaussian convolution. Following the way, an estimator cascade methodology for optimization is deployed. No pruning is applied and the processing is achieved with sharp upper and lower bounds on the execution times for predictability. Adaptation to text detection is given by a scale-space representation with the stroke model.
- A novel RT-LoG operator is proposed. This operator applies a two-step process for box selection within the spatial and spatial/scale-space domains. The overall approach results in a main optimization of the RT-LoG operator for scene text detection.
- A performance evaluation is performed on the ICDAR 2017 RRC -MLT dataset in terms of robustness and time processing. The results are compared with the state-of-the-art time-efficient and real-time operators for scene text detection. The proposed operator appears to have the performance with the best trade-off between accuracy and speed.
- Additional experiments are performed to evaluate the frame per second (FPS) rates supported by the operator using a multithread/multicore support. These experiments are performed on a regular CPU architecture with standard resolution videos. The proposed RT-LoG operator is able to process at nearly 30 FPS up to the Quad-HD resolution on a regular CPU architecture with a low-level latency.
- Our operator is embedded into a full pipeline for scene text detection. Compared to other operators in the literature, the proposed RT-LoG operator provides a meaningful scale-space and contrast information that can drive the full pipeline for detection. The system performs as one of the strongest detection accuracy of the literature with the support of the operator. It requires in addition less than two orders of magnitude for the processing resources compared to the competitors of the literature.

The remainder of this paper is organized as follows. In section 2, the state-of-the-art is presented. Section 3 details our method. The performance evaluation is discussed in section 4. Finally, conclusions are presented and some perspectives are proposed in section 5. For convenience, Table 1 gives the meaning of the main symbols used in this paper.

2 State-of-the-art

In this section, we briefly cover the family of the LoG operator. Section 2.1 introduces the mathematical formulation

Table 1: The main symbols used in the paper.

Symbols	Meaning
	Continuous/discrete domain
$f(x,y)$	An image function/raster
$g(x,y)$	A Gaussian function/operator
Π, $\lVert\Pi\rVert$	Π a step function / box filter , $\lVert\Pi\rVert$ the radius of the box
w	Stroke width parameter with $w \in [w_{min}, w_{max}]$
a	Signal amplitude
α, β, γ	Some parameters
$\otimes$	The global convolution product
	Continuous domain
σ	The scale of the Gaussian and LoG functions
g_{xx} (either y)	Second partial derivative of the g function
∇^2	Laplacian
r	Radius of a region
k	Parameter to control approximation between LoG, DoG
$\widetilde{\sigma}, k\widetilde{\sigma}$	The scales of the DoG operator
σ_s	The optimum scale of the LoG operator for the stroke model
h, h_s, h_e	Response with the stroke model, h_s / h_e the stroke / edge optimums
ϖ	The stroke model function $\sigma_s = \varpi(w)$
$\Phi(k)$	A function to control the slope in the stroke model
$f(x)$	The function for spatial / scale-space selection
	Discrete domain
O	Complexity
N	Size of the image
ω^2	Size of operator (width $\times$ height)
i, j, l	i, j matrix/vector indexes, l predefined indexes
$[\sigma_0, \ldots, \sigma_m]$	A filter bank including $(m+1)$ discrete filters
$\widehat{\ }$	Estimator
n	Number of box filters
λ, δ	Weighting parameters
A, B, C	The matrices for the global products
P	The size of matrices
B_0, C_0, B_1, C_1	The subsets of the A, B, C matrices
$[L_{\sigma_0}, \ldots, L_{\sigma_m}]$	The global convolutions obtaining on $[\sigma_0, \ldots, \sigma_m]$
$\circ$	Element-wise multiplication
s^0, s^1, s^2, s^3	Vertices of an arbitrary rectangle within integral image
$\overline{MSE}$	Average of mean square error

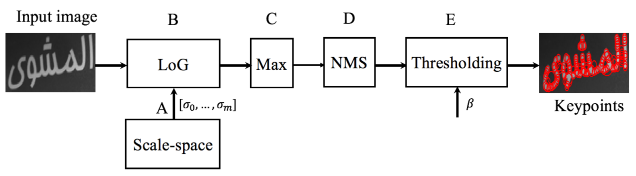


Fig. 2: The pipeline for LoG operator.

of the operator and presents method for adaptation with text elements in the scene of a text image. Sections 2.2 and 2.3 provide two levels of optimization based on spatial/scale-space domains, respectively.

2.1 Introduction

The LoG operator is defined as the Laplacian of Gaussian function and is derived from the Gaussian function. The Gaussian function is given in Eq. (1) in a multivariate form with a vectorial notation.

$$g(p|\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{\alpha}{2}}\sqrt{|\Sigma|}} e^{-\frac{1}{2}(p-\mu)^T \Sigma^{-1}(p-\mu)} \tag{1}$$

In the two-dimensional case, p is a point, and μ is the mean. Σ is the diagonal covariance matrix with Σ^{-1} the inverse and $|\Sigma|$ is the determinant, where the σ_x and σ_y parameters inside Σ are the standard deviations for dimensions x and y, respectively. $\alpha = 2$ is a weighting parameter, considering $\sigma_x = \sigma_y = \sigma$, $\mu = 0$ and a scalar notation, the Gaussian function Eq. (1) becomes Eq. (2),

$$g(x,y|\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2}$$

where x and y are the spatial coordinates.

The LoG operator is a compound operator resulting from the Laplacian ∇^2 of $g(x,y|\sigma)$, as noted Eq. (3),

$$\begin{aligned} \nabla^2 g(x,y|\sigma) &= g_{xx}(x,y|\sigma) + g_{yy}(x,y|\sigma) \\ &= \frac{1}{2\pi\sigma^4}\left(\frac{x^2+y^2}{\sigma^2} - 2\right) e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} \end{aligned} \tag{3}$$

where $g_{xx}(x,y|\sigma)$ and $g_{yy}(x,y|\sigma)$ are the second order derivative of Gaussian functions based on the dimensions x, y.

The LoG-filtered image $h(x,y)$ Eq. (4) is obtained from the global convolution $\otimes$ between the initial image $f(x,y)$ and the LoG operator $\nabla^2 g(x,y|\sigma)$.

$$h(x,y) = \nabla^2 g(x,y|\sigma) \otimes f(x,y) \tag{4}$$

As shown in Fig. 2, LoG filtering must be embedded in a full pipeline to design an end-to-end operator. The response of the LoG filtering Eq. (3) is highly dependent on the σ parameter. To deal with this problem, the standard approach is to handle the operator in the scale-space domain with a filter bank $[\sigma_0, ..., \sigma_m]$ with $\sigma_0 ... \sigma_m$ as the scale parameters. The maximum response is then selected from the LoG-filtering images at the different scales (step C). After that, the final keypoints are obtained with a non-maximum suppression (NMS) and a thresholding step (steps D, E). The final keypoints are expressed as the centroid coordinates and a radius with a normal value $r = \sqrt{2}\sigma$ for a circular blob.

The operator shown Fig. 2 can be made real-time with optimization in the spatial and scale-space domains (steps A, B of Fig. 2). This process requires specific approaches for the fast spatial filtering and an efficient scale-space representation. We detail these issues in the following sections.

2.2 Fast LoG Filtering

The strategy for fast LoG filtering is to apply an estimator cascade methodology, such as LoG $\approx$ DoG $\approx \widehat{DoG}$ where DoG and $\widehat{DoG}$ are two operators to approximate the LoG operator. LoG filtering is not separable; it is therefore time-consuming at complexity $O(N\omega^2)$ with N and ω^2, which are the image and mask sizes, respectively. The first level of approximation is acquired via reformulation of the LoG function into the DoG function.

The DoG function is inspired from the heat equation [20]. In Eq. (5), normalization of the LoG function Eq. (3) with a scale parameter σ gives the derivative of the Gaussian function in the scale-space domain. The left term of Eq. (5) can be reformulated as a local derivative of Gaussian function $\frac{\partial g(x,y|\sigma)}{\partial \sigma}$ with k as a parameter and a step offset $(k-1)\sigma$. The approximation of the derivative in this equation improves as $(k-1)\sigma$ goes to 0 when k comes to 1.

$$\begin{aligned}\sigma\nabla^2 g(x,y|\sigma) &= \frac{\partial g(x,y|\sigma)}{\partial \sigma} \\ &\approx \frac{g(x,y|k\sigma)-g(x,y|\sigma)}{(k-1)\sigma}\end{aligned} \tag{5}$$

With the reformulation of Eq. (5), the LoG function can be approximated by means of the DoG function, as noted Eq. (6),

$$\begin{aligned}&g(x,y|k\sigma)-g(x,y|\sigma) \approx (k-1)\sigma^2\nabla^2 g(x,y|\sigma) \\ &= \frac{1}{2\pi}\left(\frac{1}{(k\sigma)^2}e^{\left(-\frac{x^2+y^2}{2(k\sigma)^2}\right)} - \frac{1}{\sigma^2}e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}\right)\end{aligned} \tag{6}$$

with a normalization factor $(k-1)\sigma^2$.

The σ parameters within the LoG and DoG functions in Eq. (6) are equal when k goes to 1. Otherwise, if $k>1$, a slightly gap appears between the parameters. We obtain $\sigma \approx \tilde{\sigma}$, where $\tilde{\sigma}$ is the parameter used in the DoG function. The relation between $\sigma, \tilde{\sigma}$ is given in Eq. (7). For the sake of notation, in the remainder of the paper, we define the scale parameters of the DoG operator as $k\tilde{\sigma}$ and $\tilde{\sigma}$.

$$\sigma^2 = 2\left(\frac{k^2}{k^2-1}\ln k\right)\tilde{\sigma}^2 \tag{7}$$

The DoG function is computed with two Gaussian filters. With convolution, Gaussian filtering can be implemented in a separable manner at a complexity $O(N\omega)$ with N and ω^2 as the image and mask sizes, respectively. When ω is large, it is still a time-consuming task. Several methods have been proposed in the literature to accelerate Gaussian filtering and make it independent of the filter size at a complexity $O(N)$. These methods attempt to improve computational efficiency in the expense of accuracy. These methods referred to as fast Gaussian filtering methods and support the design of a difference of Gaussian estimator $\widehat{DoG}$.

Table 2: Time optimization and accuracy of fast Gaussian methods: (SII) Stacked Integral Image, (VYV) Vliet Young Verbeek, (KII) Kernel Integral Image, (TCF) Truncated Cosine Functions, (+++) best case, (+) medium case.

Category	Method	Time optimization	Accuracy
Box filter	Box	++	+++
	SII	+++	++
	KII	+	+
Recursive filter	Deriche	++	++
	TCF	+++	+++
	VYV	+	++

A survey with a performance evaluation can be found in [21, 22]. Two main categories are investigated including the box and recursive-based filters. Selection of a suitable method depends on the application use-case, which is supposed to be solved in terms of a good trade-off between speed and accuracy. Table 2 provides a global comparison of the methods.

In this paper, we prefer to employ the box filtering method Eq. (10), which is considered one of the most accurate methods for stroke detection [19] and is competitive with recursive filters [21, 22]. The box filtering method sums up the averaging filtering to approximate a Gaussian filter $\widehat{g}(x,y|\sigma)$, as noted in Eq. (8) with a desired standard deviation,

$$\widehat{g}(x,y|\sigma) = \sum_{i=0}^{n} \lambda_i \Pi_i(x,y) \tag{8}$$

where $\Pi_i(x,y)$ is a box filter function with a predefined size and a value 1 if (x,y) are located inside the box or 0 otherwise. The λ_i parameters weight the box filters $\Pi_i(x,y)$. $n+1$ is the number of box filters.

From Eq. (8), it is possible to approximate the DoG operator by the $\widehat{DoG}$ operator in Eq. (9) with two sets of box filter function. As the k parameter in Eq. (5) is supposed to be low[1], a similar number of filters can be applied to estimate the two Gaussian kernels,

$$\begin{aligned}\widehat{DoG} &= \widehat{g}(x,y|k\tilde{\sigma}) - \widehat{g}(x,y|\tilde{\sigma}) \\ &= \sum_{i=0}^{n}\lambda_i\Pi_i(x,y) - \sum_{j=0}^{n}\lambda_j\Pi_j(x,y)\end{aligned} \tag{9}$$

where λ_i, λ_j are the weighting parameters, $n+1$ is the number of boxes. $k\tilde{\sigma}, \tilde{\sigma}$ are the scale parameters.

The DoG-filtered image is approximately achieved by global convolution Eq. (10) between the input image $f(x,y)$ and the $\widehat{DoG}$ operator,

$$\begin{aligned}&(\widehat{g}(x,y|k\tilde{\sigma}) - \widehat{g}(x,y|\tilde{\sigma})) \otimes f(x,y) \\ &= \widehat{g}(x,y|k\tilde{\sigma}) \otimes f(x,y) - \widehat{g}(x,y|\tilde{\sigma}) \otimes f(x,y) \\ &= \sum_{i=0}^{n}\lambda_i\Pi_i(x,y) \otimes f(x,y) - \sum_{j=0}^{n}\lambda_j\Pi_j(x,y) \otimes f(x,y)\end{aligned} \tag{10}$$

[1] In practice, $k \in]1, \sqrt{2}]$.

where $\otimes$ is the global convolution product.

Obviously, the $\Pi_i(x,y) \otimes f(x,y)$ and $\Pi_j(x,y) \otimes f(x,y)$ products of Eq. (10) can be obtained with the integral image at a complexity $O(N)$ with N the image size. As a result, approximation of the DoG operator can be achieved with $2(n+1)$ accesses to the integral image where $n+1$ is the number of box filters.

A core problem with Gaussian kernel approximation is to fix the n, $\Pi_i(x,y)$, and λ_i parameters of Eq. (8). The approach used in the literature [17] is the minimization of the Mean Square Error (MSE) of Eq. (11), which can be achieved by any appropriate numerical method for regression. In [17], the LASSO algorithm is used to solve this problem,

$$MSE = \sum_{(x,y)\in[0,\omega]} \left(g(x,y|\sigma) - \widehat{g}(x,y|\sigma)\right)^2 \tag{11}$$

where ω^2 is a size of Gaussian operator.

2.3 Scale-space representation

As discussed in section 2.1 and Fig. 2, detection using the LoG operator relies on the scale parameter σ. The operator must be controlled with a filter bank at different scales $[\sigma_0,\ldots,\sigma_m]$ for optimum detection. The design of a time-efficient and well-adapted filter bank is referred to as a scale-space representation problem in the literature. The traditional and baseline approach is to control the scale-space with an exponential model as the SIFT descriptor. For stroke detection, the literature reports a linear model in which parameter σ is derived from the stroke with parameter w. This method is introduced and defined as the stroke model [18], and optimization of the model for stroke detection is investigated at the experimental level in [19].

Fig. 3 illustrates the model. The general idea is to assess for the convolution response between a LoG-based operator and a stroke signal modeled as a unit step function. We can then express the null cases with the derivatives to obtain the minimum/maximum of the convolution product. Assuming that these minimums/maximums are located at the center of the stroke $w/2$, we can present the standard deviation σ as a function $\sigma = \varpi(w)$.

Assuming the image signal is a function[2] $a\otimes\Pi(x)$, where $\Pi(x)$ is the step function Eq. (12) and a as the signal amplitude, the convolution product with the LoG operator $\nabla^2 g(x)$ is given in Eq. (13).

$$\Pi(x_0 - x) = \begin{cases} 0 & x < x_0 \\ 1 & \text{otherwise.} \end{cases} \tag{12}$$

$$\begin{aligned} h(x_0) &= a(\Pi \otimes \nabla^2 g)(x_0) \\ &= a\int_{-\infty}^{+\infty} \Pi(x_0 - x)\nabla^2 g(x)dx \end{aligned} \tag{13}$$

[2] For simplification, considering the 1D case.

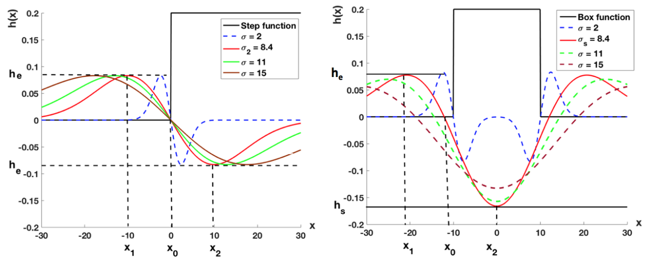


Fig. 3: LoG responses at different scales to (a) a step function (b) a boxcar function of size $w = 21$.

As $\Pi(x_0 - x)$ is located at x_0, the convolution product $\Pi(x_0 - x) \otimes \nabla^2 g(x)$ over x equals the summation $\nabla^2 g(x)$ centered at x_0.

With normalization and approximation of $\nabla^2 g(x)$, as given in Eq. (6), Eq. (13) is reformulated into Eq. (14),

$$\begin{aligned} &(k-1)\sigma^2 h(x_0) \\ &\approx \int_{-\infty}^{+\infty} a(g(x_0 - x|k\widetilde{\sigma}) - g(x_0 - x|\widetilde{\sigma}))dx \end{aligned} \tag{14}$$

where a is the signal amplitude, $h(x_0)$ is the convolution product of $a(\Pi\otimes\nabla^2 g)(x_0)$, k is the parameter that approximates the step offset, and $\widetilde{\sigma}$ is the scale parameter of the DoG function. From the derivative of Eq. (14), the local extremal optimum is obtained as Eq. (15) with the k parameter at locations $x_{1,2}$.

$$x_{1,2} = \pm k\widetilde{\sigma}\sqrt{\frac{2\ln k}{k^2-1}} \tag{15}$$

As given in Eq. (15) and illustrated in Fig. 3 (a), it is noted that the $x_{1,2}$ locations depend on the σ parameter. While bringing $x_2 = x_0 + w/2$ to the center of the stroke with w the stroke width parameter and using Eq. (15), we can obtain the optimum scale σ_s Eq. (16),

$$\sigma_s = \varpi(w) = \frac{1}{2k}\sqrt{\frac{k^2-1}{2\ln k}}w = \Phi(k)w \tag{16}$$

where $\varpi(w)$ is a linear function that has a slope controlled by $\Phi(k)$ derived from the step offset k.

As illustrated in Fig. 3 (b), two responses, h_e and h_s appear within the model at the $x_{1,2}$ locations with σ_s.

The response h_e characterizes the edge of the stroke and is obtained with Eq. (17) while bringing σ_s in Eq. (16) back to Eq. (14) and approximating the Gaussian integral at any location in Eq. (14) with the $erf(x)$ Gaussian error function $erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$.

$$h_e = \frac{a}{2}\left(erf\left(k\sqrt{\frac{\ln k}{k^2-1}}\right) - erf\left(\sqrt{\frac{\ln k}{k^2-1}}\right)\right) \tag{17}$$

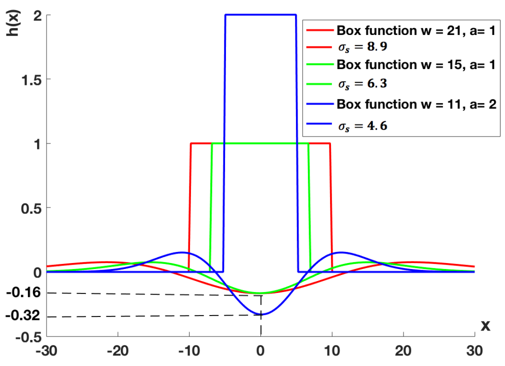


Fig. 4: LoG responses at different signal amplitudes and widths of the box function with $k = \sqrt{2}$.

For simplification of notation, Eq. (17) is given by considering $x_0 = 0$. In this equation, a is the signal amplitude and k is the offset parameter.

A peak response h_s appears at the middle of the stroke $w/2$ with w the stroke width. This response decreases while shifting the scaling parameter σ around the σ_s optimum Fig. 3 (b). It is worth noting that no mathematical formulation for h_s is proposed in [18]. The results from the proposed proof interpolate the stroke response from a step function. Simulation reveals a value h_s that is independent of the scale parameter σ and proportional to the signal amplitude a, as illustrated in Fig. 4.

The optimization within the scale-space considering the stroke model is attained while applying quantifying to $\sigma_s = \varpi(w)$ in Eq. (16) with $w \in [w_{min}, w_{max}]$ as a discrete value. The size of the filter bank is then correlated to the stroke width gap of the considered detection problem, thereby yielding $m = w_{max} - w_{min}$. With a DoG formulation, this requires $2(m+1)$ Gaussian kernels for detection.

3 Proposed method

As discussed in section 2, the RT-LoG operator for scene text detection relies on box filtering and the stroke model. In this paper, we propose a new method that takes advantages of the Gaussian kernel distribution and decomposition with box filtering to achieve the strongest optimization. Section 3.1 provides the problem statement. Then, sections 3.2 and 3.3 detail the two main optimization stages offered by our method.

3.1 Problem statement

Box filtering results in application of Eq. (10) for a filter bank $[\sigma_0, \dots, \sigma_m]$, with $\sigma_0, \dots, \sigma_m$ the scale parameters, obtained with the stroke model Eq. (16). This process can be formalized as a global product Eq. (18) achieving an element-wise multiplication $\circ$ between two matrices $f \otimes B, C$ of scalar values. Then, row summing is attained through multiplication with a column vector of 1.

$$A = ((f \otimes B) \circ C)\mathbf{1} \tag{18}$$

In Eq. (18), A, B and C have the same form as,

$$A = \begin{bmatrix} L_{\sigma_0} \\ L_{\sigma_i} \\ L_{\sigma_m} \end{bmatrix} B = \begin{bmatrix} \Pi_{00} & \dots & \Pi_{0n} \\ \vdots & \Pi_{ij} & \vdots \\ \Pi_{m0} & \dots & \Pi_{mn} \end{bmatrix} C = \begin{bmatrix} \lambda_{00} & \dots & \lambda_{0n} \\ \vdots & \lambda_{ij} & \vdots \\ \lambda_{m0} & \dots & \lambda_{mn} \end{bmatrix}.$$

B is a matrix of box filter functions Π_{ij} in which $f \otimes B$ results in a matrix of scalar values corresponding to the spatial products $f \otimes \Pi_{ij}$. C represents the weight parameters λ_{ij}. Within the matrices B, C, the columns refer to the scale-space and filter bank, respectively, whereas the rows are related to the spatial convolution and the boxes used within the $\widehat{DoG}$ operator Eq. (10). The matrices B, C have a size $(m+1, n+1)$ of $i \in [0, m]$ and $j \in [0, n]$, where $m+1$ and $n+1$ are the size of the filter bank, and number of box filters, respectively.

The global convolution of Eq. (10) using the above matrix notation can be reformulated as Eq. (19). For simplification of notation, we note $l = \frac{n+1}{2}$ is the middle index for the rows. In addition, the subtraction operation in Eq. (10) is embedded in range the $[l, n]$ of the λ_{ij} weights in C.

$$L_{\sigma_i} = \sum_{j=0}^{l-1} f \otimes \Pi_{ij} \lambda_{ij} + \sum_{j=l}^{n} f \otimes \Pi_{ij} \lambda_{ij} \tag{19}$$

The left $j \in [0, l[$ and right $j \in [l, n]$ parts of matrices B, C are related to the $g_{k\tilde{\sigma}}$ and $g_{\tilde{\sigma}}$ distributions within the $\widehat{DoG}$ operator Eq. (9), respectively. The global product Eq. (18) results in a vector A of size $(m+1)$ containing the global convolutions L_{σ_i} at the different scale $i \in [0, \dots, m]$.

The global product of Eq. (18) requires $P = (m+1) \times (n+1)$ operations for element-wise multiplication $\circ$ plus P operations for row summing. Considering a 128-bit CPU architecture with a 32 bits of Integer coding, 4 elements are processed at a time with vectorization. Vectorization can be applied to the $\circ$ product and the row summing with accumulation, which can be attained in $P/2$ operations.

The global product of Eq. (18) also requires establishment of matrices $f \otimes B, C$. Matrix C is obtained offline with regression and MSE minimization of Eq. (11). Matrix $f \otimes B$ must be computed online from the integral image to obtain the different averaging products $f \otimes \Pi_{ij}$. These products are acquired while summing the integral image values. The integral image can be obtained by $O(2N)$ operations with recurrence [23] where N is the image size. Then, the averaging products $f \otimes \Pi_{ij}$ are collected with Eq. (20) as illustrated in Fig. 5. The summing operations can be supported by vectorization, resulting in $\approx P$ operations. However, as shown in Fig. 5, the integral image values cannot be accessed in a continuous fashion in memory. This process requires $4P$ accesses to constitute the vectors $s^0, \dots, s^3$ used in Eq. (20),

$$f \otimes \begin{bmatrix} \Pi_{0j} \\ \vdots \\ \Pi_{mj} \end{bmatrix} = \left(\begin{bmatrix} s_0^0 \\ \vdots \\ s_m^0 \end{bmatrix} - \begin{bmatrix} s_0^3 \\ \vdots \\ s_m^3 \end{bmatrix} \right) + \left(\begin{bmatrix} s_0^2 \\ \vdots \\ s_m^2 \end{bmatrix} - \begin{bmatrix} s_0^1 \\ \vdots \\ s_m^1 \end{bmatrix} \right) \tag{20}$$

where $s^0, \dots, s^3$ are the vertexes of integral boxes, and $m+1$ is the number of scales.

Fig. 6 presents the overall pipeline, which is composed of fours steps, (I) to (IV). The first (I) step is the box selection step, which is performed offline to constitute the B, C matrices. The next steps are done online with (II, III) access to the integral image to obtain the vectors and averaging products Eq. (20) and (IV) the global product Eq. (18). Table 3 recapitulates the total amount of online operations (steps II, III, and IV). It can be seen that the complexity of the pipeline is mainly dominated by steps (II) and (III) to attain the averaging products $f \otimes \Pi_{ij}$. These products depend on the number of used boxes Π_{ij} and the size P of matrices B and C, which are obtained by the box selection method (I).

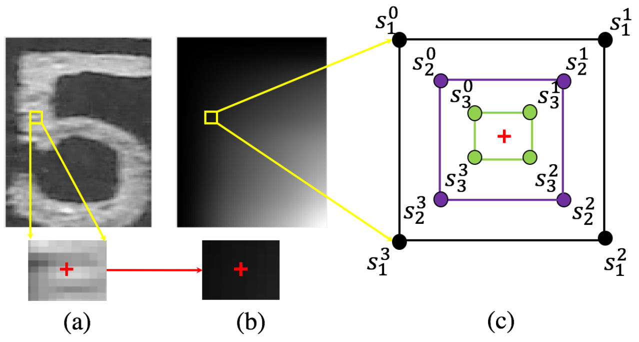


Fig. 5: Computing the averaging products in the scale-space domain using an integral image: (a) the image (b) the corresponding integral image (c) the local box functions.

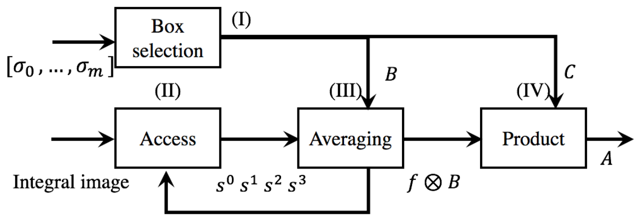


Fig. 6: The pipeline to obtain the $\widehat{DoG}$ product, (I) is an offline process whereas (II) (III) (IV) are the online processes.

In this paper, we propose a new approach for box selection as illustrated in Fig. 7. Our approach applies a two-step

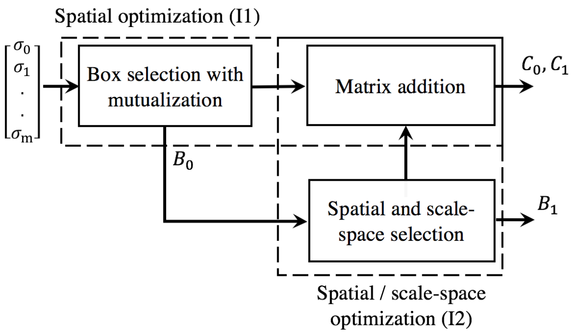

Spatial optimization (I1)
σ_0
σ_1
σ_m
Box selection with mutualization
Matrix addition
C_0, C_1
B_0
Spatial and scale-space selection
B_1
Spatial / scale-space optimization (I2)

Fig. 7: Spatial/scale-space optimization for the box selection step (I) presented in Fig. 6.

Table 3: The number of operations for the online processes in the pipeline of Fig. 6.

Access (II)	Averaging (III)	Product (IV)
4 P	$\approx P$	$P/2$

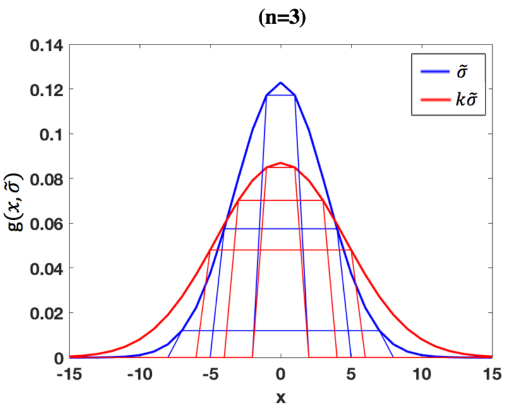


Fig. 8: The box functions to approximate two Gaussian kernels $k\tilde{\sigma}$ and $\tilde{\sigma}$ with $\tilde{\sigma} = 3.2$ and $k = \sqrt{2}$.

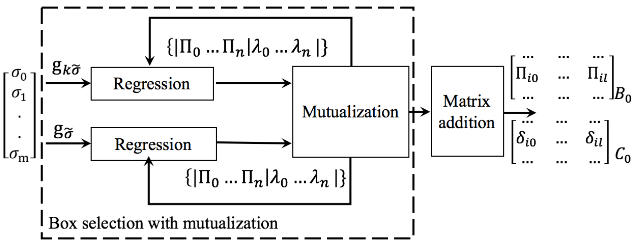
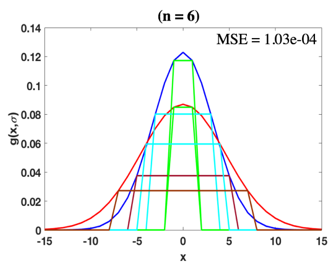
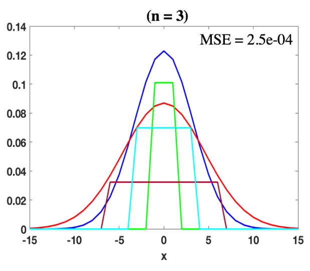


Fig. 9: Box selection with mutualization for the $\widehat{DoG}$.

process for optimization within the spatial domain (I1) and spatial/scale-space domains (I2). The first step (I1) takes advantage of box selection with mutualization within the $\widehat{DoG}$ product. The output serves as parameter training for the second step (I2) proposing a global spatial/scale-space model for selection. The overall method substitutes the box selection step (I) in the pipeline of Fig. 6 resulting in a large op-



(a)



(b)

Fig. 10: The (a) standard approach (b) proposed approach.

timization. We detail the selection within the spatial domain (I1) and the spatial/scale-space domains (I2) in next sections 3.2 and 3.3, respectively.

3.2 Optimization within the spatial domain

As we presented in section 2.2 and Eq. (8), we can approximate a Gaussian filter using a set of box functions. These box functions are then convolved and summed to get a $\widehat{DoG}$ product Eq. (9). The standard approach discussed in [17] is to apply two separate approximation processes to the Gaussian kernels $k\widetilde{\sigma}$ and $\widetilde{\sigma}$ where $\widetilde{\sigma}$ is the scale parameter and k is the parameter to approximate the step offset. However, given that the k parameter is supposed to be low for the local derivative Eq. (5), the two Gaussian distributions are close, as shown in Fig. 8. Thus, a common set of boxes could be used to approximate the two kernels. We develop these aspects here.

We can assume in Eq. (19) that we have $\Pi_{ij} = \Pi_{ij+l}$ with $l = \frac{n+1}{2}$ with $n+1$ the number of box filter function. Then, we can reformulate Eq. (19) into Eq. (21), where $\delta_{ij} = (\lambda_{ij} + \lambda_{ij+l})$ are new coefficients obtained with a numerical method for regression and minimization of the MSE Eq. (11).

$$\begin{aligned} L_{\sigma_i} &= \sum_{j=0}^{l-1} f \otimes \Pi_{ij}\lambda_{ij} + \sum_{j=l}^{n} f \otimes \Pi_{ij}\lambda_{ij} \\ &= \sum_{j=0}^{l-1} f \otimes \Pi_{ij}(\lambda_{ij} + \lambda_{ij+l}) = \sum_{j=0}^{l-1} f \otimes \Pi_{ij}\delta_{ij} \end{aligned} \tag{21}$$

To mutualize boxes, such as $\Pi_{ij} = \Pi_{ij+l}$ $\forall j \in [0,l[$, a new pipeline for box selection must be fixed. We propose a new strategy in Fig. 9. This strategy applies local mutualization with a close-loop methodology, which relies on the general observation that matrix B is composed of pairs of boxes. A pair of boxes is a group of the closest boxes in B, such as $\|\Pi_{ij}\| \approx \|\Pi_{ij+l}\|$ $\forall j \in [0,l[$, with $\|\Pi_{ij}\|$ representing the radius of the box function Π_{ij}. The pairs of boxes still fit within the constraint $(\|(\Pi_{ij}\| \approx \|\Pi_{ij+l}\|) < \|(\Pi_{ij+1}\| \approx \|\Pi_{ij+1+l}\|)$.

The local mutualization combines the pair of boxes while minimizing the MSE for the $\widehat{DoG}$ function. Fig. 10 shows the process by which three pairs of boxes (green, cyan, and brown) are merged. We detail that process here.

- $\ast$ **Initialization:** apply the method for regression to the $\widetilde{\sigma}$ and $k\widetilde{\sigma}$ Gaussian kernel distributions to obtain the B,C matrices.
- **Step 1:** for a pair of boxes Π_{ij}, Π_{ij+l} $\forall j \in [0,l[$ to fix a set of solutions for mutualization $\|\Pi_{uv}\| \in [\|\Pi_{ij+l}\|, \|\Pi_{ij}\|]$. $\forall$ Π_{uv}, apply sub-steps 1 to 3.
  - **Sub-step 1:** substitute the Π_{ij}, Π_{ij+l} functions with Π_{uv} in B.
  - **Sub-step 2:** apply the method for regression to refine the λ_{ij} coefficients for the $\widetilde{\sigma}$ and $k\widetilde{\sigma}$ Gaussian kernel distributions.
  - **Sub-step 3:** compute the MSE between the DoG and $\widehat{DoG}$ functions, as noted in Eq. 11.
- **Step 2:** the lowest MSE from sub-step 3 is used to fix the Π_{uv} solution. Then, repeat step 1 for the next pair of boxes.

This procedure is repeated to approximate all the kernels of the bank filter $[\sigma_0, ..., \sigma_m]$ with $\sigma_0, \ldots, \sigma_m$ the scale parameters, as shown in Fig. 9. The output coefficients λ_{ij} of the selection are processed via matrix addition to get C_0 while applying Eq. (21). Matrices B_0, C_0 have size $P/2$ compared to the brute-force strategy developed in section 3.1, which requires P elements. Matrix B_0 is used in the second step of selection to optimize in the spatial/scale-space domains, as shown in Fig. 7. We detail these aspects in next section 3.3.

3.3 Optimization within the spatial / scale-space domains

In section 3.2, optimization in the spatial domain was discussed. Optimization can also be extended in the scale-space domain. Therefore, we indicate these aspects here. The scale-space domain and filter bank used in the global product Eq. (18) are based on the stroke model Eq. (16). As the stroke model is a linear function, the Gaussian kernels in the scale-space domain have a compact distribution, as shown in Fig. 11 (a). Approximation of the Gaussian kernels with a box filter Eq. (8) can result in a large number of duplicate boxes at the different scales. Redundant and close averaging products will appear when computing the $f \otimes B$ components in the global product in Eq. (18).

To address this problem, we propose a mixed spatial and scale-space method for box selection. This method is shown in Fig. 11 (b,c). This method takes advantage of the linear distribution of Gaussian kernels in the scale-space domain to drive box selection. The linear distribution results in a large number of overlapping boxes among the Gaussian kernels.

A compact set of boxes could then be generated to obtain the global product of Eq. (18).

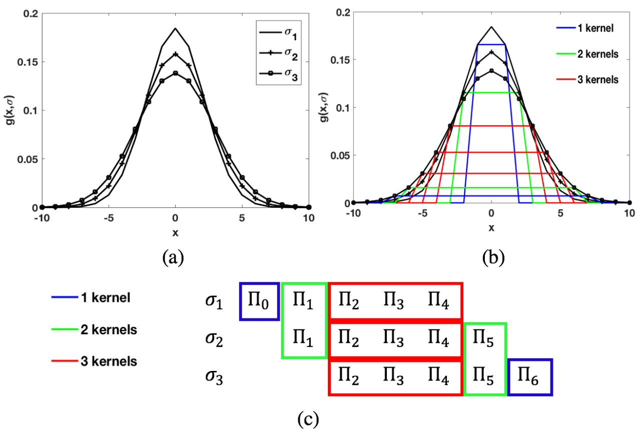


Fig. 11: (a) the Gaussian kernel distributions based on the stroke model (b,c) shared box functions between Gaussian kernels.

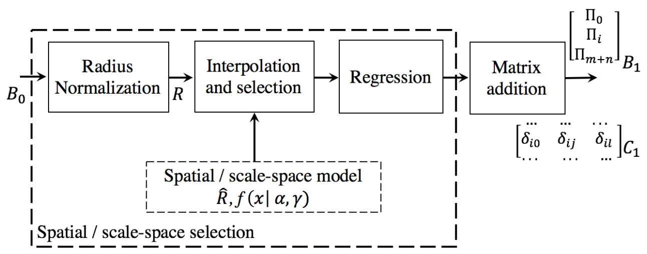

B0
Radius Normalization
R
Interpolation and selection
Regression
Matrix addition
B1
C1
Spatial / scale-space model
R̂, f(x | α, γ)
Spatial / scale-space selection

Fig. 12: Spatial/scale-space selection.

Fig. 12 details our overall approach. Our spatial/scale-space model is interpolated with the set of boxes B_0 obtained with mutualization, as shown in Fig. 9. A normalization of radius is performed on the B_0 matrix. The selection output is reprocessed with regression to refine the coefficients λ_{ij} to obtain the C_1 matrix. These different aspects are clarified in next paragraphs.

Radius normalization: We define R as a matrix of scalar values provided by the radius $\|\Pi_{ij}\|$ of the Π_{ij} functions in the B_0 matrix. The radius can be expressed as Eq. (22) with the application of normalization parameters a_{ij} to the scale parameters σ_i, $\Phi(k)$ derived from the step offset k from Eq. (16).

As the Gaussian distributions have a regular range of $[-\pi\sigma, \pi\sigma]$, the scale parameters σ_i can be weighted with the π value to bound the normalization parameters $a_{ij} \in [0,1]$. A final reformulation can be obtained with the stroke model Eq. (16). In the normalized form of the stroke model, the R matrix is given in Eq. (23), where $r()$ is a replicate function of a column vector, $n+1$ is the number of boxes at each scales, $m+1$ is the number of scale parameters.

$$\|\Pi_{ij}\| = a_{ij}\pi\sigma_i = a_{ij}\pi\Phi(k)w_i \tag{22}$$

$$R = \pi\Phi(k)\begin{bmatrix} a_{00} & \dots & a_{0n} \\ \vdots & a_{ij} & \vdots \\ a_{m0} & \dots & a_{mn} \end{bmatrix} \circ r\left(\begin{bmatrix} w_0 \\ \vdots \\ w_m \end{bmatrix}, n+1\right) \tag{23}$$

Spatial/scale-space model: We fix $\widehat{R}$ as the spatial/scale-space model for a radius that will approximate the R matrix. To fit the box selection approach shown in Fig. 11 (b,c), constraints are applied to the $\widehat{R}$ matrix Eq. (24). We first normalize the spatial distribution of the box functions in the kernels over all the scales with a single set of normalization parameters $(\hat{a}_0, \dots, \hat{a}_n)$. The $\hat{a}_i$ coefficients in the $\hat{R}$ matrix are the approximation of the a_{ij} coefficients in the R matrix. Therefore, we constrain the spatial/scale-space distributions within the stroke model while establishing equality between the diagonal elements in the matrix. We achieve the relation $\hat{a}_{j+\Delta}w_i = \hat{a}_j w_{i+\Delta}$ with Δ as the offset between the elements of the $\hat{R}$ matrix. In this relation, we have $w_{i+\Delta} = w_i + \Delta$ with quantization of the stroke model described in section 2.3.

For a proper reformulation, we fix $i = j = 0, \hat{a}_0 = \gamma, \alpha = 1/w_0$ and $\Delta = x$ to obtain the linear function Eq. (25).

$$\hat{R} = \pi\Phi(k)\begin{bmatrix} \hat{a}_0w_0 & \hat{a}_1w_0 & \hat{a}_2w_0 & \dots & \hat{a}_nw_0 \\ \hat{a}_0w_1 & \hat{a}_1w_1 & \hat{a}_2w_1 & \dots & \vdots \\ \hat{a}_0w_2 & \hat{a}_1w_2 & \hat{a}_2w_2 & \dots & \vdots \\ \dots & \dots & \dots & \dots & \hat{a}_nw_m \end{bmatrix} \tag{24}$$

$$f(x|\alpha,\gamma) = \gamma + \alpha\gamma x \tag{25}$$

This function Eq. (25) is illustrated in Fig. 13 and determines the spatial / scale-space distributions of the boxes. Here, $f(x|\alpha,\gamma)$ returns an estimation $\hat{a}_i$ for the normalized parameters a_{ij} with x as the index / offset of the box function in $\hat{R}$. This function is controlled with two parameters α, γ.

Interpolation and selection: Our model of Eq. (24) requires the α, γ parameters to be fixed. These parameters control the distribution over the spatial and scale-space domains for optimizing the box selection. However, box selection must also guarantee the accuracy of the approximation of the Gaussian kernels. To address this problem, we interpolate the α, γ parameters from the B_0 matrix. Indeed, this matrix provides a box selection for the accurate approximation of the $\widehat{DoG}$ products, as detailed in section 3.2.

The B_0 matrix is first processed with normalization Eq. (23). Then, a quantization process is applied, such as Lloyd's

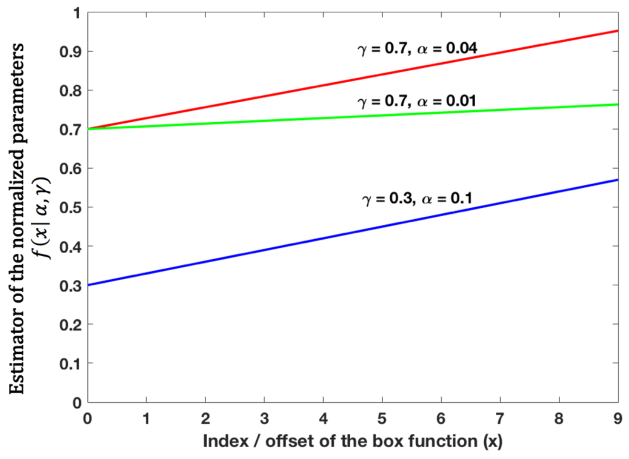


Fig. 13: The function $f(x|\alpha,\gamma)$ for spatial / scale-space selection.

algorithm, to get the prominent coefficients a_{ij} in the R matrix. The a_{ij} coefficients are normalized over the scales for the kernel distributions, and we obtain a vector of $(n+1)$ coefficients $(a_0,...,a_n)$ with the corresponding offsets $(0,...,n)$. With the reformulation of Eq. (25) and changing the variable $\gamma^{-1} = 1/\gamma$, we obtain a system of a linear equation Eq. (26). This system can be solved by using any linear solver [24].

$$\gamma^{-1}\begin{bmatrix}a_0\\ \vdots\\ a_n\end{bmatrix} - \alpha\begin{bmatrix}0\\ \vdots\\ n\end{bmatrix} = \begin{bmatrix}1\\ \vdots\\ 1\end{bmatrix} \tag{26}$$

The application of the function Eq. (26) to a discrete interval (0,1,2,...) results in a set of estimation of normalized coefficients $(\hat{a}_0,\hat{a}_1,\hat{a}_2,\ldots)$. These coefficients control the spatial/scale-space selection of the boxes where the radius is obtained with Eq. (27). Due to the equality of diagonal elements in $\hat{R}$, the number of box functions to generate with the approach is $(m+n+1)$. This corresponds to $P = \frac{1}{2}(m+n+1)$ elements that is $\ll (m+1)(n+1)$,

$$||\Pi_i|| = \pi\Phi(k)\hat{a}_i w_0 \tag{27}$$

where $||\Pi_i||$ is the box radius, $\Phi(k)$ is derived from the step offset k from Eq. (16), $\hat{a}_i$ is the estimation of the normalized coefficients a_i.

Regression and global product: As shown in Fig. 12, the set of boxes $(\Pi_0,...,\Pi_{m+n})$ is reprocessed with regression while shifting to any subset Π_i,Π_{i+n} to refine the λ_{ij} coefficients and approximate the $k\tilde{\sigma}$ and $\tilde{\sigma}$ distributions. The λ_{ij} coefficients are processed with matrix addition to obtain C_1 while applying Eq. (21). The B_1 and C_1 matrices are pushed in the pipeline shown in Fig. 6 requiring $(m+n+1)$ spatial products $f\otimes\Pi_i$. The global product of step (IV) is processed with shifting while applying B_1 to get A with Eq. (18).

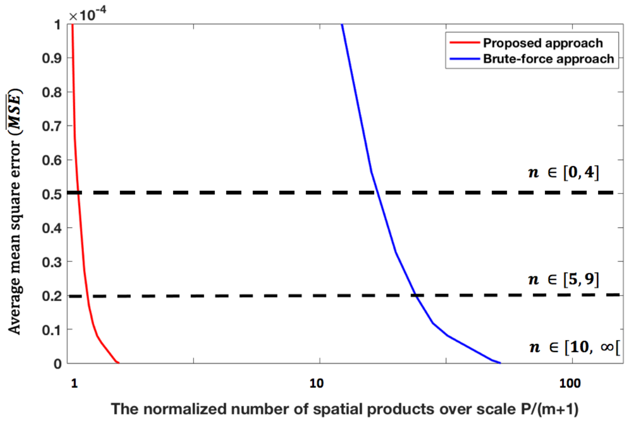


Fig. 14: DoG approximation of the operators with complexity (the $x-$axis is plotted in the logarithm domain).

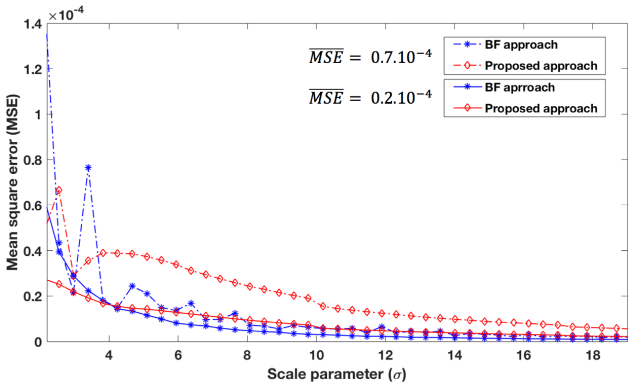


Fig. 15: DoG approximation of the operators over scales at the equal $\overline{MSE}$.

4 Performance evaluation

In this section, we present the performance evaluation and results of our method. Section 4.1 characterizes the $\widehat{DoG}$ operators with optimization in the spatial and scale-space domains. Section 4.2 characterizes the operators for scene text detection, whereas, section 4.3 addresses the processing time aspects. Finally, section 4.4 highlights how the operator can support the full pipeline for scene text detection and gives a comparison with the top systems of the literature.

4.1 Characterization of the $\widehat{DoG}$ operators with optimization

We characterize how the $\widehat{DoG}$ operators discussed in this paper approximate the DoG operator. The $\widehat{DoG}$ operators are given with and without optimization in the spatial and scale-space domains, which results in two operators, the brute-force RT-LoG operator detailed in section 3.1 and the pro-

Fig. 16: Images from ICDAR2017 RRC-MLT dataset [3].

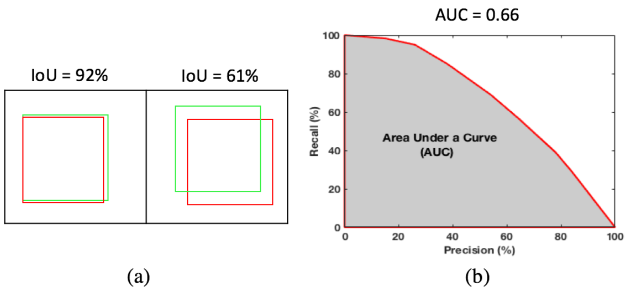


Fig. 17: The metrics of (a) Intersection over Union (IoU) (ground truth is in green, detected boxes are in red) and (b) Area under a Curve (AUC).

posed operator with spatial/scale-space optimization detailed in sections 3.2, 3.3 and Fig. 7.

Fig. 14 shows our results. Comparisons with the DoG operator are reported from the MSE Eq. (11). To consider the scale-space aspect, the MSE was computed at every scale over the range $[\sigma_0, \dots, \sigma_m]$ and then averaged to obtain the $\overline{MSE}$. In the experiments, we bounded $m = 40$ with $\sigma \in [2.1, 19.1]$. We compare the $\overline{MSE}$ obtained at every P value. For the sake of graphical representation, the P values shown in Fig. 14 are normalized with the scaling parameter $(m+1)$ and are displayed in the logarithm domain. The overall protocol characterizes the robustness of the methods against P values. All the experiments are driven with the LASSO algorithm for the box selection with regression, as done in [17].

As illustrated in Fig. 14, our approach results in one to two orders of magnitude in terms of P differences between the operators. The largest differences are obtained with a low $\overline{MSE}$ and then for a high accuracy of the operators, which is achieved when a deeper spatial filtering is applied with a large value for $n \in [5, \infty[$.

In addition, Fig. 15 details the MSE obtained at every scale $[\sigma_0, \dots, \sigma_m]$ by the two operators. For comparison, we fixed the P parameter within the two operators to achieve an equal $\overline{MSE}$. As shown in Fig. 15, the proposed operator results in a more stable response over all the scales due to the mutualization process applied in the operator. With mutualization, the DoG approximation becomes less sensitive to quantization at the low level scales. This gap is reduced when P is increased in the operators to get a lower $\overline{MSE}$.

4.2 Characterization of operators for scene text detection

In this section, we present the performance evaluation for scene text detection. Section 4.2.1 introduces the datasets. Section 4.2.2 discusses the characterization metrics and section 4.2.3 explains the protocol. The competitive operators are introduced in section 4.2.4. Finally, the scene text localization results are discussed in section 4.2.5.

4.2.1 Datasets

Several public datasets have been proposed for evaluating the performance of text detection methods. We selected the recent dataset of the international contest ICDAR2017 RRC-MLT [3]. This dataset 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). The groundtruth is given in term of bounding boxes. Bounding boxes are represented by four corner points for each text word. Fig. 16 shows examples of images. Compared to other datasets in the literature, this dataset has a particular focus on the multi-lingual text and offers a deeper challenge in terms of scalability.

For the special purpose for the evaluation of processing time, we used the Challenge 4 of ICDAR2015 dataset that is more common in the literature [25]. This dataset contains 1000 training images and 500 test images at the HD resolution (1280 x 720).

4.2.2 Characterization metrics

For the characterization metrics, we followed the recommendations of the international contest [3]. 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. Detection is obtained if a detected bounding box has more than 50% overlap (the IoU criterion given in Fig. 17 (a)) with a bounding box in the groundtruth, which plays a rule as true posivitives (TP). The unmatched boxes in the detection and groundtruth are false positives (FP) and negatives (FN), respectively. The detection cases serve to compute the regular metrics precision (P), recall (R) and F-measure (F), as noted in Eq. (28).

$$P = \frac{TP}{TP+FP} \qquad R = \frac{TP}{TP+FN} \qquad F = 2\frac{PR}{P+R} \tag{28}$$

The recall R measures the ability to detect the text of operator while the precision P evaluates the ability not to invent text. Moreover, one of the popular metrics of the method assessment is the Area Under a Curve (AUC), as shown in Fig. 17 (b). This metric is independent on any particular threshold. We estimates the AUC scores under the precision-recall curves.

Table 4: The different real-time operators for scene text detection.

	Operator	Parameters	Outputs
LoG-based operators	Proposed RT-LoG	$P = 45$ $(n = 4, m = 40)$, $k = \sqrt{2}, \sigma \in [2.1, 19.10]$	$h(x,y)$, $\sigma_s(x,y)$,w
	Brute-force RT-LoG	$P = 410$ $(n = 4, m = 40)$, $k = \sqrt{2}, \sigma \in [2.1, 19.10]$	$h(x,y)$, $\sigma_s(x,y)$,w
	SIFT	$k = \sqrt{1.03}, m = 80, \sigma \in [2.1, 19.10]$	$h(x,y)$, $\sigma(x,y)$
	SWT	$\sigma_0 = 0.9, TL = 70, TH = 250$	$mag(\nabla f)$, w
	BSV	$\sigma_0 = 0.9, TL = 0.007, TH = 0.012$	$h(x,y)$
CC-based operators	MSER	$TL = 60, TH = 14400$, $AreaVariation \in [0.1, 1]$	Region size, CC
	FASText	The contrast intensity	Corner point, circle size

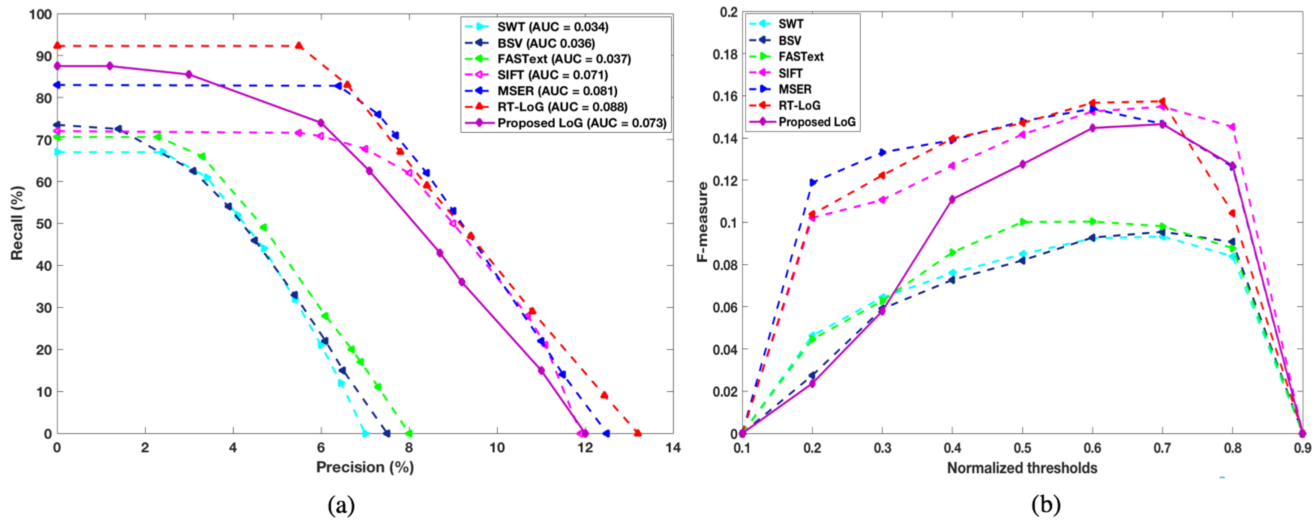


Fig. 18: Comparison of operators (a) P/R with AUC (b) F-measure scores with normalized thresholds on ICDAR2017 RRC-MLT dataset.

It is worth noting that some degraded texts in the dataset are marked as "don't care" boxes and are ignored in the evaluation process.

4.2.3 *Characterization protocol*

The groundtruth and metrics discussed in sections 4.2.1 and 4.2.2 are not adapted for the characterization of operators. Indeed, the operators provide detection results at the keypoint level. To evaluate and compare operators for scene text detection, their outputs must be processed to obtain the text regions. For performance evaluation, a relevant method must be established. This method has to be common to all the operators and be used with the same conditions for training and testing. To do that, a standard approach in the literature is the character grouping which is achieved with different algorithms, such as clustering, adaptive thresholding or the minimum-area encasing rectangle [1].

To meet the needs of our performance evaluation, we processed the outputs of the operators with a standard grouping method using fast K-means clustering [26]. K-means clustering was applied to the operator and image features. These features include the localization (x, y) of keypoints and the color information of pixels $f(x, y)$ as used in [27].

4.2.4 *Comparative operators*

We compare the proposed RT-LoG operator against LoG-based operators introduced in Table 4. These operators include the SWT, BSV and brute-force RT-LoG operators. In addition, we apply SIFT operator used as a baseline LoG-based operator for scene text detection [28].

Several alternative time-efficient and real-time operators can be used for scene text detection [1]. Recent works tend to use Connected-Component (CC) analysis, which is mainly dominated by the MSER (Maximally Stable Extremal Regions) operator [12]. This operator, thus, represents a good competitor. Different improvements of the MSER operator have been proposed [29, 30]. We selected a general implementation of the MSER operator [31] along with the FASText operator [7], which provides adaptation to scene text detection.

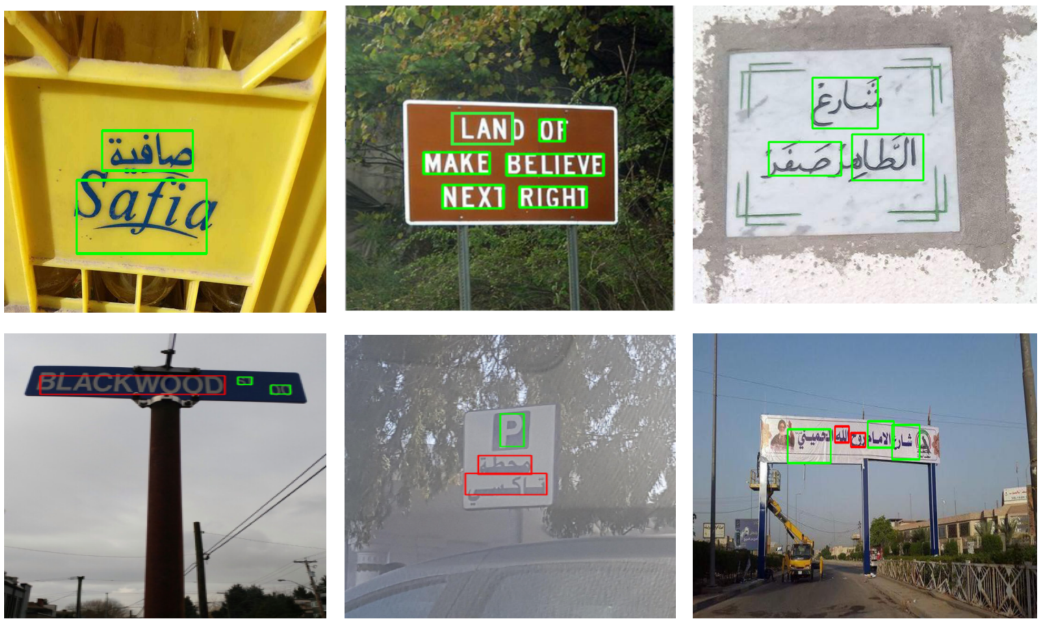
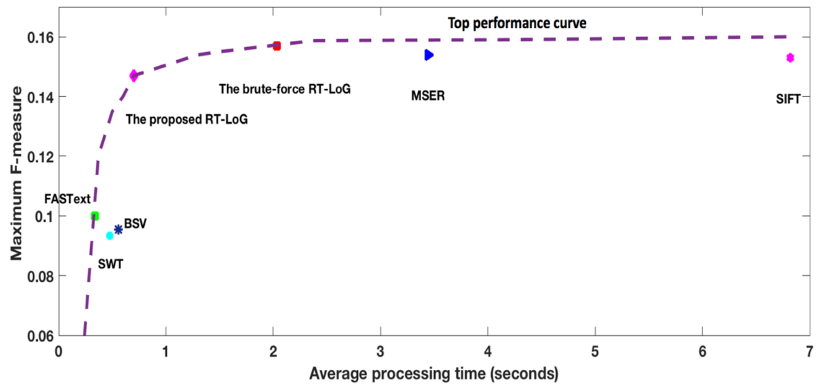


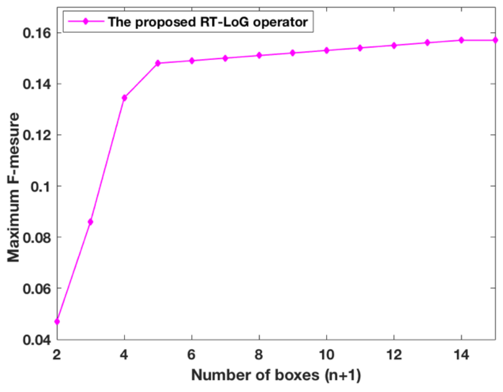
Fig. 19: Visual examples of text detection result of the proposed RT-LoG operator, (green) true positive (red) missed case.

Table 5: Processing time of detectors in (ms) performing with the C++ on a Mac-OS and an Intel(R) Core(TM) i7-4770HQ CPU 2.2 GHz with approximately a 32 GFLOPS SP performance on ICDAR2017 RRC-MLT dataset.

Resolutions / Operators	SD	HD	Full-HD	Quad HD	4K
SIFT	150	890	3254	5909	8917
MSER	90	450	1660	2496	5357
Brute-force RT-LoG	75	288	784	1750	2706
Proposed RT-LoG	65	150	320	700	990
BSV	61	136	226	541	835
SWT	40	120	170	457	756
FASText	24	75	121	312	525



(a)



(b)

Fig. 20: (a) comparison of operators with the maximum F-measure scores against the average processing time (b) the maximum F-measure score corresponding to the number of boxes for the proposed RT-LoG operator on ICDAR2017 RRC-MLT dataset.

4.2.5 Performance evaluation for text localization

The results of the performance evaluation for text localization are given in Fig. 18 (a) in terms of precision (P) and recall (R). The P/R scores are obtained while relaxing the thresholding on the operator responses and their parameters, as detailed in Table 4. These results highlight the close performance of the MSER, SIFT, brute-force RT-LoG and proposed RT-LoG operators for in balance P/R scores. At low precision, the brute-force RT-LoG operator attains the strongest recall performance among all the operators, which can be characterized by AUC scores.

Fig. 18 (b) provides the F-measure scores controlled with the normalized thresholds, which are first applied to the responses of the methods and then the parameters. The peaks within the curve correspond to the maximum F-measure score attained by the different methods. The MSER, SIFT and RT-LoG operators present a close performance with a $F \approx 0.15$. The brute-force RT-LoG operator achieves the strongest score, with a slight gap compared with the proposed RT-LoG operator. In both cases, the brute-force and proposed RT-LoG operators are set as $n+1=5$ (the number of box filters for spatial filtering), as shown Table 4.

For further experiments, shown in Fig. 20 (b) the relation between the maximum F-measure score against the number of box filters is determined. It can be seen that the number of box filters could be fixed between 4 and 6 to reach a close maximum for the F-measure score. This value ensures the robustness of the operators while maintaining a low complexity level for the processing time. Some visual examples of detection with the proposed operator are shown in Fig. 19.

4.3 Processing time

This section investigates the processing time, which depends on the complexity of the algorithms. For the FASText, RT-LoG and MSER operators, the complexity is linear $O(N)$ with N the image size. The complexity is $O(N\omega)$ with a small mask size $(\omega \times \omega)$ for the SWT and BSV operators. The filter bank for the SIFT operator is set as suggested in [19] and detailed in Table 4.

The parallelism support has a strong impact on the final results, which includes the use of a vectorization/SIMD architecture and intrinsic instructions, the multithread/ multicore or the GPU architectures. The different parallelism levels can offer an increase of one to two orders of magnitude, depending the quality of the implementation.

For an objective comparison, we aligned the operators at the same level of parallelism. The implementations are given with a single thread with the vectorization/SIMD architecture and intrinsic instructions. The goal is to exclusively evaluate the complexity side of the operators, while comparing the processing times. Table 5 presents the processing times for different image resolutions. The proposed RT-LoG operator performs better than the SIFT and MSER operators. The proposed operator is almost five times faster than the MSER operator and up to nine times compared to the SIFT operator. The proposed operator has a near equal performance compared to the those of the SWT and BSV operators, with a slight gap. The FASText operator is the fastest operator; it is one to three-fold faster compared to the proposed RT-LoG operator. Among all the operators, our experiments report minimal computational overhead with K-means clustering compatibility with a real-time strategy.

For a global comparison, Fig. 20 (a) provides the average processing time of the operators, obtained from the ICDAR 2017 RRC-MLT dataset against the maximum F-measure scores derived as shown in Fig. 18 (b). As shown in Fig. 20 (a), the FASText, brute-force and proposed operators fix the top performance curve among all the operators. The proposed RT-LoG operator appears as the top operator with a balanced performance between accuracy and time processing.

In addition, in Table 6, we provide the frame rates of the RT-LoG operator using a multithread/multicore support. Experiments are performed on a regular hardware architecture using the Intel Core i7-4770HQ CPU, 2.2 GHz with approximately a 32 GFLOPS SP[3] performance. Our computer is set with a time-sharing operating system Mac OS.

The frames are processed with gridding and each thread takes in charge of a particular area. This is a standard strategy for camera-based processing. The threads are synchronized /waked-up at any new frame. The number of threads has been set to 16 to reach the optimum performance while reducing the context switch in the system. With such a strategy, there is no guaranty to respect a deadline. The operating system is not provided specific kernel mechanisms for time management and for handling tasks with explicit time constraints. However, our operator and approach are supposed to be deployed on mobile systems where the time-sharing is the common model. Thus, our approach enters in a soft real-time methodology.

Table 6: Frames per second (FPS) with the proposed RT-LoG operator with multithreading/multicore, performing with the C++ on a Mac-OS and an Intel(R) Core(TM) i7-4770HQ CPU, 2.2 GHz with approximately a 32 GFLOPS SP performance.

Resolutions	Average FPS	Minimum FPS
Full-HD	57	46.5
Quad-HD	29.6	25.75
4K	13.6	11

[3] Single Precision.

The FPS presented in Table 6 for standard video resolutions. These FPS are derived from the distribution of the response times obtained with the different threads. Experiments have been obtained for long video sequences. As shown in Table 6, the average FPS = $1/\overline{RT}$ is computed with $\overline{RT}$ the average response time. This average FPS can be applied where a low-level latency is tolerated while processing consecutive frames. The minimum FPS = $1/RT_{max}$ is collected from the maximum response time. This FPS guarantees that there is null latency in the system. It can be seen that there is a small gap between the average and minimum FPS (15% to 20%). This can be explained by the predictability of our operator. We have a sharp upper bound appearing on the execution times.

The constraints $\overline{RT}, RT_{max}$ expressed above can be used as deadlines to target a null or a low-level latency with our operator for processing. As highlighted in Table 6, while applying these constraints the proposed RT-LoG operator can support approximately 30 FPS up to the Quad-HD resolution on a regular CPU architecture with a low-level latency.

4.4 Performance evaluation for scene text detection

The RT-LoG operator can support the full pipeline for scene text detection. For the need of the comparison, the metrics, protocol, and experiments, as detailed in sections 4.2.2, and 4.2.3, have considered the common outputs for the operators. They are the spatial coordinates (x,y) and the color information $f(x,y)$ of pixels. However, compared to the other operators, the RT-LoG operator provides additional meaningful spatial, scale-space and contrast information Table 4. This information includes the stroke width w, scale-space $\sigma_s(x,y)$ and the operator response $h(x,y)$. As discussed in [32], these features can drive a grouping method. They can serve in addition for the scale prediction, the background/ foreground normalization, and contrast correction of characters before a text verification stage.

Fig. 21 presents the general architecture of the system proposed in [32] where the RT-LoG operator is embedded. As highlighted from that, this system can achieve a strong detection accuracy and is competitive with the top systems of the literature performing with end-to-end CNN models and GPU architectures. However, results reported in [32] are obtained while using the brute-force RT-LoG operator. As discussed in sections 4.2 and 4.3, the proposed RT-LoG operator is two to three time faster for an approximately equal performance for detection.

Table 7 reports the results from system in [32] while embedding our proposed RT-LoG operator. The results are compared against the state-of-the-art methods on ICDAR 2017 RRC-MLT dataset. Similar to [32], our system appears in the top results of the literature for F-measure score. It introduces a slight gap of less than 1.5 % error compared to the brute-force RT-LoG operator. Moreover, the RT-LoG based systems achieve the strongest recall score of the literature. These results are ensured that the use of the RT-LoG operator can allow a quite high detection of the text elements.

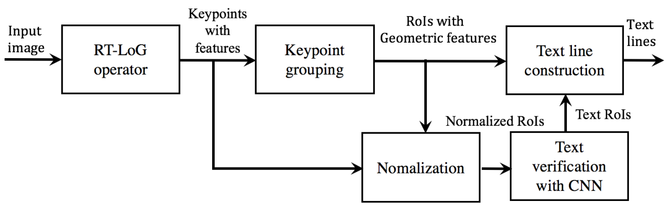


Fig. 21: The general architecture of the system of [32].

Table 7: Comparison of methods (P) precision (R) recall and (F) F-Measure on ICDAR2017 RRC-MLT.

Rank	Methods	$P(\%)$	$R(\%)$	$F(\%)$
1	PMTD [33]	**85.15**	72.77	**78.48**
2	FCN-MOML [34]	82.66	72.53	77.26
3	R-CNN-PAN [35]	80	69.8	74.3
4	LOMO MS [36]	80.2	67.2	73.1
5	Brute-force RT-LoG [32]	65.2	**82.1**	72.6
6	MOSTD [37]	74.3	70.6	72.4
7	Proposed RT-LoG	64.5	80	71.4
8	Fots [38]	81.86	62.30	70.75
9	AF-RPN [39]	75	66	70
10	Attention Model [40]	72	63.5	67.48
11	SCUT DLVClab1 [3]	80.3	54.5	65

Most of recent works reported the FPS on the ICDAR 2015 dataset. Table 8 presents the results including our system using the proposed RT-LoG operator. The evaluation is performed with a full parallelism support on the CPU while applying multicore/multithreading. For a fair comparison, Table 8 details the test architecture of different systems (either GPU or CPU) with their relative performances in TFLOPS SP. As emphasized in Table 8, our system has the second highest FPS while processing with a difference of two orders of magnitude in term of processing resources. All the top systems perform with the end-to-end CNN models requiring a GPU architecture.

As given in Table 9, the implementation with the fast RT-LoG operator attains a near 25% to 65% acceleration factors for the FPS compared to the brute-force implementation corresponding to the HD and Full-HD resolutions, respectively, whereas the operator is two to three time faster. This can be explained that the overall processing time required by the full pipeline are significantly dominated by the grouping and verification steps with the CNN, as illustrated in Table 10.

Finally, Fig. 22 shows a general comparison of the performances considering the F-measure scores, FPS and test architectures. The overall pipeline embedded the proposed RT-LoG operator has a close performance of the top performances of literature for the F-measure scores and FPS while

Table 8: Frame rate per second (FPS) among methods on the Challenge 4 of ICDAR2015 dataset.

Processing types / Methods	FPS	Architecture	Performances TFLOPS SP
FOTs-RT[38]	**22.6**	TITAN-Xp GPU	12.15
Proposed RT-LoG	20.2	CPU 2.2 GHz	**0.032**
Brute-force RT-LoG [32]	15.6	CPU 2.2 GHz	**0.032**
SSTD [43]	7.7	TITAN X GPUs	6.691
EAST [42]	6.52	TITAN-Xp GPU	12.15
MTS[41]	4.8	Titan Xp GPU	12.15
MOSTD[37]	3.6	Tesla K40m GPU	5.046

Table 9: Frame rate per second (FPS) between the brute-force and proposed RT-LoG.

Methods / Resolutions	Proposed RT-LoG	Brute-force RT-LoG	Acceleration factors
HD	20.2 FPS	15.6 FPS	25%
Full-HD	13 FPS	8.4 FPS	65%

Table 10: Average processing time in milliseconds (ms)/amounts of pixels, keypoints and RoIs of each step of the proposed method.

Types / Methods	HD	Data	Full-HD	Data
Proposed RT-LoG	180 ms	1.2 Mpixel	370 ms	2.2 Mpixel
Grouping	200 ms	5.2 Kkeypoints	360 ms	9.3 KKeypoints
Verification	336 ms	90 RoIs	420 ms	130 RoIs

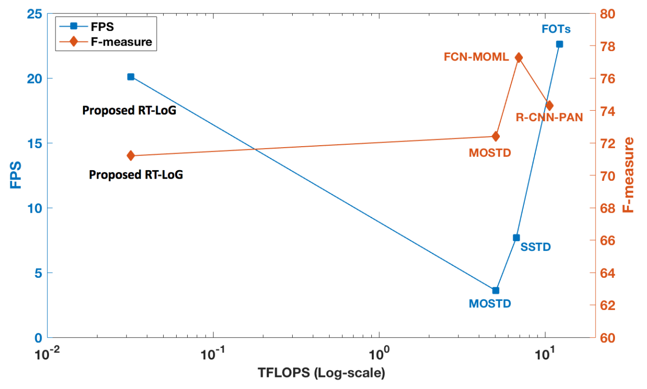


Fig. 22: The details of F-measure scores and FPS from scene text detection systems correspond to their architectures.

requiring a difference of two orders of magnitude in term of processing resources.

5 Conclusions and perspectives

This paper presents a novel RT-LoG operator for scene text detection. The proposed method achieves two levels of optimization. The first level takes advantage of box selection with mutualization within the $\widehat{DoG}$ product. The second level is a mixed spatial/scale-space method for box selection based on the linear distribution of Gaussian kernels in the scale-space domain.

Our results show that the proposed RT-LoG operator exhibits the best performance with a trade-off between speed and accuracy among all the operators in the literature. The processing is achieved with sharp upper and lower bounds on the execution times for predictability. The operator processes at approximately 30 FPS at the Quad-HD resolution on a regular CPU architecture with a low-level latency. A 25 FPS can be reached with a null latency.

In addition, our operator provides meaningful spatial, scale-space and contrast information compared to the other operators in the literature. This results in a strong optimization and support of a full system for scene text detection. With a proper system, our operator is competitive in comparison with contributions of the literature using the end-to-end CNN/GPU based systems, while processing with a difference of two orders of magnitude in term of processing resources. The proposed approach is able to process on a low-cost hardware architecture with a high frame rate while keeping a strong and competitive accuracy for detection.

Some perspectives can be further explored. The given operator is not robust to illumination changes, which is a key problem for scene text detection. Thus, a contrast invariant operator with real-time implementation should be considered. This type of implementation would optimize the precision of the operator for detection, which remains as a challenge in the literature [44]. The sampling in the spatial/scale-space domains could be further optimized. This information needs to be elaborated in a sampling strategy related to detection problems [6]. This strategy could result in a strong time optimization of the operator but will relax the predictability of processing. It will be more dedicated to scene text detection in embedded systems with weak constraints for the soft real-time processing.

References

1. Q. Ye, D. Doermann, Text detection and recognition in imagery: A survey, PAMI, 37.7, 1480-1500 (2015).
2. S. Long, X. He, C. Ya, Scene Text Detection and Recognition: The Deep Learning Era, arXiv:1811.04256 (2018)
3. N. Nayef, F. Yin, I. Bizid, H. Choi, ICDAR2017 Robust Reading Challenge on Multi-Lingual Scene Text Detection and Script Identification-RRC-MLT, ICDAR, 1454-1459 (2017)
4. L. Neumann, J. Matas, Real-time lexicon-free scene text localization and recognition, PAMI, 38.9,1872-1885 (2016)
5. G.C. Buttazzo, Hard real-time computing systems: predictable scheduling algorithms and applications, Springer Science & Business Media, 24 (2011)

6. I. Rey-Otero, J.M. Morel, An analysis of scale-space sampling in SIFT, ICIP, 4847–485 (2014)
7. M. Busta, L Neumann, J Matas, Fastext: Efficient unconstrained scene text detector, ICCV, 1206-1214 (2015)
8. H. Cho, M. Sung, B. Jun, Canny text detector: Fast and robust scene text localization algorithm, CVPR, 3566-3573 (2016)
9. B. Epshtein, E. Ofek, Detecting text in natural scenes with stroke width transform, CVPR, 2963–2970 (2010)
10. X. Girones, C. Julia, Real-Time Text Localization in Natural Scene Images Using a Linear Spatial Filter, ICDAR, 1261-1268 (2017)
11. L. Gomez, D. Karatzas, MSER-based real-time text detection and tracking, ICPR, 3110-3115 (2014)
12. H. Turki, M.B. Halima, A. Alimi, Text Detection based on MSER and CNN Features, ICDAR, 949-954 (2017)
13. R. Zhao, X. Niu, Y. Wu, W. Luk, Q. Liu, Optimizing CNN-based object detection algorithms on embedded FPGA platforms, ISARC, 255-267 (2017)
14. T.J. Maceina, G. Manduchi , Assessment of general purpose GPU systems in real-time control, TNS , 64.6, 1455–1460 (2017)
15. H. Kim, H. Nam, W. Jung, J. Lee, Performance analysis of CNN frameworks for GPUs, ISPASS, 55–64 (2017)
16. F. Wang, L. Zhao, X. Li, X. Wang, Geometry-aware scene text detection with instance transformation network, CVPR, 1381–1389 (2018)
17. V. Fragoso, G. Srivastava, A. Nagar, Z. Li, Cascade of box (CABOX) filters for optimal scale space approximation, CVPR, 126-131 (2014)
18. Y. Liu, D. Zhang, Y. Zhang, Real-time scene text detection based on stroke model, ICPR, 3116-3120 (2014)
19. D.C. Nguyen, M. Delalandre, D. Conte, T.A. Pham, Performance Evaluation of Real-time and Scale-invariant LoG Operators for Text Detection, VISAPP, 344-353 (2019).
20. T. Lindeberg, Scale-space theory: A basic tool for analysing structures at different scales, JAS, 21.224–270 (1994)
21. D. Charalampidis, Recursive implementation of the Gaussian filter using truncated cosine functions, TIP, 64.14. 3554–3565 (2016)
22. E. Elboher, M. Werman, Efficient and accurate Gaussian image filtering using running sums, ISDA, 897–902 (2011)
23. P. Viola, MJ. Jones, Robust real-time face detection, IJCV, 57.2, 137-154 (2004)
24. G. Strang, Introduction to Linear Algebra, Cambridge Press, 5th edition (1993)
25. D. Karatzas, L. Gomez-Bigorda, ICDAR 2015 competition on robust reading, ICDAR, 1156–1160 (2015)
26. K. Siddhesh, A. Amit, Faster K-Means Cluster Estimation, arXiv, vol.1701.04600 (2017)
27. G.G. Medioni J. Lim, J. Park, Text segmentation in color images using tensor voting. Image and Vision Computing , IVC, 25.5, 671– 685 (2007)
28. J. Mao, H. Li, W.Zhou, S.Yan, Q. Tian, Scale based region growing for scene text detection, ACMMM, 1007-1016 (2013)
29. W. Zhu, J. Lou, L. Chen, Q. Xia, M. Ren, Scene text detection via extremal region based double threshold convolutional network classification, PloS one, 12.8: e0182227 (2017)
30. X.C. Yin, W.Y. Pei, J. Zhang, Multi-orientation scene text detection with adaptive clustering, PAMI, 37.9, 1930-1937 (2015)
31. J. Dai, Z Wang, X. Zhao, S. Shao, Scene text detection based on enhanced multi-channels MSER and a fast text grouping process, ICCCBDA, 351-355 (2018)
32. C. Nguyen, M. Delalandre, D. Conte. T. Pham, Fast Scene Text Detection with RT-LoG Operator and CNN, VISAPP, (2020)
33. J. Liu , X. Liu,J. Sheng, D. Liang, Pyramid Mask Text Detector, arXiv preprint:1903.11800 (2019)
34. W. He, X.Y. Zhang, F Yin, CL Liu, Multi-oriented and multi-lingual scene text detection with direct regression, TIP, 27.11, 5406–5419 (2018)
35. Z. Huang, Z. Zhong, L. Sun, Q. Huo, Mask R-CNN with pyramid attention network for scene text detection, WACV, 764–772 (2019)
36. C. Zhang , B. Liang , Z. Huang, M. En , J. Han, Look More Than Once: An Accurate Detector for Text of Arbitrary Shapes, arXiv preprint:1904.06535 (2019)
37. P. Lyu, C. Yao, W. Wu, S. Yan, Multi-oriented scene text detection via corner localization and region segmentation, CVPR, 7553–7563 (2018)
38. X. Liu , D. Liang ,S.Yan, D. Chen, Fots: Fast oriented text spotting with a unified network, CVPR, 5676–5685 (2018)
39. Z. Zhong, L. Sun, Q. Huo, An anchor-free region proposal network for faster r-cnn based text detection approaches, arXiv preprint:1804.09003 (2018)
40. H. Wang , X. Rong ,Y. Tian, Towards Accurate Instance-Level Text Spotting with Guided Attention, ICME, 994–999 (2019)
41. P. Lyu , M. Liao, C .Yao, W. Wu, Mask TextSpotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes, ECCV (2018)
42. X. Zhou, C. Yao,H. Wen, Y. Wang, EAST: an efficient and accurate scene text detector, CVPR, 5551–5560 (2017)
43. P. He, W. Huang, T. He, Q. Zhu, Single shot text detector with regional attention, ICCV, 3047–3055 (2017)
44. Z. Miao, X. Jiang, Contrast invariant interest point detection by zero-norm log filter, TIP, 25.1,331-342 (2016)